Jerzy Józefczyk

Wybrane problemy podejmowania decyzji w kompleksach operacji

Oficyna Wydawnicza Politechniki Wrocławskiej
Wrocław 2001

Monografia została opracowana w ramach realizacji projektu badawczego nr 7 T11A 039 20 pt. *Algorytmy podejmowania decyzji i uczenia w systemach niepewnych i kompleksach operacji z reprezentacją wiedzy*, finansowanego przez Komitet Badań Naukowych.

Spis treści

Wykaz podstawowych oznaczeń

$\boldsymbol{x}$ – wektor stanu: w alokacji – rozmiar operacji, w sterowaniu jazdą – położenia i prędkości realizatorów

$\boldsymbol{x}_0, \boldsymbol{x}^*, \boldsymbol{X}$ – wektor stanów początkowych, wektor stanów końcowych, przestrzeń stanu

$\boldsymbol{R}, R, r$ – zbiory, liczby, indeksy operacji oraz zadań

$\underline{t}, \overline{t}, T$ – moment rozpoczęcia, moment zakończenia, czas trwania zdarzenia, np. moment rozpoczęcia, moment zakończenia, czas wykonania operacji w alokacji

$\boldsymbol{u}$ – wektor zmiennych decyzyjnych: w rozdziale zasobów – wektor wielkości zasobów, w sterowaniu jazdą – wektor wielkości sterujących

$\boldsymbol{v}$ – wektor wielkości zadań w rozdziale zadań

$\boldsymbol{\rho}$ – relacja określająca ograniczenia kolejnościowe w zbiorze operacji $\boldsymbol{R}$

T – czas wykonania kompleksu operacji

$\gamma_r, \overline{\gamma}_r, \widetilde{\gamma}_r$ – modele operacji w alokacji

U – globalna ilość zasobów

$\boldsymbol{D}_u$ – zbiór dopuszczalnych rozdziałów zasobów

V – globalna ilość zadań

$\boldsymbol{D}_v$ – zbiór dopuszczalnych rozdziałów zadań

$\boldsymbol{H}, H, h$ – zbiór, liczba, indeks zadań lub stanowisk

$\boldsymbol{\tau}_h$ – wektor czasów wykonania zadań

$\tau_{r;h}$ – czas wykonania zadania h przez realizator r

$\overline{\tau}_{r;h}$ – czas wykonania przez realizator r czynności na stanowisku h

$\hat{\tau}_{r;g,h}$ – czas dojazdu realizatora r do stanowiska h

d_h – żądany termin wykonania zadania h

Q – kryterium jakości szeregowania lub obsługi zadań

$\kappa, \kappa_{\mathrm{I}}, \kappa_{\mathrm{II}}$ – oszacowania najgorszego przypadku dla przybliżonych algorytmów rozwiązania

$\boldsymbol{c}, \boldsymbol{\gamma}, \widetilde{\boldsymbol{\gamma}}$ – macierze decyzyjne w szeregowaniu z uwzględnieniem ruchu realizatorów dla kryterium w postaci długości uszeregowania

$\boldsymbol{C}, \boldsymbol{\Gamma}, \widetilde{\boldsymbol{\Gamma}}$ – zbiory macierzy decyzyjnych w szeregowaniu z uwzględnieniem ruchu realizatorów dla kryterium w postaci długości uszeregowania

$\boldsymbol{a}, \boldsymbol{\alpha}, \widetilde{\boldsymbol{\alpha}}$ – macierze decyzyjne w szeregowaniu z uwzględnieniem ruchu realizatorów dla kryterium w postaci maksymalnego opóźnienia

$\mathbf{A}, \boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}}$ – zbiory macierzy decyzyjnych w szeregowaniu z uwzględnieniem ruchu realizatorów dla kryterium w postaci maksymalnego opóźnienia

n, v – numery taktów podejmowania decyzji (sterowania)

$\boldsymbol{z}(n)$ – macierz stanu w takcie n (w szeregowaniu zadań z uwzględnieniem ruchu realizatorów)

$\boldsymbol{z}(0)$, $\boldsymbol{z}(N)$, $\boldsymbol{Z}$ – macierz stanów początkowych, macierz stanów końcowych, przestrzeń stanu

$\boldsymbol{v}(n)$ – macierz decyzji w takcie n

$\varphi_{r,h}$ – model obiektu

δ – wskaźnik jakości rozwiązania

$\boldsymbol{Y}_{r,v}$ – obszar dopuszczalny dla realizatora r w takcie v

$\boldsymbol{D}_{s,v}$ – obszar zajęty przez realizator s w takcie v

$\bar{\boldsymbol{D}}_{g}$ – obszar zajęty przez przeszkodę stałą g

$q_{r,v}$ – lokalne kryterium jakości sterowania jazdą

$\boldsymbol{M}_r$, M_r, $m_r(j)$ – trasa, długość trasy, j-ty element trasy realizatora r

η – numer iteracji

$z_{k,n}$ – n-te zadanie technologiczne wykonywane na obiekcie typu k

$\zeta_{k,j}$ – j-te zadanie transportowe dla obiektu typu k

w – indeks środka transportu

$\overline{o}_{k,m,n}$ – operacja technologiczna

$o_{k,m,n}$ – operacja transportowa

D – długość drogi przebytej przez środek transportu

$\boldsymbol{x}'$, $\bar{\boldsymbol{x}}$ – macierze decyzyjne w zagadnieniach transportowych

$\boldsymbol{c}'$, $\bar{\boldsymbol{c}}$ – macierze kosztów transportu

J – wskaźnik jakości w alokacji z uwzględnieniem transportu lub wskaźnik porównania algorytmów w szeregowaniu z uwzględnieniem ruchu realizatorów

$\boldsymbol{w}_r$ – wektor wag w algorytmie neuropodobnym

$\hat{k}_r$ – niepewny parametr w problemie alokacji

$\boldsymbol{I}_j$ – j-ta iteracja (populacja) w algorytmie ewolucyjnym

p_1 – prawdopodobieństwo krzyżowania

p_2 – prawdopodobieństwo mutacji

$\boldsymbol{\alpha}_x$, $\boldsymbol{\alpha}_\mathrm{D}$ – ciągi formuł elementarnych

$\mathrm{F}(\boldsymbol{\alpha}_x, \boldsymbol{\alpha}_\mathrm{D})$ – fakty (reprezentacja wiedzy)

$\mathrm{F}_x(\boldsymbol{\alpha}_x)$ – własność wejściowa

Wstęp

Kompleksy operacji są ważną klasą systemów, których części składowe, czyli operacje mają swój początek, koniec i czas realizacji, a struktura wynika z różnych uwarunkowań czasowych między operacjami, przede wszystkim z tego, że pewne operacje nie mogą się rozpocząć przed zakończeniem innych. Analiza, a zwłaszcza podejmowanie decyzji dla tej klasy systemów, nazywanych również systemami operacyjnymi, są rozwijane od wielu lat. Stanowią one istotny składnik badań operacyjnych. Szczegółowe metody i algorytmy są opracowywane i prezentowane także w różnych publikacjach z zakresu badań systemowych, automatyki i informatyki, ale również stanowią przedmiot badań modelowania i identyfikacji, a przede wszystkim sterowania, a szerzej – podejmowania decyzji.

Stały rozwój tej problematyki jest spowodowany wzrostem zainteresowania praktycznymi zastosowaniami, a więc koniecznością rozpatrywania nowych, bardziej skomplikowanych zagadnień, oraz wzrostem możliwości realizacji opracowanych metod i algorytmów w związku z gwałtownym ostatnio rozwojem środków informatyki. Chodzi przede wszystkim o zastosowania w dyskretnych systemach produkcyjnych, a zwłaszcza w systemach komputerowo zintegrowanego wytwarzania i w elastycznych systemach produkcyjnych, ale także w złożonych systemach informatycznych. Problematyka badawcza dotycząca kompleksów operacji jest bardzo obszerna i była prezentowana w wielu specjalistycznych opracowaniach, np. [1, 113, 111, 6, 5]. Celem przedstawianej monografii nie jest prezentacja całokształtu tej problematyki. Jej zakres jest zawężony do takiej klasy dyskretnych systemów zdarzeniowych, w której jest uwzględniany ruch lub transport różnych elementów takich systemów. W odniesieniu do elastycznych systemów produkcyjnych prezentowany materiał dotyczy podstaw teoretycznych, tak zwanych systemów planowania przepływów produkcji oraz przepływów pracy, a bardziej ogólnie – systemów logistycznych. Rozważania ograniczono do problemów decyzyjnych, a zwłaszcza do zagadnień sterowania, które były przedmiotem prac Autora bądź były opracowywane na Politechnice Wrocławskiej i w Instytucie Badań Systemowych PAN, gdzie autor pracuje. Początkowo prace koncentrowały się

wokół rozdziału między operacje ograniczonych zasobów lub zadań i były uogólnieniem klasycznych zagadnień analizy czasowej systemów operacyjnych, dla których opracowano i wykorzystywano znane metody PERT/CPM. Rozpatrywanie przypadków i charakterystyk zasobów oraz różnych modeli matematycznych operacji doprowadziło w konsekwencji do sformułowania nowego kierunku w badaniach operacyjnych, tzw. sterowania rozdziałem zasobów i zadań w kompleksach operacji. Łączą się z tym ściśle podstawowe problemy szeregowania, a także zagadnienia z zakresu teorii systemów masowej obsługi, w których najważniejsza informacja o strumieniu operacji dotyczy rozkładu prawdopodobieństwa czasu między pojawieniem się sąsiednich operacji. Kolejny ważny etap rozwoju badań kompleksów operacji polegał na uwzględnieniu w badaniach ruchu realizatorów wykonujących operacje i obiektów, na których są wykonywane operacje, a także transportu zasobów niezbędnych do wykonania operacji. Koncentrowano się na wykorzystaniu w podejmowaniu decyzji dla kompleksów operacji wybranych metod i algorytmów sztucznej inteligencji.

W pracy zaprezentowano podstawowe problemy podejmowania decyzji, a zwłaszcza sterowania kompleksami operacji (rozdz. 1). Po wprowadzeniu tej klasy systemów przedstawiono podstawowe wyniki z zakresu czasowo-optymalnego sterowania rozdziałem zasobów i zadań dla kompleksów operacji o dowolnej strukturze scharakteryzowanych tak zwanymi modelami czasowymi. Rozważania te, dotyczące przypadku deterministycznego, czyli pełnej i pewnej informacji o systemie, uzupełniono rozpatrzeniem przypadku probabilistycznego, w którym niektóre informacje o operacjach są dane w postaci określonych funkcji rozkładów prawdopodobieństwa. W szeregowaniu zadań ograniczono się do przypadku realizatorów równoległych (jednofunkcyjnych) – omówiono podstawowe zagadnienia związane z wyznaczaniem optymalnych i przybliżonych algorytmów szeregowania, a rozważania uzupełniono przedstawieniem przypadku, gdy czasy wykonywania zadań są realizacjami zmiennych losowych. W wyborze do prezentacji podstawowych problemów podejmowania decyzji w kompleksach operacji kierowano się potrzebą przypomnienia Czytelnikowi tych treści, które są niezbędne do zrozumienia zagadnień omawianych w dalszych rozdziałach monografii. Wprowadzono również w problematykę podejmowania decyzji w systemach masowej obsługi, które, jak już wspomniano, mogą być traktowane jako ważny przypadek kompleksów operacji – istotny przede wszystkim do zastosowania w systemach informatycznych oraz w dyskretnych systemach produkcyjnych.

W kolejnym rozdziale nawiązano do znanego klasycznego zagadnienia szeregowania zadań niezależnych i niepodzielnych na realizatorach dowolnych. Opi-

sywane rozszerzenie polega na uwzględnieniu ruchu realizatorów, czyli podmiotów wykonujących zadania. Skoncentrowano się na omówieniu efektywnych, przybliżonych algorytmów szeregowania dla dwóch kryteriów jakości szeregowania, a mianowicie długości uszeregowania i maksymalnego opóźnienia. Dla pierwszego z kryteriów przedstawiono również algorytm dokładny o wykładniczej złożoności obliczeniowej, wykorzystujący zasadę podziału i ograniczeń i oparty na przedstawieniu rozważanego kompleksu operacji jako systemu zdarzeniowo-dynamicznego. Zwrócono też uwagę na ważne zagadnienie symulacyjnego badania własności uzyskanych algorytmów szeregowania, istotne zwłaszcza wtedy, gdy brak jest pełnej analitycznej oceny wyznaczonych algorytmów. Przedstawione podstawy badań symulacyjnych oraz porównania różnych algorytmów rozwiązania dla tego samego problemu mają znaczenie wykraczające poza prezentowany przypadek szeregowania zadań i mogą być wykorzystane do badań symulacyjnych dla innych algorytmów podejmowania decyzji. Prezentację dla przypadku deterministycznego uzupełniono rozważeniem przypadku probabilistycznego, w którym przyjęto, że czasy wykonywania zadań są realizacjami odpowiednich niezależnych zmiennych losowych.

Ważnym i charakterystycznym dla monografii aspektem problematyki podejmowania decyzji w kompleksach operacji, a przede wszystkim szeregowania, jest ruch oraz transport różnych elementów systemu (rozdz. 3). Rozważania metodologiczne są odnoszone do dyskretnych systemów produkcyjnych. Dlatego ruch dotyczy realizatorów wykonujących zadania lub obiektów, na których te zadania są wykonywane, a transportowanymi elementami systemów produkcyjnych są surowce, zasoby oraz produkty. Uwzględnienie ruchu oraz transportu w rozważanych zagadnieniach decyzyjnych dla kompleksów operacji prowadzi do złożonych problemów sterowania. Po wprowadzeniu klasyfikacji problemów podejmowania decyzji w kompleksach operacji z uwzględnieniem ruchu oraz transportu zaprezentowano i rozwiązano powiązane ze sobą problemy szeregowania, których wynikami są trasy przejazdów realizatorów lub środków transportu przewożących obiekty oraz problemy sterowania bezkolizyjną jazdą grupy poruszających się pojazdów. W tym przypadku rozważane problemy decyzyjne mają charakter mieszany, to znaczy dyskretno-ciągły i tworzą dwupoziomowe systemy sterowania kompleksami operacji. Dla rozpatrywanych łącznie zagadnień alokacji oraz transportu surowców i(lub) produktów, oprócz algorytmów rozwiązania przedstawiono porównanie rozwiązań z uwzględnieniem transportu oraz bez transportu.

Omówiono przykładowe wyniki najnowszych prac (rozdz. 4), które dotyczą zastosowania wybranych metod sztucznej inteligencji (sztucznych sieci neurono-

wych, algorytmów ewolucyjnych oraz rozpoznawania z reprezentacją wiedzy w postaci wyrażeń logicznych) do podejmowania decyzji w niektórych, omówionych wcześniej, zagadnieniach dla kompleksów operacji. Poruszane zagadnienia badawcze mogą stanowić inspirację do dalszych, pogłębionych badań i z pewnością w niedługim czasie doczekają się bardziej wszechstronnego podsumowania. Szczególnie interesujące i perspektywiczne są próby nieklasycznego uwzględniania niepewności w opisach kompleksów operacji oraz wykorzystywanie algorytmów uczenia w procesach uaktualniania informacji o badanym systemie. Pierwsze prace z tego zakresu przedstawiono dla rozdziału zadań w przypadku modeli kompleksu operacji z niepewnym parametrem. Ważne w metodach sztucznej inteligencji algorytmy uczenia znalazły również zastosowanie w opracowywaniu ewolucyjnego algorytmu rozwiązania jednego z przedstawionych w rozdziale 2. zagadnień szeregowania. Do rozwiązania trudnego problemu sterowania bezkolizyjną jazdą ruchomych realizatorów zastosowano nietradycyjny algorytm rozpoznawania, w którym wiedza o rozpoznawanych obiektach jest wyrażona w postaci wyrażeń logicznych, podawanych przez eksperta.

Rezultaty pracy podsumowano oraz wskazano kierunki dalszych badań w rozdziale 5.

Przedstawiony materiał tworzy określoną całość, a poszczególne rozdziały nie są elementami zbioru odrębnych, niezależnych części. Zasadniczym motywem stanowiącym podstawę wyboru problematyki oraz określenia kompozycji pracy jest uwzględnienie ruchu w różnych tradycyjnych problemach decyzyjnych dla kompleksów operacji (rozdz. 3). Ważny fragment problematyki, który dotyczy ruchu w kompleksach operacji, został specjalnie potraktowany i obszernie zaprezentowany w rozdziale 2. Szeregowanie zadań z uwzględnieniem ruchu realizatorów jest zagadnieniem decyzyjnym, które w dwupoziomowym systemie sterowania kompleksem operacji z ruchomymi realizatorami ulokowano na poziomie górnym. Ponadto zamieszczenie rozdz. 1 powoduje, że praca prezentuje wybrane zagadnienia w sposób całościowy i nie wymaga od Czytelnika wstępnych studiów z zakresu kompleksów operacji.

Monografia jest przeznaczona w zasadzie dla pracowników naukowych zajmujących się zagadnieniami badań operacyjnych i teorii sterowania oraz podejmowania decyzji zwłaszcza dla systemów dyskretnych, ale może być również przydatna dla szerszego grona specjalistów zainteresowanych wykorzystaniem algorytmów sterowania i szeregowania w nowoczesnych dyskretnych systemach produkcyjnych oraz w systemach informatycznych. Praca nie wymaga dodatkowej specjalistycznej wiedzy z zakresu ogólnie rozumianych badań operacyjnych, co czyni mono-

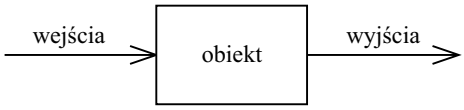
grafię przydatną również dla studentów wyższych lat studiów ze specjalności związanych z automatyką, informatyką, czy badaniami systemowymi.

Autor pragnie serdecznie podziękować Kolegom z Instytutu Sterowania i Techniki Systemów Politechniki Wrocławskiej za uwagi i możliwość przedyskutowania wielu zagadnień prezentowanych w pracy w trakcie ich opracowywania, a przede wszystkim prof. Zdzisławowi Bubnickiemu za wsparcie swoją zachętą i radą, co przyczyniło się do powstania niniejszej monografii.

1. Podstawowe problemy podejmowania decyzji w kompleksach operacji

1.1. Wprowadzenie

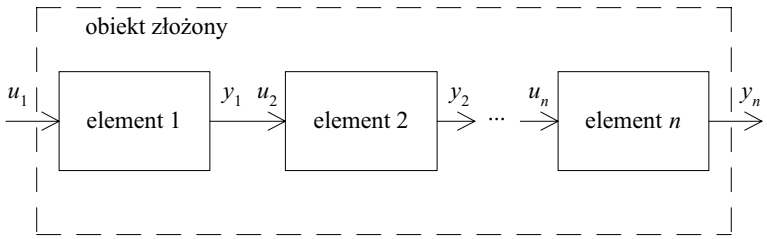
Różnorodność rozpatrywanych rzeczywistych obiektów sprawia, że nie można ich opisywać w sposób jednolity. Obiekty, które są przedmiotem rozważań w szeroko rozumianych badaniach systemowych, oprócz różnego stopnia złożoności, różnią się przede wszystkim naturą zachodzących w nich zjawisk i procesów. Oprócz najczęściej rozpatrywanych różnych obiektów technicznych są rozważane obiekty ekonomiczne, finansowe, biologiczne, medyczne i inne. Spośród różnych możliwych i stosowanych klasyfikacji obiektów należy wyróżnić podział na obiekty wejściowo-wyjściowe i obiekty typu kompleks operacji. Charakterystyczną cechą obiektów wejściowo-wyjściowych jest możliwość wyróżnienia wejść i wyjść. Graficzną ilustrację przedstawiono na rys. 1.1.



Rys. 1.1. Obiekt wejściowo-wyjściowy

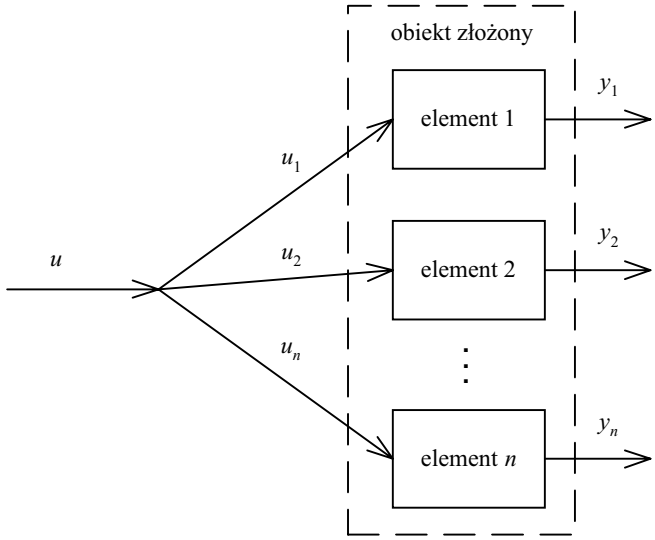
Do tej klasy można zaliczyć obiekty, w których zachodzą procesy polegające na przepływie na przykład materiałów, energii, pieniędzy, informacji. Przykładem może być reaktor chemiczny, do którego należy dostarczyć na wejście związki chemiczne i energię po to, aby po pewnym czasie na wyjściu otrzymać produkt reakcji. Następuje tu przepływ zarówno materiałów, jak i energii. Obiektem wejściowo-wyjściowym jest układ oddechowy, w którym przepływ powietrza przez płuca można przyjąć za wejście, a jego ciśnienie za wyjście. Mamy tu do czynienia ze specyficznym przepływem materiałowym. W obiekcie zarządzania, jakim jest na przykład przedsiębiorstwo przemysłowe lub jego część, na wielkość produkcji będącą wyj-

ściem mają wpływ bieżące wydatki, które mogą być traktowane jako wejście. Przykładem obiektu, w którym następuje przepływ informacji jest sieć informatyczna, gdzie są dostarczane informacje w postaci zgłoszeń do obsługi, a jako wyjście jest przyjmowana kolejka tych zgłoszeń. W kontekście zasadniczego celu tej wstępnej części rozważań, który polega na scharakteryzowaniu obiektów należących do drugiej klasy, czyli obiektów typu kompleks operacji, zwróćmy uwagę jeszcze na inną cechę obiektów wejściowo-wyjściowych. Jest nią złożoność obiektu. Można mówić o obiekcie prostym lub złożonym. Obiekt prosty to taki, który ma tylko wejścia zewnętrzne (z otoczenia) i wyjścia zewnętrzne (do otoczenia). Na rysunku 1.1 przedstawiono graficznie obiekt prosty. Obiekt złożony składa się z elementów i ma wewnętrzną strukturę, która określa sposób powiązania elementów. Spośród wielu możliwych sposobów powiązania elementów (struktur) najbardziej typowa jest struktura szeregowa (rys. 1.2) i równoległa (rys. 1.3). Na obu rysunkach przedstawiono obiekty składające się z n elementów. Bieżący, czyli i-ty element, ma wejścia oznaczane jako u_i oraz wyjścia oznaczane jako y_i, i = 1, 2, ..., n.



Rys. 1.2. Złożony obiekt wejściowo-wyjściowy o strukturze szeregowej

W obiekcie złożonym o strukturze szeregowej wyjścia wewnętrzne elementów są jednocześnie wejściami kolejnych elementów, tzn. $u_i = y_{i-1}$, i = 2, 3, ..., n. Poza tym obiekt ten ma wejście zewnętrzne u_1 i wyjście zewnętrzne y_n. W obiekcie złożonym o strukturze równoległej wszystkie elementy mają takie samo wejście $u = u_i$, i = 1, 2, ..., n, będące wejściem zewnętrznym. Struktury te mogą być podstawą do tworzenia bardziej złożonych struktur pochodnych, na przykład równoległego połączenia struktur szeregowych (struktura równoległo-szeregowa) lub szeregowego połączenia struktur równoległych (struktura szeregowo-równoległa). Poza tym w przedstawionych strukturach podstawowych mogą występować dodatkowe wejścia i wyjścia zewnętrzne – w przypadku struktury szeregowej oraz dodatkowe wejścia i wyjścia wewnętrzne – w przypadku struktury równoległej. W obiektach o różnych strukturach mogą też występować tak zwane sprzężenia zwrotne, to znaczy połą-

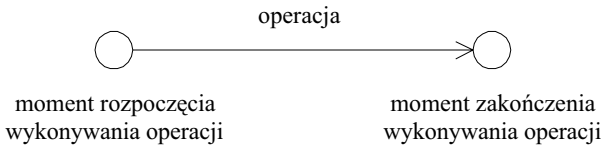


Rys. 1.3. Złożony obiekt wejściowo-wyjściowy o strukturze równoległej

czenia wyjść pewnych elementów z wejściami elementów występujących wcześniej. Przykładem złożonego obiektu wejściowo-wyjściowego o strukturze szeregowej jest kaskada reaktorów chemicznych, w której związki chemiczne będące w trakcie realizacji są przekazywane między kolejnymi elementami kaskady.

W systemach o różnej naturze występuje wiele zjawisk i procesów, których nie można przedstawić, posługując się opisem przepływów, między innymi materiałów, energii, informacji. Często istotne są zależności czasowe zachodzące w procesie, a trudno się w nim dopatrzyć występowania przepływów w takim sensie, w jakim występują one w obiektach wejściowo-wyjściowych. Przykładowo w systemie informatycznym, w którym z wykorzystaniem danego zbioru jednostek obliczeniowych oraz innych zasobów typu pamięć, urządzenia peryferyjne, należy zrealizować zestaw procedur (na przykład obliczeniowych) – istotny może być czas wykonania wszystkich procedur. Czas ten zależy od kolejności ich wykonywania oraz od zasobów przydzielanych poszczególnym procedurom. Podobnie w dyskretnym systemie produkcyjnym, w którym wytworzenie obiektu wymaga wykonania określonego zestawu czynności produkcyjnych z wykorzystaniem dostępnych maszyn (realizatorów), narzędzi, materiałów i innych zasobów – ważny może być koszt lub czas wykonania obiektu, np. [18]. Tak jak w przypadku systemu informatycznego na wspomniany koszt (czas) mają wpływ wielkości przydzielonych zasobów i ewentualnie

kolejność wykonywania czynności produkcyjnych, jeśli podlega ona wyborowi. Jeśli wspomniany obliczeniowy system informatyczny lub dyskretny system produkcyjny mają być przedmiotem analizy lub podejmowania decyzji, to należy jasno określić obiekt takich działań. Jak łatwo dostrzec, obiektem takim jest w pierwszym przypadku zestaw procedur obliczeniowych, a w drugim – zestaw czynności produkcyjnych. Podstawowym elementem jest wielkość nazywana o p e r a c j ą. Nie ma ona formalnej definicji. Jest natomiast charakteryzowana przez podanie innych wielkości i zmiennych. Odpowiedni i pełny opis przedstawiono w dalszej części rozdziału, natomiast w tym miejscu zauważmy jedynie, że istotnymi wielkościami charakteryzującymi operację są: czas jej wykonania oraz momenty rozpoczęcia i zakończenia jej realizacji. Można dostrzec pewną analogię między operacją a obiektem wejściowo-wyjściowym, polegającą na tym, że w obu przypadkach mogą być one traktowane jako elementy większej całości. Graficzną postać operacji, jedną z możliwych do stosowania, przedstawiono na rys. 1.4. Operacja jest przedstawio-



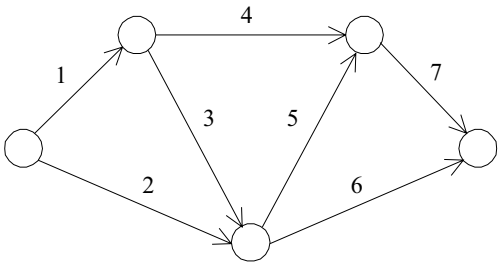
Rys. 1.4. Operacja

na w postaci łuku skierowanego, umieszczonego między wierzchołkami. Wierzchołek, z którego łuk wychodzi, oznacza moment czasu, w którym realizacja operacji się rozpoczyna, a drugi z wierzchołków – moment czasu, w którym realizacja operacji się kończy. Jeśli obiekt wejściowo–wyjściowy może być przedmiotem odrębnych rozważań, to w przypadku operacji celowe jest jedynie łączne rozpatrywanie wszystkich operacji. Operacje takie nie są zwykle niezależne. Sposób ich powiązania jest odmienny niż w przypadku złożonego obiektu wejściowo–wyjściowego. Istotne są tu momenty czasu, w których operacje się rozpoczynają i kończą. Powiązanie lub inaczej uzależnienie od siebie dwóch operacji polega na tym, że moment rozpoczęcia wykonywania drugiej z nich nie może być wcześniejszy niż moment zakończenia wykonywania pierwszej. O takiej sytuacji mówimy, że wykonanie operacji drugiej musi być poprzedzone wykonaniem operacji pierwszej. Interpretacja takiej zależności kolejnościowej dla przedstawionych dwóch przykładów poglądowych jest oczywista. Kolejność w zestawach operacji obliczeniowych i produkcyjnych często występuje i wynika z reżimów technologicznych. Odnosząc znowu roz-

ważania do złożonego obiektu wejściowo-wyjściowego, można stwierdzić, że charakteryzowany zestaw operacji jest również obiektem złożonym, którego elementami są pojedyncze operacje. Obiekt ten ma swoją strukturę określoną na zasadzie kolejności czasowych i jest nazywany k o m p l e k s e m o p e r a c j i [28, 29].

Kompleks operacji jest obiektem złożonym, którego elementami są operacje, powiązane ze sobą na zasadzie kolejności czasowych, to znaczy rozpoczęcie wykonywania niektórych operacji może się rozpocząć po zakończeniu wykonywania innych operacji.

Graficznym sposobem przedstawienia kompleksu operacji może być graf składający się z łuków i wierzchołków. Łuki przedstawiają operacje, a wierzchołki momenty czasu (rys. 1.5). Należy wyraźnie podkreślić, że typ obiektu, w którym zachodzą określone procesy czy zjawiska, nie zależy od ich natury, ale od cech, które są ważne dla rozważań dotyczących obiektu i są wyeksponowane w obiekcie. Nie można na przykład zakładać, że procesy chemiczne zachodzące w powiązanych ze sobą reaktorach chemicznych mogą się wyłącznie nadawać do opisu w postaci złożonego obiektu wejściowo-wyjściowego. Jeśli istotną cechą tego procesu byłby czas prowadzenia reakcji w poszczególnych reaktorach, a w konsekwencji czas wykonania wszystkich reakcji, to taki proces chemiczny byłby dobrym przykładem kompleksu operacji. Nie byłyby wtedy istotne przepływy materiałów i (lub) energii między reaktorami, ale na przykład czasy wykonywania reakcji.



Rys. 1.5. Przykład kompleksu operacji (operacje są oznaczone kolejnymi liczbami)

Scharakteryzowany obiekt może być przydatny do wyrażenia zjawisk i procesów o różnej naturze, nie tylko produkcyjnych i informatycznych. Jednak należy stwierdzić, że dyskretne procesy produkcyjne i systemy informacyjno-obliczeniowe – to najbardziej typowe i najlepiej zbadane obszary zastosowań kompleksów operacji. Dla tego typu obiektów jest formułowanych i rozwiązywanych wiele zagadnień analizy i podejmowania decyzji (w tym sterowania). Takie obszary badań jak

informatyka, automatyka, technika systemów, analiza systemowa czy wreszcie badania operacyjne obejmują w różnym stopniu również zagadnienia odnoszące się do kompleksów operacji.

W tym rozdziale przedstawiono wybrane tradycyjne problemy decyzyjne dla kompleksów operacji, których prezentacja jest uzasadniona ze względu na oryginalność i nowoczesność zagadnień będących treścią kolejnych rozdziałów. Najobszerniej omówiono rozdział zadań i zasobów. Problemy szeregowania zadań zostały ograniczone do przypadków, które są niezbędne do zrozumienia zaawansowanych zagadnień szeregowania z rozdziału 2. Rozważania są uzupełnione przeglądem również wybranych zagadnień decyzyjnych dla kompleksów operacji, ale dla warunków niedeterministycznych, w tym krótką prezentacją klasycznych zagadnień obsługi zadań. W większości przypadków omawiane problemy były opracowywane, a nawet inicjowane w ośrodku wrocławskim, a dokładniej w Zakładzie Systemów Sterowania Politechniki Wrocławskiej, co było jednym z kryteriów ich wyboru.

1.2. Problemy alokacji

W dalszym ciągu przyjmujemy, że najważniejszą wielkością charakteryzującą operację jest czas jej wykonania. Czas ten zależy od różnych wielkości i parametrów, które mają wpływ na wykonywanie operacji. Możliwość, a w konsekwencji czas wykonania operacji zależą od zasobów, które mogą być wykorzystywane w trakcie jej realizacji. Przykładem zasobu w informatycznym systemie obliczeniowym jest pamięć dostępna do wykonywania procedur obliczeniowych. W dyskretnym procesie przemysłowym zasobem może być energia elektryczna lub jej moc, zasilająca maszyny niezbędne do wykonania operacji technologicznych. Zasoby mogą mieć różny charakter. Mogą być podzielne w sposób ciągły, jak paliwo, moc, płynny katalizator, ale również podzielne w sposób dyskretny, jak maszyny, narzędzia. Ważną cechą zasobu jest możliwość jego odnawiania w trakcie wykonywania operacji. Przykładem może być wspomniana energia elektryczna, której jest stale tyle samo mimo zużywania w trakcie wykonywania operacji, jeśli jest ona dostarczana ze źródła o stałej mocy. Zasoby, które mają taką cechę, są nazywane zasobami odnawialnymi. Część zasobów nie ma tej cechy, wówczas przed rozpoczęciem wykonywania operacji jest dostępna określona ilość zasobu (na przykład narzędzi). W trakcie wykonywania operacji ubywa zasobu i nie ma źródła, z którego mógłby być uzupełniony ubytek. O takim zasobie mówimy, że jest nieodnawialny. Ogólnie, do wykonania operacji może być potrzebna pewna liczba różnych rodzajów zasobów. Na przykład wspomniana już energia elektryczna i narzędzia w dyskretnym procesie

przemysłowym, do których można jeszcze dodać materiały dodatkowe, palety, zamocowania. Jednym z zasobów – zwykle nie określanym tym mianem – jest realizator wykonujący operację. Realizatorem może być maszyna, procesor, ale także operator, czyli człowiek. Jeśli realizator jest zasobem, to może on podlegać rozdziałowi między operacje (zwykle są to zasoby o podziale dyskretnym). Jeśli realizator jest już przydzielony do operacji, to odrębnie może istnieć problem rozdziału zasobów w omawianym sensie. Przedstawione rozważania ograniczono do szczególnego przypadku, w którym przyjmujemy założenie, że do wykonania operacji jest potrzebny tylko jeden rodzaj zasobu. Tak więc zasób jako wielkość charakteryzująca operację jest skalarem.

Kolejną ważną wielkością określającą operację jest zadanie. Jest ono efektem wykonania operacji. Operację wykonuje się po to, aby osiągnąć pewien określony cel, wyrażony w postaci zadania. Zadanie jest w sposób naturalny wielkością skalarną i podobnie jak zasób może mieć charakter dyskretny lub ciągły. Posługując się używanym już poglądowym i bardzo ogólnym przykładem informatycznego systemu obliczeniowego, możemy zauważyć, że jeśli operacja polegałaby na realizacji procedur obliczeniowych przez daną jednostkę obliczeniową, to zadaniem jest zbiór takich procedur przydzielanych do jednostki, czyli operacji.

Inne informacje o operacji są zawarte w jej parametrach. Parametry są właściwe tylko dla danej operacji, a ich wartości nie zależą od parametrów i wielkości innych operacji. W poprzednim przykładzie takim parametrem mogłaby być prędkość jednostki obliczeniowej. Jej wartość jest stała i ma wpływ na czas wykonania operacji.

Dwie wymienione wcześniej wielkości, czyli zasób i zadanie zostały wyróżnione, ponieważ nie tylko tak jak parametry mają wpływ na czas wykonania każdej operacji z osobna, ale – co ważniejsze – są wielkościami opisującymi wszystkie operacje. Ponadto, występuje określona ilość zadań oraz ograniczona ilość zasobów, które należy rozdzielić między operacje. Powstaje zatem problem decyzyjny polegający na rozdziale, czyli alokacji zasobów i zadań do operacji w kompleksie operacji. Jako kryterium alokacji zwykle przyjmuje się koszt lub czas wykonania kompleksu operacji.

Wprowadźmy następujące oznaczenia. Niech $\boldsymbol{R} = \{1, 2, ..., R\}$ będzie zbiorem operacji o liczności R, a $r \in \boldsymbol{R}$ numerem (indeksem) oraz jednocześnie nazwą operacji. Bieżąca operacja r, jak to opisano, jest scharakteryzowana następującym zestawem wielkości i parametrów

$$r = \langle T_r, u_r, v_r, \mathrm{a}_r \rangle, \tag{1.1}$$

gdzie: T_r – czas wykonania operacji r,

u_r – zasoby przydzielone do realizacji operacji *r*,
v_r – zadanie przydzielone do wykonania w ramach operacji *r*,
$\boldsymbol{a}_r = [a_r^{(1)}, a_r^{(2)}, ..., a_r^{(K)}]^{\mathrm{T}}$ – *K*-elementowy wektor parametrów charakteryzujących operację *r*.

Wielkości T_r i v_r przyjmują wartości nieujemne ze zbioru liczb rzeczywistych, natomiast parametry tworzące wektor a_r są zmiennymi o wartościach ze zbioru liczb rzeczywistych. W ogólnym przypadku zasób może się zmieniać w trakcie wykonywania operacji. Jest więc funkcją rzeczywistą czasu, czyli zmiennej rzeczywistej nieujemnej – o wartościach nieujemnych. Dalej będzie wykorzystywany zapis $u_r(t)$ oznaczający wielkość zasobu przydzielonego operacji *r* w momencie czasu *t*, gdzie $t \in [\underline{t}_r, \bar{t}_r]$ oraz $\underline{t}_r, \bar{t}_r \geq 0$ jest odpowiednio momentem rozpoczęcia i zakończenia wykonywania operacji *r*.

Problem rozdziału zasobów oraz zadań przedstawiono oddzielnie. Zagadnienie jednoczesnego rozdziału zasobów i zadań nie będzie tu prezentowane.

1.2.1. Rozdział zasobów w kompleksie operacji

Model kompleksu operacji

Pierwszym etapem formalizacji problemu jest określenie modelu kompleksu operacji dla rozpatrywanego zagadnienia rozdziału zasobów. Należy dodać, że model taki zależy od zagadnienia, które jest formułowane dla kompleksu operacji. Model dla odrębnych zagadnień ma różną postać.

Najbardziej ogólnym opisem kompleksu operacji jest model w przestrzeni stanu. Stan jest definiowany jako rozmiar operacji dotychczas wykonanej. Przez rozmiar operacji należy rozumieć jej część wykonaną do rozpatrywanego momentu czasu, dla którego jest określona wartość stanu. Jest on wielkością skalarną oznaczaną jako x_r. Tak więc $x_r(t)$ jest rozmiarem operacji *r* wykonanej do momentu czasu *t*, gdzie $t \in [\underline{t}_r, \bar{t}_r]$. Przestrzeń stanu $\boldsymbol{X}_r \subset \boldsymbol{R}^+ \cup \{0\}$ jest podzbiorem liczb rzeczywistych nieujemnych. Wyróżniamy w niej stan początkowy $x_{r,0} \stackrel{\Delta}{=} x_r(t = \underline{t}_r)$ oraz stan końcowy $x_r^* \stackrel{\Delta}{=} x_r(t = \bar{t}_r)$. Wartość x_r^* jest nazywana rozmiarem całkowitym operacji *r*. W większości przypadków rozmiar całkowity ma interpretację zadania v_r, które jest wykonywane w ramach realizacji operacji *r* oraz w rozpatrywanym problemie nie podlega wyborowi tylko zostało dane a priori. Model operacji w przestrzeni stanu, czyli następujące równanie stanu, podaje zależność między wielkością zasobu $u_r(t)$ i wektorem parametrów $\boldsymbol{a}_r$ a prędkością zmian stanu $\dot{x}_r(t)$

$$\dot{x}_r(t) = \frac{dx_r(t)}{dt} = \varphi_r(x_r(t), u_r(t); \boldsymbol{a}_r). \tag{1.2}$$

Funkcja φ_r musi być ciągła ze względu na swe argumenty, ściśle rosnąca ze względu na $u_r(t)$ dla każdego $\boldsymbol{a}_r$ oraz przyjmować wartości nieujemne, przy czym $\varphi_r(x_r(t),0,\boldsymbol{a}_r)=0$. Wymaganie monotoniczności zapewnia wpływ wielkości przydzielonego zasobu na prędkość realizacji operacji. Jest to najbardziej ogólna postać modelu operacji, w której prędkość zmiany stanu zależy od wartości stanu w danej chwili i przydział zasobu do operacji może się zmieniać w trakcie jej wykonywania. W szczegółowych problemach rozdziału zasobów, które były rozpatrywane i dla których uzyskano algorytmy rozwiązania, stosowano modele będące szczególnymi przypadkami (1.2). Jednym z takich modeli jest zależność

$$\dot{x}_r(t)=\overline{\varphi}_r(u_r(t);\boldsymbol{a}_r), \tag{1.3}$$

w której prędkość zmiany stanu zależy od zmiennej w czasie wielkości przydzielonego zasobu. Problem rozdziału zasobów z modelem (1.3) był rozwiązywany m.in. w pracach [115, 5].

Model, dla którego uzyskano najwięcej rezultatów ma postać

$$T_r=\gamma_r(u_r;\boldsymbol{a}_r). \tag{1.4}$$

Jest to tak zwany model czasowy, podający zależność między stałą w trakcie wykonywania operacji wielkością przydzielonych zasobów a czasem wykonywania operacji. Zakładamy dalej, że funkcje γ_r są ciągłe, ściśle malejące ze względu na $\boldsymbol{u}_r$ dla każdego $\boldsymbol{a}_r$, przyjmują wartości nieujemne w zbiorze liczb rzeczywistych oraz prawdziwe są dla nich następujące warunki graniczne dla dowolnych $\boldsymbol{a}_r$

$$\lim_{u_r\to 0}\gamma_r(u_r;\boldsymbol{a}_r)=+\infty, \tag{1.5a}$$

$$\lim_{u_r\to +\infty}\gamma_r(u_r;\boldsymbol{a}_r)=0. \tag{1.5b}$$

Rozpatrzmy teraz kilka typowych przypadków i przykładów modeli operacji.

1. Szczególny przypadek (1.3), gdy wielkość zasobu jest stała w czasie, to znaczy $u_r(t)=u_r$. Wtedy

$$\dot{x}_r(t)=\overline{\varphi}_r(u_r;\boldsymbol{a}_r). \tag{1.6}$$

Po obustronnym scałkowaniu (1.6) w przedziale $[\underline{t}_r,\bar{t}_r]$ i założeniu, że $x_r(\underline{t}_r)=0$ otrzymujemy

$$x_r(t)=\overline{\varphi}_r(u_r;\boldsymbol{a}_r)\,t,$$

czyli dla $t = T_r$

$$x_r^* = x_r(T_r) = \overline{\varphi}_r(u_r; \boldsymbol{a}_r) \cdot T_r$$

oraz model czasowy

$$T_r = \frac{x_r^*}{\overline{\varphi}_r(u_r; \boldsymbol{a}_r)} = \gamma_r(u_r; \boldsymbol{a}_r). \tag{1.6a}$$

Niech na przykład $\dot{x}_r(t) = k_r\, u_r$, gdzie parametr $\boldsymbol{a}_r = k_r$ jest wielkością skalarną. Wówczas $x_r(t) = k_r\, u_r\, t$, $x_r^* = k_r\, u_r\, T_r$ oraz

$$T_r = \frac{x_r^*}{k_r\, u_r}. \tag{1.6b}$$

2. Przypadek analogiczny do (1.6), ale zmiana stanu zależy nie tylko od ilości zasobu, ale i od czasu, to znaczy

$$\dot{x}_r(t) = \overline{\varphi}_r(u_r, t; \boldsymbol{a}_r), \tag{1.7}$$

czyli

$$x_r(t) = \int_{t_r}^{\bar{t}_r} \overline{\varphi}_r(u_r, t; \boldsymbol{a}_r)\, dt,$$

dla $x_{r,0} = 0$. Dla otrzymania modelu postaci (1.4) musimy za t podstawić T_r, rozwiązać równanie $x_r(T_r) = x_r^*$ względem T_r i w ten sposób otrzymać zależność T_r od u_r. Rozpatrzmy następujący przykład, w którym jak poprzednio k_r jest wielkością skalarną i przyjmujemy zerowe warunki początkowe

$$\dot{x}_r(t) = k_r\, u_r\, t,$$

czyli

$$x_r(t) = k_r\, u_r \int_{t_r}^{\bar{t}_r} t\, dt = \frac{1}{2} k_r\, u_r\, t^2.$$

Stąd

$$x_r^* = \frac{1}{2} k_r\, u_r\, T_r^2$$

oraz

$$T_r = \sqrt{\frac{2\,x_r^*}{k_r\,u_r}}\,. \tag{1.7a}$$

3. Rozpatrzmy teraz szczególną postać modelu (1.2), w której prędkość zmiany stanu zależy od wartości stanu w danej chwili, to znaczy

$$\dot{x}_r(t) = \varphi_r(x_r(t), u_r; \boldsymbol{a}_r). \tag{1.8}$$

Aby otrzymać model (1.4), należy rozwiązać równanie różniczkowe (1.8) dla danego warunku początkowego $x_{r,0}$, czyli wyznaczyć $x_r(t)$. Dalsze postępowanie jest takie, jak w poprzednim przypadku. Na przykład dla

$$\dot{x}_r(t) = k_r\,u_r\,x_r(t)$$

po wykorzystaniu rozwiązania tego równania

$$x_r(t) = x_{r,0}\exp(k_r\,u_r\,t)$$

otrzymujemy

$$T_r = \frac{\ln x_r^* - \ln x_{r,0}}{k_r\,u_r}\,. \tag{1.8a}$$

4. Kolejny przykład dotyczy modelu (1.4). Ważna klasa tego typu modeli jest określona przez funkcje potęgowe postaci

$$T_r = \frac{1}{k_r\,u_r^{\alpha_r}}\,, \tag{1.9}$$

gdzie $k_r = a_r^{(1)}$ i $\alpha_r = a_r^{(2)}$.

Warto zauważyć, że wszystkie podane przykłady modeli są szczególnymi przypadkami modelu (1.9), a mianowicie w kolejnych modelach (1.6b), (1.7a) i (1.8a) potęgi α_r są stałe i równe odpowiednio 1, $\frac{1}{2}$ i 1, natomiast parametry $a_r^{(1)} - \frac{k_r}{x_r^*}$, $\sqrt{\frac{k_r}{2\,x_r^*}}$ i $\frac{k_r}{\ln x_r^* - \ln x_{r,0}}$.

Model kompleksu operacji jako obiektu złożonego musi się składać z dwóch części: z modeli elementów, czyli operacji oraz z modelu struktury opisującego strukturę, czyli powiązania między operacjami. Po prezentacji modeli operacji zostanie

teraz przedstawiony model struktury. Model taki ma opisywać, tak zwane ograniczenia kolejnościowe, występujące w zbiorze operacji $\boldsymbol{R}$. Wprowadźmy w zbiorze $\boldsymbol{R}$ relację $\boldsymbol{\rho}$ zachodzącą między dwiema operacjami q oraz r, określoną przez warunek: „wykonywanie operacji r może się rozpocząć bezpośrednio po zakończeniu wykonywania operacji q", czyli $q\ \boldsymbol{\rho}\ r$ lub $((q,r)\in\boldsymbol{\rho})$. Relacja $\boldsymbol{\rho}$ jest antysymetryczna [100]. Struktura kompleksu operacji jest określona przez podanie wszystkich uporządkowanych par (q,r) będących w relacji $\boldsymbol{\rho}$. Forma ich prezentacji może być różna. Podamy dwa przykłady. W pierwszym z nich pary $q\ \boldsymbol{\rho}\ r$ tworzą zbiór $\boldsymbol{\Theta}$ będący podzbiorem iloczynu kartezjańskiego $\boldsymbol{R}\times\boldsymbol{R}$, czyli

$$\Theta \subset \boldsymbol{R}\times\boldsymbol{R} = \{(q,r) : q\ \boldsymbol{\rho}\ r\}. \tag{1.10}$$

Innym sposobem zapisu par operacji, między którymi zachodzą ograniczenia kolejnościowe, jest macierz incydencji

$$\boldsymbol{\kappa} = [\kappa_{q,r}]_{q,r=1,2,\ldots,R}, \tag{1.11}$$

gdzie

$$\kappa_{q,r} = \begin{cases} 1, & \text{jeśli } q\ \boldsymbol{\rho}\ r, \\ 0, & \text{w przeciwnym przypadku.} \end{cases}$$

Struktura kompleksu operacji, czyli przede wszystkim ograniczenia kolejnościowe, może być wyrażona w formie graficznej, jak to już zostało wspomniane w punkcie 1.1. Wierzchołki oznaczają zdarzenia polegające na rozpoczęciu i zakończeniu wykonywania operacji. Graf jest spójny, acykliczny i ma po jednym wierzchołku początkowym i końcowym. Dla problemu rozdziału zasobów przyjmujemy konwencję „operacja na łuku". Dla kompleksu operacji z rys. 1.5 zbiór $\boldsymbol{\Theta}$ i macierz $\boldsymbol{\kappa}$ są następujące:

$$\boldsymbol{\Theta} = \{(1,3), (1,4), (2,5), (2,6), (3,5), (3,6), (4,7), (5,7)\},$$

$$\boldsymbol{\kappa} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Istotą rozpatrywanego zagadnienia jest określenie wielkości zasobów przydzielonych każdej operacji w sytuacji, gdy ilość zasobów, którą dysponujemy, jest ograniczona. Oznaczmy przez $\boldsymbol{u}(t)$ wektor wielkości zasobów przydzielonych wszystkim operacjom, czyli

$$\boldsymbol{u}(t) = [u_1(t), u_2(t), ..., u_R(t)]^{\mathrm{T}}$$

oraz przez $\boldsymbol{D}_{\boldsymbol{u}}(t)$ zbiór dopuszczalnych rozdziałów zasobów jako

$$\boldsymbol{D}_{\boldsymbol{u}}(t) = \{\boldsymbol{u}(t):\ u_r(t) \geq 0,\ r = 1,2, ..., R,\ \sum_{r=1}^{R} u_r(t) \leq U\},\ 0 \leq t \leq T\ , \quad (1.12)$$

gdzie U i T są odpowiednio globalną ilością zasobów i czasem wykonania wszystkich operacji. Warunki definiujące zbiór $\boldsymbol{D}_u(t)$ oznaczają, że w każdym momencie czasu, w którym są wykonywane operacje, suma nieujemnych wielkości zasobów przydzielanych tym operacjom nie może przekroczyć globalnej ilości zasobów U. Jeśli na zasoby dodatkowo jest nałożone ograniczenie

$$\sum_{r=1}^{R} \int_{\underline{t}_r}^{\bar{t}_r} u_r(t)\, dt \leq U_c\ , \quad (1.12a)$$

gdzie U_c jest zadaną wartością całkowitej ilości zasobu, to mówimy, że zasoby są podwójnie ograniczone. Suma występująca w (1.12) zapewnia chwilowe ograniczenie zasobu, podczas gdy U_c w (1.12a) jest górnym ograniczeniem na całkowity wydatek zasobu w całym horyzoncie czasu, w którym jest wykonywany kompleks operacji. Jeśli na przykład zasoby mają na celu zasilanie urządzeń elektrycznych biorących udział w wykonywaniu operacji, to pierwsza suma ma interpretację mocy, natomiast druga nakłada ograniczenie na całkowity wydatek energii.

Czas T wykonania kompleksu operacji, czyli wszystkich operacji jest zdefiniowany jako

$$T \stackrel{\Delta}{=} \bar{t}^{\max} - \underline{t}^{\min}\ ,$$

gdzie $\bar{t}^{\max} = \max\limits_{r=1,2, ..., R} \{\bar{t}_r\}$, $\underline{t}^{\min} = \min\limits_{r=1,2, ..., R} \{\underline{t}_r\} = 0$. Jego wartość zależy od momentów zakończenia wykonania poszczególnych operacji, czyli

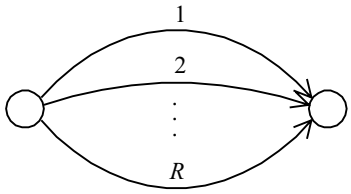
$$T = \overline{\Phi}(\bar{t}_1, \bar{t}_2, ..., \bar{t}_R)\ , \quad (1.13)$$

i wobec $\bar{t}_r = \underline{t}_r + T_r$

$$T = \Phi(T_1, T_2, ..., T_R), \tag{1.13a}$$

gdzie $\overline{\Phi}$ i Φ są odwzorowaniami zależnymi od struktury kompleksu operacji. Dla prostych struktur w łatwy sposób można podać analityczne postaci $\overline{\Phi}$ i Φ. Przykładowo, dla struktury równoległej, czyli dla kompleksu operacji niezależnych (rys. 1.6), bez straty ogólności można założyć, że $\underline{t}_r = 0,\ r = 1, 2, ..., R$ i wtedy

$$T = \max\{\bar{t}_1, \bar{t}_2, ..., \bar{t}_R\} = \max\{T_1, T_2, ..., T_R\}. \tag{1.14}$$



Rys. 1.6. Kompleks operacji o strukturze równoległej
(kompleks operacji niezależnych)

Tak więc czas wykonania całego kompleksu jest równy czasowi wykonania najdłuższej operacji.

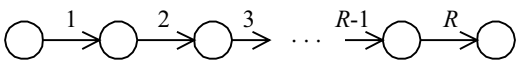
Dla struktury szeregowej (rys. 1.7) $\underline{t}_r = \bar{t}_{r-1},\ r = 2,3, ..., R$ i $\underline{t}_1 = 0$. Stąd

$$T = \bar{t}_R = \sum_{r=1}^{R} T_r, \tag{1.15}$$

czyli czas wykonania kompleksu jest sumą czasów wykonania wszystkich operacji.

Czas T jest najczęściej stosowanym kryterium rozdziału zasobów. Przypadki innych kryteriów, na przykład koszt wykonania kompleksu operacji z ograniczeniem czasu jego wykonania, zysk związany z realizacją kompleksu, różne przypadki wielokryterialne, nie będą tu rozważane.

Na podstawie dotychczasowego opisu problem rozdziału zasobów można określić następująco.



Rys. 1.7. Kompleks operacji o strukturze szeregowej

Dla kompleksu R operacji o modelach (1.2), (1.3) lub (1.4) z danymi stanami końcowymi x_r^* i o strukturze określonej w postaci zbioru (1.10) należy wyznaczyć wektor $\boldsymbol{u}(t) \in \boldsymbol{D}_u(t)$ tak, aby minimalizować czas T.

Algorytm rozwiązania tak postawionego problemu zależy od postaci modeli operacji. Dla najbardziej ogólnego przypadku (1.2) uzyskano rezultaty dla kompleksu operacji niezależnych i specyficznych postaci funkcji φ_r, na przykład dla postaci multiplikatywnej, to znaczy

$$\varphi_r[x_r(t), u_r(t); \boldsymbol{a}_r] = \varphi_r^{(1)}[x_r(t)]\varphi_r^{(2)}[u_r(t)]$$

lub dla funkcji wypukłych ze względu na u_r [89]. Problem sprowadzono do trudnego zagadnienia sterowania czasowo-optymalnego dla specyficznej struktury obiektu dynamicznego i nietypowych ograniczeń. Jak już wspomniano wcześniej, problem z modelem (1.3) rozważano w [114, 115, 5]. Rozpatrywano tam między innymi przypadek, gdy dodatkowo są wprowadzone żądania zasobowe dla operacji, to znaczy $u_r(t) \in [\underline{u}_r, \overline{u}_r]$, $0 \le \underline{u}_r \le \overline{u}_r$, gdzie $\underline{u}_r$ i $\overline{u}_r$ są znanymi stałymi, przy czym operacje są podzielne, to znaczy można przerywać ich wykonywanie. Do wyznaczenia algorytmu rozwiązania zastosowano metody analizy funkcjonalnej, a zwłaszcza tak zwanego funkcjonału Minkowskiego. W dalszym ciągu szczegółowo będzie rozpatrywany problem z modelem (1.4).

Algorytm rozwiązania

Algorytm rozwiązania będzie wyznaczany dla modeli (1.4). Najpierw problem rozdziału zasobów zostanie sformułowany jako odpowiedni problem optymalizacyjny, a następnie do jego rozwiązania zostanie zastosowana metoda optymalizacji dwupoziomowej [40, 2]. Zastosowanie tej metody optymalizacji jest związane z dekompozycją problemu. Polega ona na odpowiednim podziale horyzontu czasu, w którym jest realizowany kompleks operacji na krótsze odcinki (przedziały), wyznaczone przez zdarzenia, czyli momenty czasu, w których co najmniej jedna operacja się rozpoczyna lub kończy. Następnie decyzje o rozdziale zasobów są podejmowane niezależnie w poszczególnych przedziałach czasu oraz, w celu uwzględnienia ich wzajemnych powiązań, jest rozwiązywane zadanie koordynacji przewidziane w optymalizacji dwupoziomowej. Ważne jest, że operacje należące do jednego przedziału są traktowane jako niezależne.

Problem czasowo-optymalnego rozdziału zasobów faktycznie polega na minimalizacji względem wektora rozdziału zasobów $\boldsymbol{u}$, czasu wykonywania kompleksu operacji

$$T = F(\boldsymbol{u}) = \Phi[\bar{\gamma}_1(u_1), \bar{\gamma}_2(u_2), ..., \bar{\gamma}_R(u_R)], \tag{1.16}$$

gdzie $\bar{\gamma}_r(u_r) \triangleq \gamma_r(u_r; \boldsymbol{a}_r)$, dla ograniczeń

$$u_r \geq 0, \ r = 1, 2, ..., R, \tag{1.17}$$

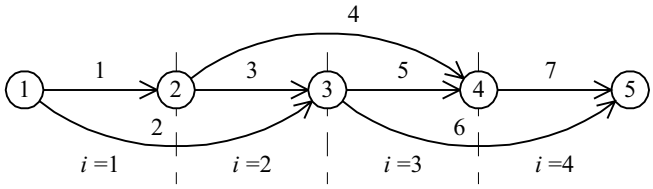
$$\sum_{r=1}^{R} u_r \leq U, \tag{1.18}$$

$$\underline{t}_r - \underline{t}_q \geq \bar{\gamma}_q(u_q), \ \text{dla} \ (q, r) \in \boldsymbol{\rho}. \tag{1.19}$$

Nierówności (1.19) są analitycznym zapisem ograniczeń kolejnościowych. Wynikiem minimalizacji jest optymalny rozdział zasobów $\boldsymbol{u}^* = [u_1^*, u_2^*, ..., u_R^*]^T$ oraz minimalny czas wykonywania kompleksu operacji $T^* \triangleq F(\boldsymbol{u}^*)$. Prezentacja właściwego algorytmu rozwiązania zostanie poprzedzona podaniem sposobu dekompozycji kompleksu operacji na przedziały i przedstawieniem algorytmu rozdziału zasobów dla kompleksu operacji niezależnych.

Dekompozycja kompleksu operacji

Oznaczmy przez *l* liczbę zdarzeń w kompleksie operacji. Jest to jednocześnie liczba wierzchołków w grafie prezentującym kompleks. Liczba operacji, a więc i łuków w grafie wynosi *R*. Sposób dekompozycji będzie prościej zaprezentować odwołując się do grafu. Wierzchołki (zdarzenia) należy uporządkować w ten sposób, aby zajście zdarzenia opisanego przez wierzchołek *i* nie było późniejsze niż zajście zdarzenia opisanego przez wierzchołek *j*, gdy $1 \leq i \leq j \leq l$. Określmy zbiór $\boldsymbol{S}_i$ wszystkich operacji, które mogą być wykonywane w przedziale czasu między zdarzeniami *i* oraz $i+1$, $i = 1, 2, ..., l-1$ oraz zbiór $\boldsymbol{Q}_r$ numerów przedziałów, w których jest wykonywana operacja *r*, $r = 1, 2, ..., R$. Indeksem *i* będziemy oznaczać również numer przedziału czasu o początku w wierzchołku *i* oraz końcu w wierzchołku *i* + 1. Dla przykładu rozważmy kompleks operacji z rys. 1.5. Można go przedstawić w następującej uporządkowanej postaci (rys. 1.8). Zbiory $\boldsymbol{S}_i$ oraz $\boldsymbol{Q}_r$ są następujące



Rys. 1.8. Kompleks operacji z rys. 1.5 po uporządkowaniu

$\boldsymbol{S}_1 = \{1,2\}$, $\boldsymbol{S}_2 = \{2,3,4\}$, $\boldsymbol{S}_3 = \{4,5,6\}$, $\boldsymbol{S}_4 = \{6,7\}$, $\boldsymbol{Q}_1 = \{1\}$, $\boldsymbol{Q}_2 = \{1,2\}$, $\boldsymbol{Q}_3 = \{2\}$, $\boldsymbol{Q}_4 = \{2,3\}$, $\boldsymbol{Q}_5 = \{3\}$, $\boldsymbol{Q}_6 = \{3,4\}$, $\boldsymbol{Q}_7 = \{4\}$. Łatwo zauważyć, że postać zbiorów $\boldsymbol{S}_i$ w sposób jednoznaczny determinuje postać zbiorów $\boldsymbol{Q}_r$ i na odwrót. Zbiory te są jeszcze jednym sposobem opisu struktury kompleksu operacji.

Rozdział zasobów dla kompleksu operacji niezależnych

Rozważmy kompleks operacji niezależnych (rys. 1.6) o modelach (1.4). Dla tego szczególnego przypadku problem czasowo-optymalnego rozdziału zasobów polega na minimalizacji kryterium (1.16) o szczegółowej postaci

$$T = \overline{F}(\boldsymbol{u}) = \max_{r=1,2,...,R} \{\bar{\gamma}_r(u_r)\}$$

z ograniczeniami (1.17) i (1.18). Prawdziwe jest twierdzenie, które podaje warunki istnienia rozwiązania czasowo-optymalnego. Są nimi: wykorzystanie całej dostępnej ilości zasobu U oraz taki sam czas trwania wszystkich operacji.

Twierdzenie 1.1

Czas wykonywania kompleksu operacji niezależnych dla problemu rozdziału zasobów jest minimalny wtedy i tylko wtedy, gdy

$$\sum_{r=1}^{R} u_r^* = U \tag{1.20}$$

oraz

$$\bar{\gamma}(u_r^*) = T^*, \ r = 1,2,...,R. \tag{1.21}$$

Dowód. Przy założeniu, że $\boldsymbol{u}^*$ jest rozwiązaniem czasowo-optymalnym, przyjmujemy, że teza nie jest prawdziwa, to znaczy, że co najmniej jeden z warunków (1.20), (1.21) nie jest spełniony. Gdy równość (1.20) nie jest prawdziwa, można znaleźć rozdział różny od $\boldsymbol{u}^*$, na przykład $\tilde{\boldsymbol{u}} = [u_1^*, u_2^*, ..., u_r^* + \Delta\tilde{u}_r, ..., u_R^*]^{\mathrm{T}}$ taki, że $\Delta\tilde{u}_r > 0$ oraz $\Delta\tilde{u}_r + \sum_{r=1}^{R} u_r^* \le U$. Ponieważ funkcja $\bar{\gamma}_r$ jest ściśle monotoniczna, więc $\bar{\gamma}_r(u_r^* + \Delta\tilde{u}_r) < \bar{\gamma}_r(u_r^*) = T^*$ i wobec (1.21) rozdział $\tilde{\boldsymbol{u}}$ jest lepszy niż $\boldsymbol{u}^*$, co przeczy przyjętemu założeniu.

Przyjmijmy teraz, że dla czasowo-optymalnego rozdziału $\boldsymbol{u}^*$ nie jest prawdziwy warunek (1.21). Wtedy istnieją dwie takie operacje q oraz r, dla których $\bar{\gamma}_q(u_q^*) < \bar{\gamma}_r(u_r^*) = T^*$. Korzystając ze ścisłej monotoniczności modelu oraz nie na-

ruszając równości (1.20), można znaleźć taką wartość $\Delta u_r > 0$, że $\bar{\gamma}_q(u_q^* - \Delta u_r) < \bar{\gamma}_r(u_r^* + \Delta u_r) \le T^*$ i rozdział $\boldsymbol{u}^*$ nie jest optymalny.

Załóżmy teraz prawdziwość (1.20) oraz (1.21) i przyjmijmy, że istnieje rozdział lepszy niż $\boldsymbol{u}^*$. Oznaczmy go jako $\bar{\boldsymbol{u}} = [u_1^* + \Delta\bar{u}_1, u_2^* + \Delta\bar{u}_2, ..., u_R^* + \Delta\bar{u}_R]^{\mathrm{T}}$. Ze względu na (1.20) $\sum_{r=1}^{R} \Delta\bar{u}_r = 0$, a więc istnieje co najmniej jedna taka operacja *r*, że $\Delta\bar{u}_r < 0$. Wówczas $\bar{\gamma}_r(u_r^* + \Delta\bar{u}_r) > \bar{\gamma}_r(u_r^*) = T^*$, co prowadzi do sprzeczności wobec przyjętego założenia

■

Warunki podane w twierdzeniu można bezpośrednio wykorzystać do wyznaczenia czasowo-optymalnego algorytmu rozdziału zasobów. Traktujemy je jako układ $R+1$ równań z $R+1$ niewiadomymi, to znaczy u_r^*, $r = 1, 2, ..., R$ oraz T^*. Po rozwiązaniu tego układu równań można uzyskać szukany algorytm. Możliwość analitycznego rozwiązania układu zależy od postaci funkcji $\bar{\gamma}_r$. Załóżmy, że istnieją funkcje odwrotne do $\bar{\gamma}_r$, a więc i do funkcji γ_r ze względu na u_r. Wtedy

$$u_r^* = \bar{\gamma}_r^{-1}(T^*) = \gamma_r^{-1}(T^*; \boldsymbol{a}_r), \quad r = 1,2, ..., R. \tag{1.22}$$

Po wstawieniu do (1.20) otrzymujemy

$$\sum_{r=1}^{R} \gamma_r^{-1}(T^*; \boldsymbol{a}_r) = U, \tag{1.23}$$

czyli równanie algebraiczne z jedną niewiadomą T^*. Gdy istnieje jego rozwiązanie analityczne, mamy wyrażenie na optymalny czas wykonywania kompleksu operacji

$$T^* = G(U, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R), \tag{1.24}$$

który po wstawieniu do zalezności (1.22) daje algorytm rozdziału zasobów

$$u_r^* = \gamma_r^{-1}[G(U, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R); \boldsymbol{a}_r] \stackrel{\Delta}{=} \eta_r(U, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R). \tag{1.25}$$

Algorytm ten zależy od globalnej ilości zasobów oraz od parametrów operacji. Algorytm opisany równaniami (1.22)–(1.25), po założeniu istnienia funkcji $\bar{\gamma}_r^{-1}$ oraz analitycznego rozwiązania (1.23), nazywamy algorytmem analitycznym.

Przykład 1.1

Przyjmijmy modele operacji w postaci (1.7a). Wtedy należy rozwiązać układ równań

$$\begin{cases} \sum_{r=1}^{R} u_r^* = U \\ \sqrt{\dfrac{2 \cdot x_r^*}{k_r \cdot u_r^*}} = T^*, \ r = 1,2, ..., R . \end{cases} \tag{1.26}$$

Po przekształceniach otrzymujemy kolejno zależności (1.22), (1.23), (1.24), (1.25), czyli

$$u_r^* = \frac{2 \cdot x_r^*}{k_r \cdot (T^*)^2}, \quad \sum_{r=1}^{R} \frac{2 \cdot x_r^*}{k_r \cdot (T^*)^2} = U ,$$

$$T^* = \sqrt{\frac{2}{U} \sum_{r=1}^{R} \frac{x_r^*}{k_r}}, \quad u_r^* = U \frac{\dfrac{x_r^*}{k_r}}{\sum_{r=1}^{R} \dfrac{x_r^*}{k_r}} .$$

■

W przypadkach, gdy nie istnieje co najmniej jedna funkcja odwrotna do $\bar{\gamma}_r$ względem u_r lub przy istnieniu wszystkich takich funkcji, nie można w sposób analityczny rozwiązać równania algebraicznego (1.23) – do wyznaczania algorytmu rozdziału zasobów należy zastosować metody numeryczne rozwiązywania układu równań algebraicznych lub równania algebraicznego. W pierwszym przypadku otrzymujemy tak zwany algorytm analityczno-numeryczny. Polega on na wyznaczeniu funkcji (1.22) oraz (1.23), numerycznym rozwiązaniu (1.23) i otrzymaniu liczbowej wartości T^*, a następnie na wstawieniu tej wartości do (1.22) i uzyskaniu liczbowych wartości rozdziału zasobów. W drugim przypadku mamy algorytm numeryczny. Liczbę równań w układzie $R+1$ równań (1.20), (1.21) można zmniejszyć o dwa, stosując podstawienia

$$u_1^* = U - \sum_{r=2}^{R} u_r^*$$

oraz

$$T^* = \bar{\gamma}_1\left(U - \sum_{r=2}^{R} u_r^*\right).$$

W rezultacie otrzymujemy układ $R - 1$ równań z niewiadomymi $u_2^*, u_3^*, ..., u_R^*$, który należy rozwiązać numerycznie.

Rozważmy prosty przykład ilustrujący zastosowanie problemu rozdziału zasobów dla kompleksu operacji niezależnych, w oddziale konwertorów tlenowych huty żelaza.

Przykład 1.2

Proces produkcji stali prowadzony w R niezależnych urządzeniach, nazywanych konwertorami tlenowymi, polega na poddawaniu procesowi utleniania w wysokiej temperaturze mieszaniny ciekłej surówki i złomu stalowego w celu usunięcia domieszek, głównie węgla. Każda operacja jest wykonywana w oddzielnym konwertorze. Zadanie, które ma być zrealizowane w ramach wykonywania operacji, polega na zmniejszeniu, we wsadzie znajdującym się w konwertorze, procentowej zawartości domieszek od wartości $x_{r,0}$ do wartości x_r^*. Jedynym rozważnym zasobem jest tlen dostarczany do urządzeń z jednego generatora o stałej wydajności U. W rzeczywistym procesie są jeszcze inne zasoby, na przykład topniki. Problem polega na takim rozdziale zasobu o wielkości U między urządzenia, aby czas wykonania wszystkich operacji był najkrótszy. Można przyjąć następujący uproszczony model operacji o postaci (1.6)

$$\dot{x}_r(t) = -k_r u_r ,$$

gdzie $x_r(t)$ jest procentową zawartością domieszek we wsadzie znajdującym się w r-tym konwertorze, przy czym $x_r^* \le x_r(t) \le x_{r,0}$ oraz k_r jest stałym nieujemnym parametrem charakteryzującym wydajność konwertora, stosowanego do realizacji operacji r. Ponieważ warunki początkowe nie są zerowe, więc po scałkowaniu w granicach $\underline{t}_r \neq 0$ do $\bar{t}_r$ otrzymamy rozszerzoną wersję modelu (1.6b), a mianowicie

$$T_r = \frac{x_{r,0} - x_r^*}{k_r\, u_r} .$$

Szukany algorytm rozdziału zasobów jest wynikiem rozwiązania układu równań

$$\begin{cases} \sum_{r=1}^{R} u_r^* = U \\ \dfrac{x_{r,0} - x_r^*}{k_r \cdot u_r^*} = T^*, \quad r = 1,2, ..., R \end{cases}$$

i ma postać

$$u_r^* = U \frac{\dfrac{x_{r,0} - x_r^*}{k_r}}{\sum_{r=1}^{R} \dfrac{x_{r,0} - x_r^*}{k_r}}.$$

Optymalny czas wykonania kompleksu operacji wynosi

$$T^* = \frac{1}{U} \sum_{r=1}^{R} \frac{x_{r,0} - x_r^*}{k_r}.$$

■

Powróćmy teraz do wyznaczenia algorytmu rozwiązania problemu rozdziału zasobów dla kompleksu operacji o dowolnej strukturze, nazywanego też kompleksem operacji zależnych.

Rozdział zasobów dla kompleksu operacji zależnych

Kompleks operacji został zdekomponowany na przedziały. Poszczególne operacje mogą być wykonywane w więcej niż jednym przedziale. Ponieważ decyzje są podejmowane w każdym przedziale osobno, w stosunku do jednej operacji może być podejmowanych kilka decyzji. Stosowaną dotychczas notację należy dostosować do tej sytuacji. Niech $u_{r,i}$ oraz $x_{r,i}$ oznaczają odpowiednio zasoby przydzielone operacji r w przedziale i oraz rozmiar operacji r wykonywanej w przedziale i, czyli część rozmiaru całkowitego operacji r. Do dalszych rozważań przyjmijmy wersję (1.6a) modelu (1.4), która dla i-tego przedziału ma postać

$$T^i = \frac{x_{r,i}}{\overline{\varphi}_r(u_{r,i}; \boldsymbol{a}_r)} \stackrel{\Delta}{=} \tilde{\gamma}_r(u_{r,i}, x_{r,i}), \tag{1.27}$$

gdzie T^i jest czasem trwania przedziału i.

Równanie to oznacza, że po przydziale zasobów o wielkości $u_{r,i}$ w przedziale i jest zrealizowana część operacji o wielkości $x_{r,i}$. Czas trwania kompleksu jest sumą długości wszystkich przedziałów. Zmiennymi optymalizacyjnymi są teraz zasoby $u_{r,i}$ oraz rozmiary częściowe $x_{r,i}$, które tworzą zbiory

$$\boldsymbol{U}_i \stackrel{\Delta}{=} \{u_{r,i} : r \in \boldsymbol{S}_i\},\ i = 1, 2, ..., l-1, \tag{1.28}$$

$$\boldsymbol{X}_i \stackrel{\Delta}{=} \{x_{r,i} : r \in \boldsymbol{S}_i\},\ i = 1, 2, ..., l-1. \tag{1.29}$$

Wtedy kryterium (1.16) przyjmuje postać

$$T = \sum_{i=1}^{l-1} T^i = \sum_{i=1}^{l-1} F_i(\boldsymbol{U}_i, \boldsymbol{X}_i) = \sum_{i=1}^{l-1} \max_{r \in \boldsymbol{S}_i} \{\tilde{\gamma}_r(u_{r,i}, x_{r,i})\}, \tag{1.30}$$

a na elementy zbiorów $\boldsymbol{U}_i$ oraz $\boldsymbol{X}_i$ obowiązują ograniczenia

$$u_{r,i} \geq 0,\ r \in \boldsymbol{S}_i,\ i = 1, 2, ..., l-1, \tag{1.31}$$

$$\sum_{r \in \boldsymbol{S}_i} u_{r,i} \leq U,\ i = 1, 2, ..., l-1, \tag{1.32}$$

$$x_{r,i} \geq 0,\ r \in \boldsymbol{S}_i,\ i = 1, 2, ..., l-1, \tag{1.33}$$

$$\sum_{i \in \boldsymbol{Q}_r} x_{r,i} = x_r^*,\ r = 1, 2, ..., R. \tag{1.34}$$

Twierdzenie 1.2

Minimalizacja kryterium (1.16) z ograniczeniami (1.17)–(1.19) jest równoważna minimalizacji kryterium (1.30) z ograniczeniami (1.31)–(1.34).

Dowód. Równoważność kryteriów (1.16) i (1.30) wynika z zastosowanej dekompozycji kompleksu operacji, która zachowuje ograniczenia kolejnościowe, ponieważ nie ulegają zmianie zdarzenia, między którymi poszczególne operacje są wykonywane. Kryterium (1.30) jest formalnym zapisem ogólnego odwzorowania Φ występującego w (1.16). Równoważność ograniczeń (1.17) i (1.18) oraz (1.31) i (1.32) jest jasna. Obie pary ograniczeń zapewniają, że wielkości zasobów są nieujemne oraz że nie zostanie wykorzystane więcej zasobów niż jest dostępne. Warunek (1.34) osiągnięcia rozmiaru całkowitego operacji w pierwszym problemie optymalizacyjnym jest zapewniony w modelu operacji (1.6a). Występująca tam wartość x_r^* jest nieujemna.

■

Postać funkcji celu (1.30) oraz ograniczeń (1.31)–(1.34) umożliwia dekompozycję problemu minimalizacji i w konsekwencji zastosowanie metody optymalizacji dwupoziomowej (np. [40, 2]) do wyznaczenia optymalnego rozdziału zasobów [16, 17].

Na poziomie dolnym (rys. 1.9) jest rozwiązywanych $l-1$ niezależnych problemów optymalizacyjnych, polegających na minimalizacji kryterium

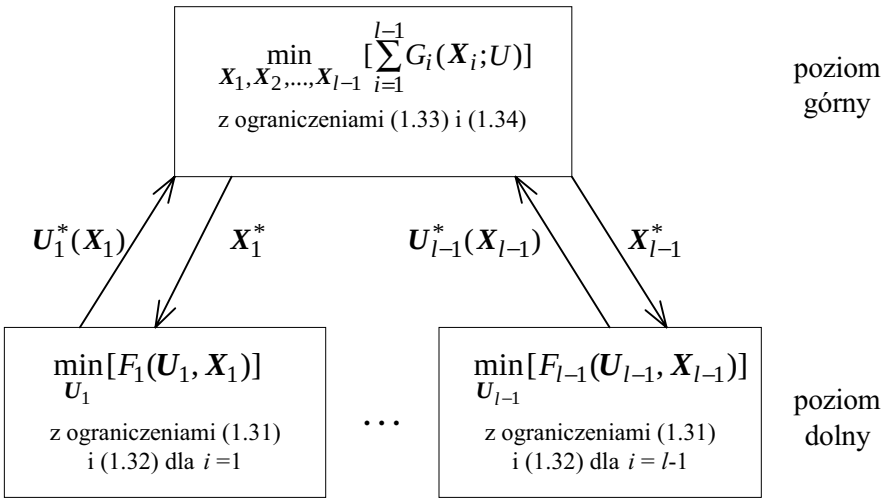
$$T^i = F_i(\boldsymbol{U}_i, \boldsymbol{X}_i) = \max_{r \in \boldsymbol{S}_i} \{\tilde{\gamma}_r(u_{r,i}, x_{r,i})\} \tag{1.35}$$

względem $\boldsymbol{U}_i$, $i = 1, 2, ..., l-1$ dla ustalonych $\boldsymbol{X}_i$ oraz z ograniczeniami (1.31) oraz (1.32). Problem ten, poruszony wcześniej w zagadnieniu rozdziału zasobów dla kompleksu operacji niezależnych, sprowadza się do rozwiązania układu równań

$$\begin{cases} \sum\limits_{r \in \boldsymbol{S}_i} u^*_{r,i} = U \\ \tilde{\gamma}_r(u^*_{r,i}, x_{r,i}) = T^{i,*}, \ r \in \boldsymbol{S}_i \ . \end{cases}$$

Stosując znane już przekształcenia, otrzymujemy

$$u^*_{r,i}(\boldsymbol{X}_i) = \eta_{r,i}(\boldsymbol{X}_i; U), \ r \in \boldsymbol{S}_i \tag{1.36}$$



Rys. 1.9. Schemat algorytmu rozdziału zasobów opartego na metodzie optymalizacji dwupoziomowej

oraz

$$T^{i,*}(\boldsymbol{X}_i) = G_i(\boldsymbol{X}_i; U)\,. \tag{1.37}$$

Rozwiązania (1.36) tworzą zbiór $\boldsymbol{U}_i^*(\boldsymbol{X}_i)$. Zmienne wchodzące w skład zbiorów $\boldsymbol{X}_i$, $i = 1,2,...,l-1$ są tak zwanymi zmiennymi koordynacyjnymi, zapewniającymi powiązanie między rozwiązywanymi niezależnie w poszczególnych przedziałach podproblemami minimalizacji ze zmiennymi ze zbiorów $\boldsymbol{U}_i$ – które faktycznie nie są niezależne. Zmienne koordynacyjne są wyznaczane na poziomie górnym jako wynik minimalizacji kryterium

$$T = \sum_{i=1}^{l-1} G_i(\boldsymbol{X}_i; U) \tag{1.38}$$

z ograniczeniami (1.33), (1.34). Rezultatem tej minimalizacji są rozmiary operacji $x_{r,i}^*$ tworzące zbiory $\boldsymbol{X}_i^*$ oraz optymalny czas wykonania kompleksu operacji T^*. Po podstawieniu $\boldsymbol{X}_i^*$ do (1.36) otrzymujemy optymalne rozdziały zasobów w przedziałach, czyli $u_{r,i}^*$. Możliwość uzyskania rozwiązań $x_{r,i}^*$ i w konsekwencji $\boldsymbol{X}_i^*$ w formie analitycznej zależy od postaci funkcji G_i, a pośrednio od modeli operacji. Do wyznaczenia warunków koniecznych istnienia rozwiązania optymalnego można zastosować metodę Kuhna–Tuckera.

Warto zauważyć, że zmiennymi optymalizacyjnymi są tylko te wielkości $x_{r,i}$, które odpowiadają operacjom wykonywanym w więcej niż jednym przedziale. Im więcej jest takich operacji, tym większy jest rozmiar problemu optymalizacyjnego.

Następna uwaga dotyczy przypadku braku możliwości uzyskania rozwiązań analitycznych. Po pierwsze sytuacja taka może dotyczyć któregokolwiek z podproblemów z poziomu dolnego. Wtedy należy stosować dla niego metody numeryczne w sposób opisany dla kompleksu operacji niezależnych. Wówczas nie jest możliwe analityczne rozwiązanie problemu z poziomu górnego i należy zastosować metodę optymalizacji dwupoziomowej w wersji iteracyjnej [40, 2]. Druga z możliwości polega na braku rozwiązania analitycznego na poziomie górnym dla danych rozwiązań analitycznych wszystkich podproblemów z poziomu dolnego. Wtedy należy w sposób numeryczny rozwiązać problem z poziomu górnego i uzyskane liczbowe wartości ze zbioru $\boldsymbol{X}_i^*$ podstawić do analitycznych zależności określających elementy zbiorów $\boldsymbol{U}_i^*(\boldsymbol{X}_i)$ i uzyskać liczbowe ich wartości.

Okazuje się, że problem z poziomu górnego ma ciekawe własności dla szczególnych postaci modeli operacji.

Przykład 1.3

Modele liniowe. Dla modeli liniowych, to znaczy takich, w których prędkość realizacji operacji jest wprost proporcjonalna do ilości przydzielonych zasobów, to znaczy $\dot{x}_r(t) = k_r\, u_r$, na poziomie dolnym uzyskujemy następujące rozwiązanie (porównaj przykład 1.2 z zerowymi warunkami początkowymi)

$$u_{r,i}^{*}(\boldsymbol{X}_i) = U\,\frac{\dfrac{x_{r,i}}{k_r}}{\displaystyle\sum_{r\in\boldsymbol{S}_i}\frac{x_{r,i}}{k_r}}$$

oraz

$$G_i(\boldsymbol{X}_i) = \frac{1}{U}\sum_{r\in\boldsymbol{S}_i}\frac{x_{r,i}}{k_r}\,.$$

Po wstawieniu G_i do (1.38) i wykorzystaniu (1.34) otrzymujemy

$$T = \frac{1}{U}\sum_{i=1}^{l-1}\sum_{r\in\boldsymbol{S}_i}\frac{x_{r,i}}{k_r} = \frac{1}{U}\sum_{r=1}^{R}\sum_{i\in\boldsymbol{Q}_r}\frac{x_{r,i}}{k_r} = \frac{1}{U}\sum_{r=1}^{R}\frac{1}{k_r}\sum_{i\in\boldsymbol{Q}_i}x_{r,i} = \frac{1}{U}\sum_{r=1}^{R}\frac{x_r^{*}}{k_r}\,,$$

czyli czas wykonywania kompleksu operacji jest stały i nie zależy od rozmiarów częściowych $x_{r,i}$ tylko od danych rozmiarów całkowitych x_r^{*}. Nie jest więc potrzebne rozwiązywanie problemu optymalizacyjnego na poziomie górnym.

■

Na podstawie otrzymanego wyniku można jeszcze sformułować dwie uwagi. Uzyskana własność jest prawdziwa dla kompleksu operacji o dowolnej strukturze. Rozwiązanie jest niejednoznaczne, co oznacza, że dla operacji wykonywanych w więcej niż jednym przedziale, o ile jest spełniony warunek (1.34), w sposób dowolny można przyjmować wartości rozmiarów częściowych $x_{r,i}$. Ta ostatnia własność wpływa na odpowiednie kształtowanie $u_{r,i}(\boldsymbol{X}_i)$. Dla dodatkowego wymagania, aby $u_{r,i}^{*} = \text{const}$ dla $r \in \boldsymbol{S}_i$, opracowano rekurencyjny algorytm wyznaczania $x_{r,i}$ [95]. Spełnienie tego wymagania znacznie ułatwia przydział zasobów do operacji, ponieważ nie jest wymagane przenoszenie zasobów między operacjami w trakcie wykonywania kompleksu. Wystarcza wyznaczenie i nastawienie odpowiedniego rozdziału na początku jego realizacji.

Inny ciekawy wynik uzyskano dla tak zwanych modeli potęgowych.

Przykład 1.4

Modele potęgowe. Rozważmy kompleks operacji o modelach $\dot{x}_r(t) = k_r u_r^{\alpha}$, czyli $T_r = \dfrac{x_r^*}{k_r\, u_r^{\alpha}}$ dla $\alpha > 0$ i $\alpha \neq 1$. Na poziomie dolnym uzyskujemy rozwiązanie

$$u_{r,i}^*(\boldsymbol{X}_i) = U \frac{\left(\dfrac{x_{r,i}}{k_r}\right)^{\frac{1}{\alpha}}}{\displaystyle\sum_{r\in \boldsymbol{S}_i}\left(\frac{x_{r,i}}{k_r}\right)^{\frac{1}{\alpha}}} \tag{1.39}$$

oraz

$$G_i(\boldsymbol{X}_i) = \frac{1}{U^{\alpha}}\left(\sum_{r\in \boldsymbol{S}_i}\left(\frac{x_{r,i}}{k_r}\right)^{\frac{1}{\alpha}}\right)^{\alpha}. \tag{1.40}$$

Umożliwia to sformułowanie kryterium optymalizacji na poziomie górnym

$$T = \sum_{i=1}^{l-1} G_i(\boldsymbol{X}_i) = \frac{1}{U^{\alpha}} \sum_{i=1}^{l-1}\left(\sum_{r\in \boldsymbol{S}_i}\left(\frac{x_{r,i}}{k_r}\right)^{\frac{1}{\alpha}}\right)^{\alpha}. \tag{1.41}$$

Wewnętrzna funkcja wykładnicza jest określona tylko dla nieujemnych argumentów, dlatego ograniczenie (1.33) jest nieaktywne i wystarczy minimalizować (1.41) z ograniczeniami równościowymi (1.34). Stosując metodę Lagrange'a, otrzymujemy funkcję Lagrange'a

$$L(\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_{l-1}, \boldsymbol{\lambda}) = \frac{1}{U^{\alpha}} \sum_{i=1}^{l-1}\left(\sum_{r\in \boldsymbol{S}_i}\left(\frac{x_{r,i}}{k_r}\right)^{\frac{1}{\alpha}}\right)^{\alpha} + \sum_{r=1}^{R}\lambda_r\left(x_r^* - \sum_{i\in \boldsymbol{Q}_r} x_{r,i}\right),$$

gdzie $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, ..., \lambda_R]^{\mathrm{T}}$ jest wektorem mnożników Lagrange'a. Licząc pochodne funkcji Lagrange'a względem $x_{r,i}$ i przyrównując je do zera, otrzymujemy

$$\lambda_r = \frac{1}{U^{\alpha}} \left(\sum_{r \in \boldsymbol{S}_i} \left(\frac{x_{r,i}}{k_r} \right)^{\frac{1}{\alpha}} \right)^{\alpha-1} \left(\frac{x_{r,i}}{k_r} \right)^{\frac{1}{\alpha}-1} \frac{1}{k_r}$$

a po wykorzystaniu (1.40) oraz modelu, po prostych przekształceniach można wyznaczyć zależność na $u_{r,i}$ jako

$$u_{r,i} = \left(\lambda_r \, U \, k_r\right)^{\frac{1}{1-\alpha}}.$$

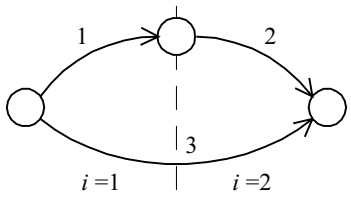
Tak więc przydział zasobów do każdej operacji jest jednakowy we wszystkich przedziałach, w których ta operacja jest wykonywana i w konsekwencji własność postulowana w przykładzie 1.3 dla modeli liniowych, w przypadku modeli potęgowych jest zawsze prawdziwa niezależnie od struktury kompleksu operacji.

■

Rozwiązanie problemu optymalizacji na poziomie górnym uzyskujemy, stosując metodę mnożników Lagrange'a dla funkcji (1.41). W celu uniknięcia skomplikowanych przekształceń dla przypadku ogólnego, sposób uzyskiwania rozwiązania przedstawimy dla elementarnego przykładu kompleksu operacji zależnych.

Przykład 1.5

Niech będzie dany kompleks trzech operacji zależnych o modelach $\dot{x}_r = \sqrt{u_r}$, czyli $T_r = x_r^* / \sqrt{u_r}$ i o strukturze przedstawionej na rys. 1.10. Ponieważ $\boldsymbol{S}_1 = \{1,3\}$, $\boldsymbol{S}_2 = \{2,3\}$ oraz zbiory $\boldsymbol{Q}_1$ i $\boldsymbol{Q}_2$ są jednoelementowe, więc $u_{1,1} = u_1^*$, $x_{1,1} = x_1^*$, $u_{2,2} = u_2^*$, $x_{2,2} = x_2^*$ oraz $\boldsymbol{X}_1 = \{x_{3,1}\}$ i $\boldsymbol{X}_2 = \{x_{3,2}\}$. Wzory (1.39) i (1.40) dla pierwszego i drugiego przedziału przyjmują teraz postać:



Rys. 1.10. Kompleks operacji w przykładzie 1.5

– dla $i = 1$

$$u_1^*(x_{3,1}) = U\frac{(x_1^*)^2}{(x_1^*)^2 + (x_{3,1})^2}, \qquad u_{3,1}^*(x_{3,1}) = U\frac{(x_{3,1})^2}{(x_1^*)^2 + (x_{3,1})^2},$$

$$G_1(x_{3,1}) = \frac{1}{\sqrt{U}}\sqrt{(x_1^*)^2 + (x_{3,1})^2}$$

– dla $i = 2$

$$u_2^*(x_{3,2}) = U\frac{(x_2^*)^2}{(x_2^*)^2 + (x_{3,2})^2}, \qquad u_{3,2}^*(x_{3,2}) = U\frac{(x_{3,2})^2}{(x_2^*)^2 + (x_{3,2})^2},$$

$$G_2(x_{3,2}) = \frac{1}{\sqrt{U}}\sqrt{(x_2^*)^2 + (x_{3,2})^2}\ .$$

Funkcja Lagrange'a jest następująca:

$$L(x_{3,1}, x_{3,2}, \lambda)$$
$$= \frac{1}{\sqrt{U}}\left(\sqrt{(x_1^*)^2 + (x_{3,1})^2} + \sqrt{(x_2^*)^2 + (x_{3,2})^2}\right) + \lambda(x_3^* - x_{3,1} - x_{3,2}).$$

Przyrównując pochodne funkcji Lagrange'a po $x_{3,1}$, $x_{3,2}$ i λ do zera otrzymujemy układ równań

$$\begin{cases} \dfrac{1}{\sqrt{U}}\dfrac{x_{3,1}}{\sqrt{(x_1^*)^2 + (x_{3,1})^2}} - \lambda = 0 \\ \dfrac{1}{\sqrt{U}}\dfrac{x_{3,2}}{\sqrt{(x_2^*)^2 + (x_{3,2})^2}} - \lambda = 0, \\ x_3^* - x_{3,1} - x_{3,2} = 0 \end{cases}$$

skąd obliczamy

$$x_{3,1}^* = \frac{x_1^* x_3^*}{x_1^* + x_2^*}, \qquad x_{3,2}^* = \frac{x_2^* x_3^*}{x_1^* + x_2^*},$$

a następnie

$$u_1^* = u_2^* = U\frac{(x_1^* + x_2^*)^2}{(x_1^* + x_2^*)^2 + (x_3^*)^2}, \qquad u_{3,1}^* = u_{3,2}^* = U\frac{(x_3^*)^2}{(x_1^* + x_2^*)^2 + (x_3^*)^2}.$$

Optymalny czas wykonania kompleksu operacji wynosi

$$T^* = G_1(x_{3,1}^*) + G_2(x_{3,2}^*)$$
$$= \frac{1}{\sqrt{U}}\left(\frac{\sqrt{(x_1^*)^2(x_1^* + x_2^*)^2 + (x_1^* x_3^*)^2} + \sqrt{(x_2^*)^2(x_1^* + x_2^*)^2 + (x_2^* x_3^*)^2}}{x_1^* + x_2^*}\right).$$

■

Należy podkreślić, że dla minimalizacji na poziomie górnym istotne są tylko operacje, które są wykonywane w co najmniej dwóch przedziałach. Oznaczmy przez $\boldsymbol{S}^{\mathrm{D}}$ zbiór takich operacji, to znaczy $\boldsymbol{S}^{\mathrm{D}} = \{r \in \boldsymbol{R} : |\boldsymbol{Q}_r| > 1\}$, gdzie $|\boldsymbol{Q}_r|$ jest licznością zbioru $\boldsymbol{Q}_r$ oraz przez $\boldsymbol{X}$ zbiór wszystkich rozmiarów częściowych tych operacji, czyli $\boldsymbol{X} = \{x_{r,i} : r \in \boldsymbol{S}^{\mathrm{D}},\ i \in \boldsymbol{Q}_r\}$.

Wtedy, zgodnie z (1.38), oznaczając czas wykonania kompleksu jako $T = \overline{G}(\boldsymbol{X})$, możemy utworzyć funkcję Lagrange'a

$$L(\boldsymbol{X}, \boldsymbol{\lambda}) = \overline{G}(\boldsymbol{X}) + \sum_{r \in \boldsymbol{S}^{\mathrm{D}}} \lambda_r \left(x_r^* - \sum_{i \in \boldsymbol{Q}_r} x_{r,i} \right).$$

Punkt stacjonarny możemy obliczyć, przyrównując do zera pochodne

$$\begin{cases} \dfrac{\partial L(\boldsymbol{X}, \boldsymbol{\lambda})}{\partial x_{r,i}} = 0, \ r \in \boldsymbol{S}^{\mathrm{D}}, \ i \in \boldsymbol{Q}_r \\ \dfrac{\partial L(\boldsymbol{X}, \boldsymbol{\lambda})}{\partial \lambda_r} = 0, \ r \in \boldsymbol{S}^{\mathrm{D}} \end{cases}.$$

Dalsze postępowanie zależy od własności funkcji $\overline{G}(\boldsymbol{X})$. Przedstawiony tu analityczny sposób rozwiązywania problemu optymalizacyjnego z poziomu górnego, oczywiście może być stosowany tylko wtedy, gdy istnieje analityczna postać funkcji $\overline{G}$. W przeciwnym razie należy, jak już wcześniej wspomniano, stosować metody numeryczne.

1.2.2. Rozdział zadań w kompleksie operacji

Problem rozdziału zadań w kompleksie operacji zostanie tu przedstawiony dla podstawowego przypadku struktury równoległej. W tym zakresie jest on analogiczny do omówionych zagadnień rozdziału zasobów. Dlatego prezentowany opis będzie nawiązywał do treści punktu 1.2.1, a niektóre wspólne fragmenty będą pominięte.

Model kompleksu operacji

W rozważaniach przyjmujemy naturalne założenie o stałości przydziału zadań do poszczególnych operacji w trakcie ich wykonywania. Jest to pierwsza różnica w stosunku do problemu rozdziału zasobów, w którym ogólnie zakładaliśmy, że u_r mogły być zmienne w czasie. Dlatego będziemy rozważać tylko model czasowy typu (1.4). Ponadto, aby nie komplikować zapisu, przyjmujemy to samo oznaczenie dla modelu. Zasada ta będzie obowiązywać też w dalszych rozważaniach, jeżeli nie będzie to odwołane. Model operacji ma więc postać

$$T_r = \gamma_r(v_r; \boldsymbol{a}_r), \tag{1.42}$$

gdzie v_r i $\boldsymbol{a}_r$ to odpowiednio zadanie przydzielone do operacji r i wektor parametrów tej operacji. Funkcje γ_r są ciągłe, ściśle rosnące ze względu na v_r dla każdego $\boldsymbol{a}_r$ oraz przyjmują wartości nieujemne w zbiorze liczb rzeczywistych, przy czym $\gamma_r(0; \boldsymbol{a}_r) = 0$. Przyjmujemy założenie, że do operacji została przydzielona niezbędna ilość zasobów w celu jej przeprowadzenia. Przydzielone zadania mają często interpretację surowców, które w ramach wykonywania operacji należy przerobić, w celu uzyskania produktów. Częstą formą modeli (1.42) jest model potęgowy

$$T_r = k_r v_r^{\alpha}, \tag{1.43}$$

gdzie $\boldsymbol{a}_r = [a_r^{(1)}, a_r^{(2)}]^{\mathrm{T}} = [k_r, \alpha]^{\mathrm{T}}$ oraz $k_r, \alpha > 0$. Inne modele to na przykład model potęgowy o zmiennych wykładnikach

$$T_r = k_r v_r^{\alpha_r}, \tag{1.43a}$$

model wykładniczy

$$T_r = k_r^{v_r} - 1, \; k_r > 1,$$

model logarytmiczny

$$T_r = \ln(k_r v_r + 1), \; k_r > 0.$$

Model struktury kompleksu jest taki sam jak dla problemu rozdziału zasobów. Zbiór dopuszczalnych rozdziałów zadań określamy jako

$$\boldsymbol{D}_v = \{\boldsymbol{v} : v_r \geq 0,\ r = 1, 2, ..., R,\ \sum_{r=1}^{R} v_r = V\}, \tag{1.44}$$

gdzie $\boldsymbol{v} = [v_1, v_2, ..., v_r]^{\mathrm{T}}$, a V jest globalną ilością zadań. Warunki definiujące ten zbiór zapewniają odpowiednio, że wielkość przydzielonego zadania jest nieujemna i wszystkie zadania mają być rozdzielone między operacje. Czas wykonania kompleksu operacji T jest funkcją czasów wykonania operacji, czyli

$$T = \Phi(T_1, T_2, ..., T_R) = \Phi[\bar{\gamma}_1(v_1), \bar{\gamma}_2(v_2), ..., \bar{\gamma}_R(v_R)] = F(\mathrm{v}), \tag{1.45}$$

gdzie Φ jest odwzorowaniem zależnym od struktury kompleksu operacji oraz $\bar{\gamma}_r(v_r) \stackrel{\Delta}{=} \gamma_r(v_r; \boldsymbol{a}_r)$. Dla kompleksu operacji niezależnych mamy

$$T = \max\{T_1, T_2, ..., T_R\} = \max\{\bar{\gamma}_1(v_1), \bar{\gamma}_2(v_2), ..., \bar{\gamma}_R(v_R)\} = \overline{F}(\mathrm{v}). \tag{1.45a}$$

Problem rozdziału zadań w kompleksie operacji stawiamy następująco:

Dla kompleksu R operacji o modelach (1.42) i o strukturze określonej w postaci zbioru (1.10) należy wyznaczyć wektor $\boldsymbol{v} \in \boldsymbol{D}_v$ tak, aby minimalizować czas T.

Algorytm rozwiązania dla kompleksu operacji niezależnych

Algorytm rozwiązania w tym przypadku jest oparty na podobnych przesłankach jak w rozdziale zasobów i polega na minimalizacji (1.45a) z ograniczeniami definiującymi zbiór $\boldsymbol{D}_v$, czyli

$$v_r \geq 0,\ r = 1, 2, ..., R, \tag{1.46}$$

$$\sum_{r=1}^{R} v_r = V. \tag{1.47}$$

W wyniku minimalizacji otrzymujemy $\boldsymbol{v}^* = [v_1^*, v_2^*, ..., v_R^*]^{\mathrm{T}}$ oraz minimalny czas $T^* = \overline{F}(\boldsymbol{v}^*)$. Warunek rozwiązania optymalnego jest określony w twierdzeniu.

Twierdzenie 1.3

Czas wykonywania kompleksu operacji niezależnych dla problemu rozdziału zadań jest minimalny wtedy i tylko wtedy, gdy

$$\bar{\gamma}(v_r^*) = T^*,\ r = 1, 2, ..., R. \tag{1.48}$$

Dowód. Dowód jest podobny jak w twierdzeniu 1.1 w części dotyczącej warunku (1.21). Przyjęcie założenia, że warunek (1.48) nie jest spełniony oznacza, że istnieje taka operacja, dla której czas wykonania jest mniejszy niż T^*, co przeczy optymalności rozdziału $\boldsymbol{v}^*$. Jeśli przyjmiemy z kolei, że istnieje inny lepszy rozdział niż $\boldsymbol{v}^*$ i oznaczymy go przez $\bar{\boldsymbol{v}} = [v_1^* + \Delta v_1, v_2^* + \Delta v_2, ..., v_R^* + \Delta v_R]^{\mathrm{T}}$, to na mocy (1.47) musi istnieć co najmniej jedna operacja r, dla której $\Delta v_r > 0$, a więc $\bar{\gamma}_r(v_r^* + \Delta v_r) > \bar{\gamma}_r(v_r^*) = T^*$. Przeczy to przyjętemu założeniu i kończy dowód. ■

Optymalny rozdział zadań oraz minimalny czas wykonywania kompleksu operacji otrzymujemy jako rozwiązanie układu równań utworzonych przez (1.47) dla $v_r = v_r^*$ oraz (1.48). Wyróżniamy trzy przypadki algorytmów w zależności od możliwości dokonania odpowiednich przekształceń algebraicznych układu równań. W algorytmie analitycznym zakładamy wpierw, że istnieją funkcje odwrotne do $\bar{\gamma}_r$ lub γ_r względem v_r. Wtedy z (1.48) uzyskujemy

$$v_r^* = \gamma_r^{-1}(T^*; \boldsymbol{a}_r) = \bar{\gamma}_r(T^*), \ r = 1, 2, ..., R, \tag{1.49}$$

a po wstawieniu wyniku do (1.47)

$$\sum_{r=1}^{R} \gamma_r^{-1}(T^*; \boldsymbol{a}_r) = V. \tag{1.50}$$

Drugim wymaganiem warunkującym możliwość stosowania algorytmu analitycznego jest istnienie analitycznego rozwiązania równania (1.50). Zakładając, że tak jest, otrzymujemy

$$T^* = G(V, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R) \tag{1.51}$$

oraz

$$v_r^* = \gamma_r^{-1}[G(V, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R), \boldsymbol{a}_r] \stackrel{\Delta}{=} \eta_r(V, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R), \tag{1.52}$$

gdzie η_r jest algorytmem rozdziału zadań.

Przykład 1.6

Rozważmy kompleks operacji o modelach (1.43) i wyznaczmy wzór określający czasowo-optymalny rozdział zadań o wielkości V według algorytmu analitycznego. Zależność (1.49) przyjmuje postać

$$v_r^* = \left(\frac{T^*}{k_r} \right)^{\frac{1}{\alpha}}. \tag{1.53}$$

Równanie (1.50) jest zatem następujące

$$\sum_{r=1}^{R} \left(\frac{T^*}{k_r} \right)^{\frac{1}{\alpha}} = V.$$

Stąd

$$T^* = \frac{V^{\alpha}}{\left(\sum_{r=1}^{R} \left(\frac{1}{k_r} \right)^{\frac{1}{\alpha}} \right)^{\alpha}}, \tag{1.54}$$

a po wstawieniu do (1.53)

$$v_r^* = V \frac{\left(\frac{1}{k_r} \right)^{\frac{1}{\alpha}}}{\sum_{r=1}^{R} \left(\frac{1}{k_r} \right)^{\frac{1}{\alpha}}}. \tag{1.55}$$

■

Algorytm analityczno-numeryczny stosujemy wówczas, gdy nie istnieje analityczne rozwiązanie równania (1.50) ze względu na T^*. Należy wtedy stosować przybliżone metody numeryczne. Są to metody iteracyjne, za pomocą których można obliczyć kolejne przybliżenia rozwiązania dla konkretnych wartości liczbowych parametrów. Typowy algorytm kolejnych przybliżeń jest następujący [22, 14]

$$\hat{T}(n+1) = \hat{T}(n) + \mu(n) \left[\sum_{r=1}^{R} \gamma_r^{-1}(\hat{T}(n); \boldsymbol{a}_r) - V \right], \; n = 0, 1, 2, ..., \tag{1.56}$$

gdzie $\hat{T}(n)$ oznacza *n*-te przybliżenie T^*. Początkowe przybliżenie $\hat{T}(0)$ jest wybierane w sposób arbitralny. Współczynniki $\mu(n)$ należy przyjmować tak, aby algorytm był zbieżny, to znaczy

$$\lim_{n \to +\infty} \hat{T}(n) = T^* .$$

Warunek zakończenia algorytmu zwykle przyjmowany jest jako

$$\left| \sum_{r=1}^{R} \gamma_r^{-1}(\hat{T}(n); \boldsymbol{a}_r) - V \right| < \varepsilon ,$$

gdzie $\varepsilon > 0$ jest wymaganą dokładnością rozwiązania. Po uzyskaniu wartości $\hat{T}(n)$ wstawiamy ją do wzoru (1.49) w miejsce T^* i obliczamy liczbowe wartości zadań przydzielonych do każdej operacji.

A l g o r y t m n u m e r y c z n y stosujemy, gdy nie istnieje co najmniej jedna funkcja odwrotna do γ_r. Wtedy, podobnie jak dla rozdziału zasobów, problem można sprowadzić do rozwiązania układu $R-1$ równań z tyloma samymi niewiadomymi, który należy rozwiązać za pomocą procedury numerycznej, aby w konsekwencji uzyskać liczbowe wartości zadań przydzielonych operacjom oraz czasu ich wykonania. Rozważmy teraz ilustracyjny przykład praktyczny dotyczący rozdziału surowców jako szczególnego przypadku rozdzielanych zadań.

Przykład 1.7

W czterech jednocześnie pracujących oddziałach zakładu przetwórstwa owocowego tworzących kompleks operacji równoległych odbywa się produkcja soku owocowego z surowców (owoców), dostarczanych przez dostawców. Operacje polegają tu na wytwarzaniu soku w poszczególnych oddziałach produkcyjnych. Zakładamy, że czas pracy każdego oddziału jest związany z ilością dostarczonego surowca zależnością (1.43) i przyjmujemy następujące dane liczbowe: $k_1 = 4$, $k_2 = 5$, $k_3 = 8$, $k_4 = 10$, $\alpha = 2$ oraz $V = 100$. Wykorzystanie omówionego algorytmu analitycznego prowadzi do następujących wyników. Optymalny rozdział surowców: $v_1^* = 19{,}55$, $v_2^* = 21{,}85$, $v_3^* = 27{,}65$, $v_4^* = 30{,}95$ i minimalny czas produkcji $T^* = 95{,}6$. Otrzymane wartości liczbowe są wyrażone w odpowiednich jednostkach masy oraz czasu, na przykład w tonach i godzinach.

■

Komentarza wymaga sprawa rozszerzenia rozważań na kompleksy operacji, w których występują ograniczenia kolejnościowe w zbiorze operacji. Struktura takich kompleksów nie jest równoległa. Odrzucając przypadki trywialne, jak na przy-

kład strukturę szeregową, łatwo zauważyć, że wtedy w grafie reprezentującym strukturę występują różne drogi o początku i końcu odpowiednio w wierzchołkach reprezentujących zdarzenia polegające na rozpoczęciu wykonywania kompleksu i jego zakończeniu. Drogi te zawierają takie same operacje. Wartość czasu T określona ogólnie w (1.45) jest największym z czasów dla wszystkich takich dróg. W sposób podobny jak w twierdzeniu 1.3 można wykazać, że czas T jest najmniejszy wówczas, gdy czasy dla wspomnianych dróg są takie same. Oznaczmy przez $\boldsymbol{D}_\lambda$, $\lambda = 1, 2, ..., D$ zbiór operacji należących do drogi λ łączącej wierzchołki początkowy i końcowy, gdzie D jest liczbą takich dróg w grafie. Układ równań, z którego można wyznaczyć optymalny rozdział zadań, jest następujący

$$\begin{cases} \sum_{r=1}^{R} v_r^* = V \\ \sum_{r \in \boldsymbol{D}_\lambda} \bar{\gamma}_{\boldsymbol{r}}(v_r^*) = T^*, \ \lambda = 1,2, ..., D . \end{cases} \tag{1.57}$$

Jego rozwiązanie, jeśli istnieje, jest zwykle niejednoznaczne. Prześledźmy prosty przykład obliczeniowy.

Przykład 1.8

Globalną ilość zadań należy tak rozdzielić między trzy operacje o modelach $T_r = k_r v_r$ i o strukturze przedstawionej na rys. 1.10, aby minimalizować czas wykonania wszystkich operacji. Liczba dróg w grafie D wynosi dwa. Dlatego z (1.57) otrzymujemy układ dwóch równań

$$\begin{cases} v_1^* + v_2^* + v_3^* = V \\ k_1 v_1^* + k_2 v_2^* = k_3 v_3^* . \end{cases}$$

Układ ten ma nieskończenie wiele rozwiązań. Na przykład po wyborze v_1^* z przedziału $\left[0, V \frac{k_3}{k_1 + k_3}\right]$ pozostałe szukane wartości możemy obliczyć jako $v_2^* = V \frac{k_3}{k_1 + k_3} - \frac{k_1 + k_3}{k_2 + k_3} \cdot v_1^*$ oraz $v_3^* = V - v_1^* - v_2^*$, $T^* = k_3 v_3^*$.

W przypadku, gdy $D_\lambda = 1$, $\lambda = 1, 2, ..., D$, czyli każda droga zawiera tylko jedną operację, wówczas $D = R$ i otrzymujemy kompleks operacji równoległych, a układ równań (1.57) jest równoważny układowi (1.48) i (1.47) dla $\boldsymbol{v} = \boldsymbol{v}^*$.

■

Zakończmy rozważania dotyczące problemów alokacji jeszcze jednym przykładem, dotyczącym rozdziału zadań w kompleksie R operacji niezależnych.

Przykład 1.9

Dotychczas rozpatrywaliśmy przypadek, gdy zadania (również zasoby) przyjmowały wartości w zbiorze liczb rzeczywistych nieujemnych. Rozważmy teraz sytuację, gdy wielkość zadania v_r przydzielonego operacji r jest całkowitą wielokrotnością pewnego kwantu Δv, czyli $v_r = n_r \cdot \Delta v,\ n_r = 0,1,2,\ldots$. Jest to przypadek zadań o podziale dyskretnym. Ograniczenia dotyczące v_r określamy analogicznie do przypadku ciągłego, to znaczy zbiór dopuszczalnych rozdziałów zadań ma postać

$$\boldsymbol{D}_v = \{\boldsymbol{v} : v_r \geq 0,\ v_r = n_r \Delta v,\ n_r = 0,1,2,\ldots,\ r = 1,2,\ldots,R,\ \sum_{r=1}^{R} v_r = V,\ \Delta v > 0\},$$

czyli globalna ilość zadań V jest całkowitą wielokrotnością kwantu Δv. Naszemu problemowi nadamy teraz inną interpretację. Niech operacja polega na wykonaniu zadań przez realizator. Wielkości zadań o podziale dyskretnym, tworzących zbiór $\boldsymbol{H}$ są wielokrotnościami kwantu Δv. Zbiór zadań przydzielonych operacji r (realizatorowi r) oznaczamy przez $\boldsymbol{H}^r$, przy czym $\boldsymbol{H}^r \cap \boldsymbol{H}^s = \varnothing$ oraz $\bigcup_{r \in \boldsymbol{R}} \boldsymbol{H}^r = \boldsymbol{H}$. Wówczas dla modelu (1.43) i $\alpha = 1$ czas T_r wykonania operacji, czyli czas pracy realizatora, jest sumą czasów wykonywania przez niego przydzielonych zadań, czyli

$$T_r = k_r v_r = \sum_{h \in \boldsymbol{H}^r} \bar{k}_{r,h} \Delta v = \sum_{h \in \boldsymbol{H}^r} \tau_{r,h}\ , \tag{1.58}$$

gdzie $\bar{k}_{r,h}$ – czas wykonania przez realizator r w ramach wykonywania zadania h jednostkowej liczby zadań (kwantu) o wielkości Δv, dla $h \in \boldsymbol{H}^r$,

$\tau_{r,h}$ – czas wykonania zadania h przez realizator r.

■

Dla takiego sformułowania istotne jest rozwiązanie następującego problemu.

Dla danych czasów realizacji zadań $\tau_{r,h}$ należy tak wyznaczyć rozłączne podzbiory $\boldsymbol{H}^1, \boldsymbol{H}^2, \ldots, \boldsymbol{H}^R$, aby minimalizować $T = \max\{T_1, T_2, \ldots, T_R\}$. Jest to przykład problemu szeregowania zadań. Szeregowaniu zadań jest poświęcony następny punkt monografii.

1.3. Problemy szeregowania

Problematyka szeregowania zadań jest bardzo szeroka i poświęcono jej wiele opracowań przeglądowych, np. [9, 7, 42, 86, 6, 31]. W niniejszej pracy zaprezentujemy jedynie wybrany fragment tej problematyki, odpowiadający nieklasycznym problemom szeregowania, które będą prezentowane w rozdziale 3. Kluczowymi pojęciami występującymi w problematyce szeregowania są: z a d a n i e, które wystąpiło już w poprzednim punkcie dotyczącym alokacji, oraz r e a l i z a t o r rozumiany tu jako podmiot wykonujący zadania i mogący mieć różną naturę i interpretację. Zarówno zadania, jak i realizatory są elementami zbiorów dyskretnych i skończonych. Problem szeregowania można ogólnie określić jako wyznaczanie takiego dopuszczalnego przyporządkowania elementów jednego zbioru elementom drugiego zbioru, które jest najlepsze ze względu na przyjęte kryterium szeregowania.

Model kompleksu operacji

Aby podać dokładne sformułowanie problemu, rozpocznijmy od opisu obu zbiorów. Zbiór realizatorów oznaczamy tak samo jak zbiór operacji w poprzednim punkcie, czyli $\boldsymbol{R} = \{1, 2, ..., R\}$, gdzie R jest liczbą realizatorów. Podkreślamy przez to związek interpretacyjny między problemem szeregowania, a specyficznie rozumianym problemem alokacji zadań dyskretnych w kompleksie operacji równoległych, w którym operacja polega na wykonywaniu zadań przez realizator. Jak się okaże, ta analogia jest do utrzymania dla szczególnych założeń, dotyczących zadań, a więc w żadnym razie nie można przyjmować, że wspomniane problemy alokacji i omawiane problemy szeregowania są równoważne. Indeks $r \in \boldsymbol{R}$ określa bieżący realizator. W literaturze używa się innych określeń oznaczających realizator, na przykład procesor, maszyna: w pierwszym przypadku głównie do stosowania w systemach informatycznych, a w drugim – w dyskretnych systemach produkcyjnych. Stosuje się różne klasyfikacje realizatorów, biorące pod uwagę zakres wykonywanych przez nie funkcji oraz parametry określające prędkość wykonywania przez nie zadań. Wynikają z tego różne konkretne i bardziej szczegółowe problemy szeregowania. I tak można podzielić realizatory na wyspecjalizowane (inaczej: dedykowane), czyli takie, które są przeznaczone do wykonywania tylko określonych zadań oraz uniwersalne (inaczej: równoległe) mogące wykonywać wszystkie zadania. W pierwszym przypadku jest formułowany i rozwiązywany trudny problem szeregowania zwany problemem ogólnym (ang.: *job-shop scheduling problem*) lub jego szczególne wersje, na przykład problem przepływowy (ang.: *flow-shop scheduling problem*). Dla problemów tych często stosuje się nazwę harmonogramowanie. Badania nad opra-

cowywaniem algorytmów szeregowania oraz wyznaczaniem ich własności dla szczegółowych przypadków oraz zastosowań, zwłaszcza w dyskretnych systemach produkcyjnych, są prowadzone od wielu lat i ponieważ wykraczają poza zakres niniejszej pracy – nie będą tu omawiane. Skoncentrujemy nasze rozważania na tych przypadkach, w których są stosowane realizatory równoległe. W zależności od prędkości pracy jest stosowany ich podział na:

- realizatory identyczne (ang.: *identical*) – wykonujące wszystkie zadania z równymi prędkościami,
- realizatory jednorodne (ang.: *uniform*) – różniące się prędkościami wykonywania zadań, ale prędkość każdego realizatora jest stała i nie zależy od wykonywanego zadania,
- realizatory dowolne (ang.: *unrelated*), których prędkości wykonywania zadań są różne dla różnych zadań.

Oznaczmy przez $\boldsymbol{H} = \{1, 2, ..., H\}$ zbiór zadań, gdzie H jest liczbą zadań. Bieżący element zbioru $\boldsymbol{H}$, czyli zadanie oznaczamy literą h, a więc $h \in \boldsymbol{H}$. Każde zadanie jest scharakteryzowane przez następujące wielkości.

1. Czas wykonania zadania $\boldsymbol{\tau}_h$. Ogólnie jest to wektor

$$\boldsymbol{\tau}_h = [\tau_{1,h}, \tau_{2,h}, ..., \tau_{R,h}]^{\mathrm{T}}, \tag{1.59}$$

gdzie $\tau_{r,h}$ jest czasem wykonania zadania h przez realizator r. Dla realizatorów identycznych $\tau_{r,h} = \tau_h$, $r = 1, 2, ..., R$ i τ_h jest skalarem. Dla realizatorów jednorodnych $\tau_{r,h} = \tau_h / b_r$, gdzie b_r jest współczynnikiem wyrażającym prędkość wykonywania zadań przez realizator r – w stosunku do innego wybranego realizatora.

2. Moment przybycia (moment gotowości do wykonania zadania) ρ_h . Jest to moment czasu, w którym może być rozpoczęte wykonywanie zadania h. Jeśli ρ_h jest równe dla wszystkich operacji, to przyjmuje się, że $\rho_h = 0$, $h = 1, 2, ..., H$.

3. Termin zakończenia d_h (linia krytyczna). Jest to moment czasu, do którego wykonywanie zadania powinno zostać zakończone.

4. Waga (priorytet) w_h. Jest to współczynnik wyrażający względną ważność zadania h.

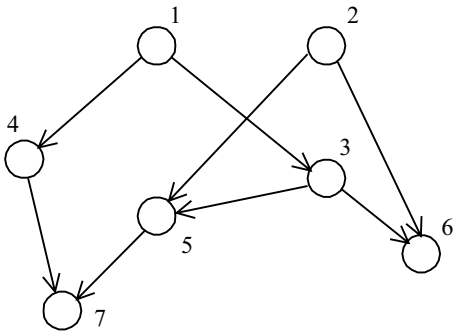
5. Podzielność zadania. Jest to nieliczbowa wielkość charakteryzująca zadanie. Może ono być podzielne i wówczas można przerywać (zawieszać) jego wykonywanie. W przeciwnym razie wykonywanie zadania nie może być przerwane, co oznacza, że realizator przydzielony do wykonania zadania przed jego zakończeniem nie może być użyty do wykonania innego zadania. Jeśli w zbiorze zadań występuje co najmniej jedno zadanie, którego wykonywanie nie może być przerywane, to wszy-

stkie zadania traktujemy jako niepodzielne. W przeciwnym przypadku zadania są podzielne.

Podobnie jak w przypadku operacji dla problemu alokacji, w zbiorze zadań mogą obowiązywać analogicznie zdefiniowane ograniczenia kolejnościowe. Mogą one być opisane za pomocą wprowadzonej w zbiorze $\boldsymbol{H}$ relacji $\boldsymbol{\rho}$, określonej przez warunek „wykonanie zadania h może się rozpocząć bezpośrednio po zakończeniu wykonywania zadania g" i w konsekwencji przez podanie wszystkich uporządkowanych par (g,h) będących w relacji $\boldsymbol{\rho}$. Analitycznym zapisem ograniczeń, czyli par (g,h) mogą być: graf $\boldsymbol{\Theta}$ (1.10) lub macierz $\boldsymbol{\kappa}$ (1.11). Forma prezentacji graficznej jest zwyczajowo różna od sposobu przedstawionego dla problemu alokacji. Stosuje się zwykle konwencję „zadanie w wierzchołku", to znaczy wierzchołki przedstawiają zadania, a ograniczenia kolejnościowe są wyrażone w postaci łuków. Przykład ograniczeń kolejnościowych w zbiorze zadań przedstawiono na rys. 1.11. Zbiór $\boldsymbol{\Theta}$ dla tego przykładu ma postać

$$\boldsymbol{\Theta} = \{(1,3), (1,4), (2,5), (2,6), (3,5), (3,6), (4,7), (5,7)\}.$$

Łatwo spostrzec, że graf z rys. 1.11 jest odpowiednikiem grafu z rys. 1.5. Obie reprezentacje graficzne o różnych konwencjach przedstawiają tę samą strukturę (ograniczenia kolejnościowe): zbioru operacji (rys. 1.5) i zbioru zadań (rys. 1.11). Ogólnie, strukturę można przedstawić w obu konwencjach, chociaż przekształcenie jednego opisu w drugi nie zawsze jest proste i wygodne. Wymaga czasami na przykład wprowadzenia tak zwanych operacji (zadań) pustych w celu zachowania spójności dla konwencji „operacja (zadanie) na łuku". W reprezentacji postaci „zadanie w wierzchołku" graf nie musi być spójny. Przykładowo, dla zadań bez ograniczeń



Rys. 1.11. Przykład ograniczeń kolejnościowych w zbiorze zadań

kolejnościowych graf składa się z samych wierzchołków, bez żadnego łuku. Jego odpowiednikiem w drugiej konwencji jest graf równoległy. Jeśli w zbiorze zadań występuje co najmniej jedno ograniczenie kolejnościowe, to o zadaniach mówimy, że są zależne, w przeciwnym przypadku są niezależne.

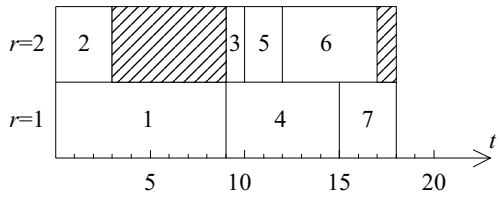
Jak to już zostało wspomniane, celem szeregowania jest przyporządkowanie wszystkich elementów zbioru ***R*** do elementów zbioru ***H***. Przyporządkowanie takie nosi nazwę uszeregowania, jeśli są spełnione następujące warunki.

- Każdy realizator w jednym momencie czasu wykonuje co najwyżej jedno zadanie.
- Każde zadanie jest wykonywane w skończonym przedziale czasu, począwszy od momentu ρ_h.
- Wszystkie zadania są wykonane.
- Są zachowane ograniczenia kolejnościowe.
- Jest zachowana własność niepodzielności dla zadań niepodzielnych, a dla zadań podzielnych liczba przerwań jest skończona.

Wygodną reprezentacją graficzną uszeregowania jest tak zwany wykres Gantta, w którym na osi czasu są przedstawione kolejno czasy pracy poszczególnych realizatorów. Czasy wykonywania zadań są reprezentowane przez prostokąty o stałej szerokości i o długościach proporcjonalnych do tych czasów. Na rys. 1.12 przedstawiono przykładowy wykres Gantta dla siedmiu zadań niepodzielnych o równych momentach gotowości i ograniczeniach kolejnościowych przedstawionych na rys. 1.11. Przyjęto, że są dane dwa realizatory identyczne, a czasy wykonywania zadań są następujące: $\tau_1 = 9$, $\tau_2 = 3$, $\tau_3 = 1$, $\tau_4 = 6$, $\tau_5 = 2$, $\tau_6 = 5$, $\tau_7 = 3$.

Prostokąty zakreskowane oznaczają okresy bezczynności realizatorów, które mogą wystąpić w przypadku zadań zależnych.

Problem szeregowania, czyli wyznaczania uszeregowania jest istotny tylko wtedy, gdy $H > R$. Zwykle liczba zadań jest znacznie większa niż liczba realizatorów.



Rys. 1.12. Wykres Gantta

Jakość uszeregowania jest oceniana przez kryteria jakości szeregowania. Większość z nich jest obliczana na podstawie momentów zakończenia wykonywania zadań C_h, czyli

$$Q = f(C_1, C_2, ..., C_H), \tag{1.60}$$

gdzie Q jest kryterium jakości szeregowania, a f jest funkcją rzeczywistą o wartościach nieujemnych, spełniającą warunek

$$f(C_1, C_2, ..., C_H) < f(C_1^{'}, C_2^{'}, ..., C_H^{'}) \rightarrow (\exists\, h \in \boldsymbol{H})\,(C_h < C_h^{'}).$$

Momenty C_h są podstawą do wyznaczania innych wielkości stosowanych w kryteriach jakości, a mianowicie:

$F_h = C_h - \rho_h$ – czasu przepływu (czasu przebywania zadania h w systemie szeregowania),

$L_h = C_h - d_h$ – opóźnienia w wykonywaniu zadania h,

$T_h = \max\{C_h - d_h, 0\}$ – spóźnienia w wykonywaniu zadania h,

$E_h = \max\{d_h - C_h, 0\}$ – przyspieszenia w wykonywaniu zadania h,

U_h – wskaźnika terminowości wykonania zadania h ($U_h = 1(0)$, jeśli $C_h > d_h$ ($C_h \le d_h$)).

Najczęściej są stosowane następujące kryteria szeregowania Q:

– długość uszeregowania

$$C_{\max} = \max_{h=1,2,...,H}\{C_h\}, \tag{1.60a}$$

– średni czas przepływu

$$\overline{F} = \frac{1}{H}\sum_{h=1}^{H} F_h, \tag{1.60b}$$

– średni ważony czas przepływu

$$\overline{F}_w = \frac{1}{H}\sum_{h=1}^{H} \frac{w_h F_h}{\overline{w}}, \tag{1.60c}$$

gdzie $\overline{w} = \sum_{h=1}^{H} w_h$,

– maksymalne opóźnienie

$$L_{\max} = \max_{h=1,2,...,H}\{L_h\}, \tag{1.60d}$$

– średnie spóźnienie

$$\bar{T} = \frac{1}{H}\sum_{h=1}^{H} T_h \,, \tag{1.60e}$$

– średnie ważone spóźnienie

$$\bar{T}_w = \frac{1}{H}\sum_{h=1}^{H} \frac{w_h T_h}{\bar{w}} \,, \tag{1.60f}$$

– średnie przyspieszenie

$$\bar{E} = \frac{1}{H}\sum_{h=1}^{H} E_h \,, \tag{1.60g}$$

– średnie ważone przyspieszenie

$$\bar{E}_w = \frac{1}{H}\sum_{h=1}^{H} \frac{w_h E_h}{\bar{w}} \,, \tag{1.60h}$$

– liczba zadań wykonanych po terminie (spóźnionych)

$$U = \sum_{h=1}^{H} U_h \,. \tag{1.60i}$$

Dla danych: zbioru zadań $\boldsymbol{H}$, zbioru realizatorów $\boldsymbol{R}$ wraz z ich charakterystykami, m.in. czasów wykonywania zadań, które tworzą macierz

$$\boldsymbol{\tau} \stackrel{\Delta}{=} [\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, ..., \boldsymbol{\tau}_H] = [\tau_{r,h}]_{\substack{r=1,2,...,R \\ h=1,2,...,H}} \,,$$

klasyczny problem szeregowania polega na wyznaczeniu takiego uszeregowania, które minimalizuje wybrane kryterium jakości (1.60).

Algorytmy szeregowania są opracowywane dla konkretnych problemów szeregowania. W celu usystematyzowania problemów zaproponowano ich klasyfikację [44, 10], wyrażoną w postaci zapisu trójpolowego $\alpha|\beta|\gamma$. Pole α opisuje liczbę i rodzaj zastosowanych realizatorów, pole β – zadania oraz ich współzależności, a pole γ – rodzaj kryterium jakości szeregowania.

Algorytmy rozwiązania

Problemy szeregowania zadań są w większości NP-trudnymi problemami optymalizacyjnymi (np. [43, 4]). Niekiedy tak nie jest, na przykład dla jednostkowych czasów wykonywania lub w szczególnych przypadkach dla zadań podzielnych i niezależnych, np. [6, 31]. Wtedy optymalne algorytmy szeregowania mają złożoność

wielomianową. Dla problemów NP-trudnych można wyznaczać algorytmy dokładne (optymalne), ale o złożoności wykładniczej. Stosuje się wtedy zwykle dwie metody wyznaczania rozwiązania: programowanie dynamiczne lub zasadę podziału i ograniczeń. Algorytmy efektywne o złożoności wielomianowej dają rozwiązania nieoptymalne. Jeżeli jakość otrzymanego rozwiązania można oszacować w stosunku do rozwiązania optymalnego, to takie algorytmy są nazywane przybliżonymi, a jeśli takiego oszacowania brak, to – heurystycznymi. W celach ilustracyjnych zostanie przedstawionych kilka przykładów, reprezentujących wymienione rodzaje algorytmów, które zostały wybrane z punktu widzenia ich wykorzystania w dalszych rozdziałach pracy.

Przykładem problemu, dla którego można wyznaczyć algorytm optymalny o złożoności wielomianowej, jest zagadnienie szeregowania zadań niezależnych i podzielnych na realizatorach identycznych w celu minimalizacji $C_{\max}$, np. [6].

1. Rozpocznij wykonywanie dowolnego zadania na dowolnym realizatorze.

2. Wybierz dowolne nie uszeregowane jeszcze zadanie i rozpocznij jego wykonywanie na tym samym realizatorze w chwili zakończenia wykonywania poprzedniego zadania. Powtarzaj ten krok do chwili, gdy wszystkie zadania zostaną uszeregowane lub zostanie osiągnięty moment czasu

$$C^*_{\max} = \frac{1}{R}\sum_{h=1}^{H}\tau_h \, .$$

3. Część zadania pozostałą do wykonania po osiągnięciu momentu $C^*_{\max}$ przydziel do innego wolnego realizatora i przejdź do kroku 2.

Algorytm ten ma złożoność obliczeniową O(H).

Rozważane zagadnienie staje się problemem NP–trudnym (nawet dla dwóch realizatorów), gdy założymy niepodzielność zadań. Algorytmem przybliżonym o bardzo dobrych własnościach, który rozwiązuje ten problem, jest algorytm LPT (ang. *Longest Processing Time*), np. [31].

1. Uporządkuj zbiór zadań w kolejności nierosnących czasów ich wykonania, tworząc ciąg (listę) zadań.

2. Przydzielaj kolejne zadania z ciągu do aktualnie wolnych realizatorów.

Złożoność obliczeniowa algorytmu wynosi O($H \log H$).

Inną wielkością charakteryzującą algorytm przybliżony jest tak zwane oszacowanie najgorszego przypadku

$$\kappa \stackrel{\Delta}{=} \frac{\overline{Q}^{\,\mathrm{alg}}}{Q^*} \, , \tag{1.61}$$

gdzie $\overline{Q}^{\,\mathrm{alg}}$ oraz Q^* to odpowiednio największa możliwa (najgorsza) wartość kryterium jakości szeregowania uzyskana przez oceniany algorytm oraz wartość kryterium dla uszeregowania optymalnego. Dla algorytmu LPT $\kappa = \frac{4}{3} - \frac{1}{3R}$.

Jak już wspomniano, problemy szeregowania są problemami optymalizacji dyskretnej. Przedstawimy teraz sformułowanie odpowiedniego problemu optymalizacyjnego dla zagadnienia szeregowania zadań niepodzielnych i niezależnych na realizatorach dowolnych w celu minimalizacji $C_{\max}$. Niech $\boldsymbol{c} = [c_{r,h}]_{\substack{r=1,2,\ldots,R \\ h=1,2,\ldots,H}}$ będzie binarną macierzą decyzyjną, której elementy mają następującą interpretację

$$c_{r,h} = \begin{cases} 1, & \text{jeśli realizator } r \text{ wykonuje zadanie } h, \\ 0, & \text{w przeciwnym przypadku.} \end{cases} \tag{1.62}$$

Aby macierz $\boldsymbol{c}$ mogła określać uszeregowanie, muszą być spełnione ograniczenia

$$\sum_{r=1}^{R} c_{r,h} = 1, \quad h = 1, 2, \ldots, H, \tag{1.63}$$

które oznaczają, że każde zadanie musi być wykonane jeden raz. Długość uszeregowania wyrażamy wzorem

$$C_{\max} = \max_{r=1,2,\ldots,R} \left\{ \sum_{h=1}^{H} c_{r,h}\, \tau_{r,h} \right\}. \tag{1.64}$$

Problem optymalizacyjny polega na wyznaczeniu takiej binarnej macierzy $\boldsymbol{c}^*$ spełniającej ograniczenia (1.63), dla której wartość kryterium (1.64) jest najmniejsza.

W pracy [47] podano następujący przybliżony algorytm szeregowania.

1. Nadaj wszystkim elementom macierzy $\boldsymbol{c}$ oraz czasowi pracy wszystkich realizatorów wartość zero, czyli $c_{r,h} = 0$ oraz $T_r = 0$.

2. Wybierz dowolne nie uszeregowane jeszcze zadanie h i znajdź dla niego realizator r o najmniejszym numerze, dla którego jest spełniona nierówność

$$T_r + \tau_{r,h} \le T_s + \tau_{s,h}, \quad s = 1, 2, \ldots, R.$$

Następnie ustal $c^*_{r,h} = 1$ oraz podstaw $T_r = T_r + \tau_{r,h}$. Powtarzaj ten krok, aż wszystkie zadania zostaną uszeregowane.

Złożoność obliczeniowa tego algorytmu jest niewielka, to znaczy wynosi $O(H)$, natomiast jakość oceniana przez wielkość (1.61) jest dosyć słaba, mianowicie $\kappa = R$.

Rozważmy przykład ilustrujący algorytm rozwiązania.

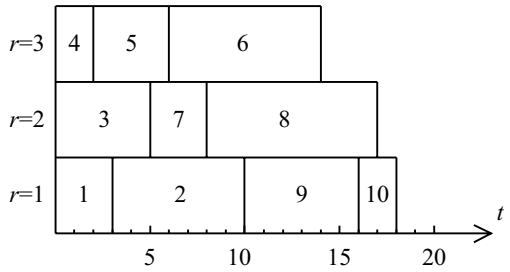
Przykład 1.10

Niech $H = 10$ i $R = 3$. Czasy wykonania zadań podano w tabeli 1.1.

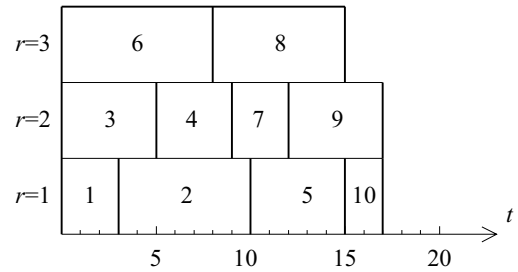
Rozwiązanie uzyskane z wykorzystaniem omówionego algorytmu szeregowania przedstawiono na wykresie Gantta (rys. 1.13). Długość uszeregowania $C_{\max}^{\text{alg}} = 18$. Na rys. 1.14 jest zaprezentowane jedno z uszeregowań optymalnych,

Tabela 1.1. Czasy $\tau_{r,h}$ w przykładzie 1.10

r \ h	1	2	3	4	5	6	7	8	9	10
1	3	7	10	6	5	12	4	10	6	2
2	13	13	5	4	6	10	3	9	5	4
3	3	13	13	2	4	8	2	7	4	6



Rys. 1.13. Uszeregowanie uzyskane przez algorytm przybliżony w przykładzie 1.10



Rys. 1.14. Uszeregowanie optymalne w przykładzie 1.10

uzyskane przez zastosowanie przeglądu zupełnego, dla którego $C^*_{\max} = 17$. Stosunek $C^{\text{alg}}_{\max} / C^*_{\max} \approx 1{,}06 < \kappa = 3$.

■

Na podstawie analizy rozpatrywanego przykładu należy podać jeszcze dwie uwagi ogólne.

1. Optymalne rozwiązanie problemu szeregowania nie musi być jednoznaczne.

2. Dla kryterium w postaci długości uszeregowania i dla zadań niezależnych kolejność wykonywania zadań przez realizator jest dowolna, a więc faktycznie problem szeregowania polega w tym przypadku na wyznaczaniu podzbiorów zadań do wykonania przez poszczególne realizatory.

Rozważymy teraz to samo zagadnienie, ale dla kryterium $L_{\max}$ i sformułujemy je jako problem optymalizacyjny. Bez straty ogólności przyjmijmy, że linie krytyczne tworzą ciąg niemalejący, to znaczy $d_g \le d_h$, dla $g, h = 1,2,...,H$, $g < h$. Wprowadźmy pojęcie przedziału czasu, będącego czasem od początku procedury szeregowania do momentu równego linii krytycznej d_l, gdzie $l = 1, 2, ..., H$ jest indeksem przedziału czasu. Niech $\boldsymbol{a} = [a_{r,h,l}]_{\substack{r=1,2,...,R \\ h,\, l=1,2,...,H}}$ będzie trójwymiarową binarną macierzą decyzyjną określoną następująco:

$$a_{r,h,l} = \begin{cases} 1, \text{ jeśli wykonanie zadania } h \text{ przez realizator } r \\ \quad \text{rozpoczyna się przed końcem przedzialu } l, \\ 0, \text{ w przeciwnym przypadku.} \end{cases}$$

Na elementy macierzy $\boldsymbol{a}$ nakładamy ograniczenia

$$\sum_{r=1}^{R} a_{r,h,l} = 1, \quad h = 1, 2, ..., H, \quad l \ge h, \tag{1.65}$$

które zapewniają, że każde zadanie będzie wykonane jeden raz. Jest również prawdziwa własność

$$a_{r,h,l} = 1 \rightarrow a_{r,h,m} = 1 \text{ dla } m = 2, 3, ..., H, \quad m > l.$$

Maksymalne opóźnienie zapisujemy jako

$$L_{\max} = \max_{r=1,2,...,R} \left\{ \max_{l=1,2,...,H} \left\{ \sum_{h=1}^{H} a_{r,h,l}\, \tau_{r,h} - d_l \right\} \right\}. \tag{1.66}$$

Wtedy rozważany problem polega na wyznaczeniu takiej binarnej macierzy $\boldsymbol{a}^*$ spełniającej ograniczenia (1.65), dla której wartość kryterium (1.66) jest najmniejsza. W pracy [6] podano przybliżony algorytm rozwiązania dla szczególnego przypadku realizatorów identycznych.

1. Uporządkuj zbiór zadań w kolejności niemalejących linii krytycznych, tworząc odpowiedni ciąg zadań.

2. Przydziel kolejne nie uszeregowane jeszcze zadanie do realizatora, na którym, o ile to możliwe, można je wykonać bez opóźnienia i jak najbliżej jego linii krytycznej. Powtarzaj ten krok, aż do wyczerpania ciągu utworzonego w kroku 1.

3. Jeśli w otrzymanym uszeregowaniu są okresy bezczynności realizatorów, to przesuń wszystkie zadania w kierunku początku procedury szeregowania tak, aby okresy te zlikwidować.

Rozważmy teraz elementarny przykład ilustrujący algorytm rozwiązania.

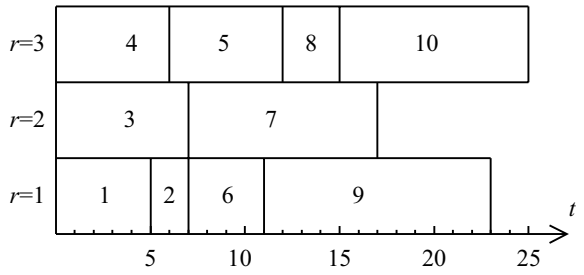
Przykład 1.11

Niech podobnie jak w przykładzie 1.10 $H = 10$ i $R = 3$. Teraz zakładamy, że realizatory są identyczne, a czasy wykonywania zadań przez realizatory są takie same jak czasy przedstawione w tabeli 1.1 dla $r = 1$. Ponadto, są dane linie krytyczne d_h. Po przenumerowaniu zadań, aby zachować warunek, że linie krytyczne tworzą ciąg niemalejący czasy wykonywania i linie krytyczne przedstawiono w tabeli 1.2.

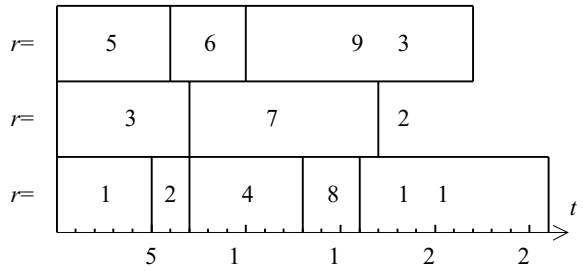
Tabela 1.2. Czasy τ_h i linie krytyczne d_h w przykładzie 1.11

h	1	2	3	4	5	6	7	8	9	10
τ_h	5	2	7	6	6	4	10	3	12	10
d_h	7	8	10	11	12	13	15	17	20	24

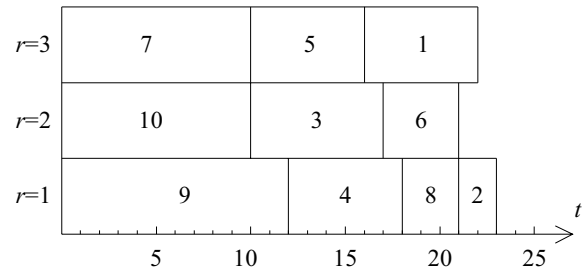
Rozwiązanie przedstawiono na wykresie Gantta (rys. 1.15). Opóźnienia dla kolejnych zadań mają wartości: $L_1 = -2$, $L_2 = -1$, $L_3 = -3$, $L_4 = -5$, $L_5 = -5$, $L_6 = -2$, $L_7 = 2$, $L_8 = -2$, $L_9 = 3$, $L_{10} = 1$, a więc $L_{\max} = 3$. Przykładowe rozwiązanie optymalne uzyskane przez przegląd wszystkich rozwiązań przedstawiono na rys. 1.16. Optymalna wartość kryterium $L^*_{\max} = 2$. Dla obu uszeregowań wartość długości uszeregowania $C_{\max}$ wynosi odpowiednio 25 i 26. Gdyby dla czasów danych jak w tabeli 1.2. rozwiązywać problem szeregowania z kryterium $C_{\max}$, wówczas po zastosowaniu algorytmu LPT można uzyskać rozwiązanie przedstawione na rys. 1.17. Długość uszeregowania $C^{\mathrm{LPT}}_{\max} = 23$ jest oczywiście mniejsza niż ta, która



Rys. 1.15. Uszeregowanie nieoptymalne w sensie $L_{\max}$ w przykładzie 1.11

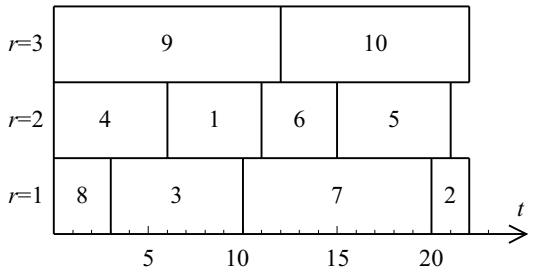


Rys. 1.16. Uszeregowanie optymalne w sensie $L_{\max}$ w przykładzie 1.11



Rys. 1.17. Uszeregowanie uzyskane przez algorytm LPT w przykładzie 1.11

odpowiada rozwiązaniu z rys. 1.15 i rys. 1.16. Jest ona jednak większa niż $C^*_{\max} = 22$ dla uszeregowania optymalnego (rys. 1.18), które również uzyskano stosując przegląd zupełny. Stosunek $C^{\mathrm{LPT}}_{\max} / C^*_{\max} \approx 1{,}05$ jest mniejszy niż $\kappa = \frac{4}{3} - \frac{1}{3 \cdot 3} = \frac{11}{9} \approx 1{,}22$.



Rys. 1.18. Uszeregowanie optymalne w sensie $C_{\max}$ w przykładzie 1.11

■

1.4. Probabilistyczne problemy podejmowania decyzji

W poprzednich punktach zakładaliśmy, że wszystkie informacje o wielkościach właściwych dla kompleksu operacji, a więc na przykład o operacjach, zasobach, zadaniach, realizatorach, strukturze kompleksu były znane, a ich realizacje były dokładnie określone. Rozważaliśmy problemy deterministyczne. Teraz podamy przykłady dwu problemów, dla których wymienione warunki nie zachodzą, to znaczy informacja o niektórych wielkościach właściwych dla kompleksu operacji nie jest pełna lub pewna. Ograniczymy się do szczególnego przypadku tak zwanych problemów probabilistycznych (stochastycznych), w których informacja o pewnych wielkościach jest określona z wykorzystaniem rozkładów prawdopodobieństwa i ma charakter stochastyczny. Najpierw rozważymy szczególny przypadek problemu rozdziału zasobów w kompleksie operacji niezależnych, w którym parametry modeli operacji są realizacjami zmiennych losowych. Następnie krótko scharakteryzujemy obszerną problematykę obsługi zadań, nawiązując do punktu 1.3, dotyczącego szeregowania zadań.

1.4.1. Rozdział zasobów i szeregowanie zadań dla losowych parametrów

Brak determinizmu w rozpatrywanym problemie polega na niedokładnej znajomości modeli operacji. Ograniczamy rozważania do modeli czasowych (1.4). Często w praktyce występuje sytuacja, gdy wartości parametrów $\boldsymbol{a}_r$ modelu nie są znane lub pewne. Możliwym i stosowanym sposobem opisu operacji jest wtedy model probabilistyczny, w którym zakładamy, że parametr $\boldsymbol{a}_r$ jest realizacją zmiennej losowej $\underline{\boldsymbol{a}}_r$, np. [12, 13, 15]. W celu uniknięcia niepotrzebnej komplikacji zapisu rozważymy w dalszym ciągu przypadek, gdy losowy jest tylko jeden element wektora

$\boldsymbol{a}_r$, oznaczany jako z_r. Informacja o parametrze jest dana nie w postaci wartości liczbowej jak w przypadku deterministycznym, ale w formie rozkładu prawdopodobieństwa – ogólnie lub jako jedna z funkcji rozkładu, na przykład gęstość, wartość oczekiwana – w przypadkach szczególnych, gdy funkcje takie istnieją. Jeśli parametr jest zmienną losową, to zmienną losową jest również czas wykonania operacji, czyli

$$\underline{T}_r = \gamma_r^{\mathrm{E}}(u_r; \underline{z}_r). \tag{1.67}$$

Przypadek tak zwanej pełnej informacji probabilistycznej oznacza znajomość rozkładu prawdopodobieństwa zmiennych losowych $\underline{z}_r$, $r = 1, 2, ..., R$. W szczególnym i najczęściej rozpatrywanym przypadku zakładamy istnienie i znajomość gęstości rozkładu prawdopodobieństwa f_r zmiennych losowych $\underline{z}_r$. Wówczas problem rozdziału zasobów w kompleksie operacji niezależnych w warunkach probabilistycznych polega na minimalizacji względem rozdziału zasobów $\boldsymbol{u}$ wartości oczekiwanej czasu wykonania kompleksu operacji

$$\underset{\underline{T}}{\mathrm{E}}(\underline{T}) = \underset{\underline{T}_1, \underline{T}_2, ..., \underline{T}_R}{\mathrm{E}}(\max\{\underline{T}_1, \underline{T}_2, ..., \underline{T}_R\}), \tag{1.68}$$

z ograniczeniami (1.17) i (1.18).

Korzystając z zależności określającej gęstość statystyki pozycyjnej (np. [99]) wartość oczekiwaną (1.68) możemy przedstawić w postaci analitycznej jako kryterium jakości rozdziału zasobów zależne od wektora $\boldsymbol{u}$, to znaczy

$$T^{\mathrm{E}}(\boldsymbol{u}) = \int_0^{+\infty} t \left\{ \sum_{r=1}^{R} \left[f_{T_r}(t; u_r) \prod_{\substack{s=1 \\ s \neq r}}^{R} \left(\int_0^t f_{T_s}(\lambda; u_s)\, d\lambda \right) \right] \right\} dt, \tag{1.69}$$

gdzie f_{T_r} – gęstość prawdopodobieństwa zmiennych losowych $\underline{T}_r$. Gęstość tę można wyznaczyć na podstawie znanych gęstości f_r, jeżeli istnieją funkcje odwrotne do γ_r^{E} względem z_r. Wtedy dla $t \in [0, T_r]$

$$f_{T_r}(t; u_r) = f_r[\gamma_r^{\mathrm{E}\,-1}(t; u_r)] \left| \frac{\partial \gamma_r^{\mathrm{E}\,-1}(t; u_r)}{\partial t} \right|. \tag{1.70}$$

Możliwość uzyskania analitycznej postaci kryterium $T^{\mathrm{E}}(\boldsymbol{u})$ zależy od postaci gęstości f_r oraz modeli γ_r^{E}. W przypadku gdy można ją uzyskać, rozpatrywany problem polega na minimalizacji kryterium względem $\boldsymbol{u}$ dla ograniczeń (1.17) i (1.18). W przeciwnym przypadku należy stosować metody numeryczne.

Przykład 1.12

Rozważmy kompleks dwóch operacji o modelach typu (1.9), które zapiszemy w postaci $T_r = g(u_r)\, z_r$, gdzie $g(u_r) = \dfrac{1}{u_r^{\alpha}}$ oraz $\dfrac{1}{k_r} = z_r$ jest realizacją zmiennej losowej $\underline{z}_r$ o rozkładzie

$$f_r(z_r) = \begin{cases} b\exp(-b\, z_r), & z_r \geq 0 \\ 0, & z_r < 0 \end{cases}, \quad b > 0, \quad r = 1, 2.$$

Po żmudnych przekształceniach otrzymujemy wyrażenie na średni czas wykonania operacji

$$T^{\mathrm{E}}(u_1, u_2) = \frac{1}{b} \frac{\left(\dfrac{u_2}{u_1}\right)^{\alpha} + \left(\dfrac{u_1}{u_2}\right)^{\alpha} + 1}{u_1^{\alpha} + u_2^{\alpha}},$$

które po podstawieniu $u_2 = U - u_1$ wynikającym z ograniczeń, należy minimalizować względem u_1.

■

Ze względu na trudności obliczeniowe, często zachodzi potrzeba rozwiązania tak zwanego problemu zastępczego, który umożliwia jedynie uzyskanie rozwiązania przybliżonego. Polega on na zastąpieniu kryterium (1.69) przez kryterium postaci

$$\tilde{T}^{\mathrm{E}}(\boldsymbol{u}) = \max\{\underset{\underline{T}_1}{\mathrm{E}}(\underline{T}_1), \underset{\underline{T}_2}{\mathrm{E}}(\underline{T}_2), \ldots, \underset{\underline{T}_R}{\mathrm{E}}(\underline{T}_R)\}, \tag{1.71}$$

co natychmiast prowadzi do przypadku deterministycznego, ponieważ do równań (1.21) w miejsce $\bar{\gamma}_r(u_r^*)$ wystarczy wstawić

$$\underset{\underline{T}_r}{\mathrm{E}}(\underline{T}_r) = \underset{\underline{z}_r}{\mathrm{E}}\left(\gamma_r^{\mathrm{E}}(u_r; \underline{z}_r)\right) \stackrel{\Delta}{=} \tilde{\gamma}_r^{\mathrm{E}}(u_r^*). \tag{1.72}$$

Przykład 1.13

Dla danych jak w przykładzie 1.12

$$f_{T_r}(t; u_r) = b\exp\left(-b\frac{t}{g(u_r)}\right)\frac{1}{g(u_r)}.$$

Wtedy

$$\underset{\underline{T}_r}{\mathrm{E}}(\underline{T}_r) = \int_0^{+\infty} t\, b \exp\left(-b\frac{t}{g(u_r)}\right)\frac{1}{g(u_r)}\,\mathrm{d}t = \frac{g(u_r)}{b} = \frac{1}{b \cdot u_r^{\alpha}} \quad \stackrel{\Delta}{=} \tilde{\gamma}_r^{\mathrm{E}}(u_r)$$

■

Rozważania w przypadku losowego parametru $\boldsymbol{a}_r$ ograniczyliśmy do kompleksu operacji niezależnych ze względu na fakt, że dla kompleksu operacji o strukturze dowolnej można je rozszerzyć w sposób podobny jak dla przypadku deterministycznego.

Rozpatrywanie problemu zastępczego dla rozdziału zasobów zamiast problemu minimalizacji kryterium (1.69) jest przykładem często stosowanego i ogólnego podejścia do rozwiązywania różnych problemów decyzyjnych w warunkach niedeterministycznych. Polega ono na tym, że zamiast rozwiązywać niedeterministyczny problem decyzyjny sformułowany na podstawie niepełnej informacji o obiekcie podejmowania decyzji, postępujemy w sposób następujący. Najpierw dokonujemy determinizacji, czyli ukonkretnienia informacji o obiekcie, a następnie dla takiej – już deterministycznej – wiedzy o obiekcie rozwiązujemy deterministyczny problem decyzyjny. W omówionym przypadku właśnie tak postąpiliśmy. Deterministyczny problem rozdziału zasobów był rozwiązywany dla ukonkretnionych czasów wykonywania operacji. Ukonkretnienie polegało na obliczeniu wartości średniej zmiennej losowej.

W sposób analogiczny możemy rozpatrzyć problem szeregowania z losowymi parametrami. Rozważmy go dla przypadku szeregowania zadań niezależnych i niepodzielnych w celu minimalizacji kryterium $C_{\max}$, czyli dla problemu minimalizacji kryterium (1.64) z ograniczeniami (1.62) i (1.63). Zakładamy, że nie są dokładnie znane wartości czasów wykonywania operacji $\tau_{r,h}$. Przyjmujemy model probabilistyczny, w którym są one realizacjami wzajemnie niezależnych zmiennych losowych $\underline{\tau}_{r,h}$. Wtedy informacja o czasach, w przypadku tak zwanej pełnej informacji probabilistycznej, jest dana w postaci funkcji gęstości rozkładów prawdopodobieństwa $f_{r,h}$ – jeżeli funkcje takie istnieją. W przeciwnym razie wykorzystujemy dystrybuanty rozkładów prawdopodobieństwa. Wtedy kryterium jakości szeregowania może być określone jako wartość oczekiwana kryterium $C_{\max}$, brana ze względu na wszystkie zmienne losowe $\underline{\tau}_{r,h}$, tworzące macierz $\underline{\boldsymbol{\tau}} = [\underline{\tau}_{r,h}]_{\substack{r=1,2,\ldots,R \\ h=1,2,\ldots,H}}$, czyli

$$C_{\max}^{\mathrm{E}} = \underset{\underline{\tau}}{\mathrm{E}}\left(\max_{r=1,2,\ldots,R} \left\{ \sum_{h=1}^{H} c_{r,h}\, \underline{\tau}_{r,h} \right\} \right). \tag{1.73}$$

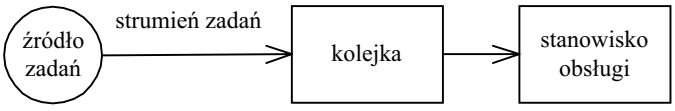
Problem bez determinizacji polega więc na minimalizacji (1.73) z ograniczeniami (1.62) i (1.63). Rozwiązanie tak sformułowanego problemu wiąże się z dużymi problemami nie tylko natury obliczeniowej, dlatego często stosuje się wspomniany sposób determinizacji, który w tym przypadku polega na obliczeniu wartości oczekiwanych zmiennych losowych $\underline{\tau}_{r,h}$, a następnie na rozwiązaniu problemu deterministycznego, np. [66].

1.4.2. Obsługa zadań

Rozważymy teraz krótko problem obsługi zadań. W problemach szeregowania zadań (p. 1.3) należało wyznaczać kolejność wykonywania zadań na poszczególnych realizatorach dla zbioru zadań ***H***, który był z góry określony przed podjęciem i realizacją decyzji. Obecnie zbiór ***H*** nie jest dany a priori lub nie pojawia się co pewien czas, ale poszczególne zadania pojawiają się na bieżąco, w pewnych odstępach czasu. Mówimy wówczas o o b s ł u d z e z a d a ń, a odpowiedni system nazywamy systemem obsługi. Problematyka obsługi zadań jest także często nazywana teorią kolejek lub teorią masowej obsługi. Jest ona ważną częścią badań operacyjnych. Ze względu na stosowany aparat matematyczny jest również częścią teorii procesów stochastycznych.

W systemie obsługi wyróżniamy źródło zadań, z którego wychodzi strumień zadań, kolejkę, w której zadania czekają na obsługę i stanowisko obsługi (rys. 1.19).

Zagadnienia obsługi zadań są inspirowane przez konkretne sytuacje praktyczne, w których występują zjawiska losowego pojawiania się zadań, oczekiwania na ich przetwarzanie, a także realizacji zadań o nieznanych z góry czasach wykonywania. Wzorcowym przykładem teorii masowej obsługi jest praca central telefonicznych. Inne ważne obszary zastosowań to obsługa klientów w bankach, urzędach itp., obsługa operacji technologicznych w dyskretnych systemach produkcyjnych, obsługa ruchu lotniczego przez lotniska. Ostatnio szczególnego znaczenia nabierają proble-



Rys. 1.19. Prosty system obsługi

my obsługi zadań w systemach informatycznych. Nazwy takie jak zadania, operacja, klient, abonent są obejmowane wspólnym pojęciem zgłoszenie. Jest ono rozumiane jako żądanie spełnienia przez system określonej czynności. Ważne jest, że zgłoszenie jest utożsamiane z jego nośnikiem. Aby nawiązać do treści poprzednich punktów, w dalszym ciągu pozostaniemy przy terminie zadanie. Pozostałe podstawowe pojęcia systemu obsługi, czyli kolejka i stanowisko obsługi mogą mieć bardziej skomplikowaną strukturę, to znaczy może być wiele kolejek oraz wiele różnych stanowisk obsługi (mówimy wtedy o wielu kanałach obsługi). W takiej sytuacji mówimy już nie o systemie obsługi, ale o sieciach obsługi lub sieciach kolejkowych, np. [38, 39].

W teorii masowej obsługi wyróżniamy problemy analizy oraz podejmowania decyzji (syntezy). Celem analizy jest podanie wartości podstawowych wskaźników charakteryzujących proces obsługi i umożliwiających ocenę jakości pracy systemu kolejkowego. Efektem rozwiązywania problemów podejmowania decyzji jest m.in. wybór optymalnej struktury i organizacji obsługi. Uzyskiwane wyniki są ważne zarówno dla użytkowników, jak i dla zarządzających systemami obsługi. Metody rozwiązywania tych problemów, zwłaszcza dotyczy to problemów analizy, można podzielić na analityczne i symulacyjne. Metody analityczne sprowadzają się do wyznaczenia oraz rozwiązania układów równań różniczkowych, zawierających prawdopodobieństwa zdarzeń występujących w procesie obsługi. W metodach symulacyjnych odwzorowuje się pracę rozpatrywanego systemu obsługi, a wielokrotna realizacja procesu obsługi umożliwia uzyskanie empirycznych wskaźników jakości badanego systemu obsługi. Obecnie istnieje wiele specjalistycznych pakietów programowych, zawierających m.in. funkcje analizy systemów i sieci kolejkowych.

Pojęcia podstawowe i klasyfikacja

Na rysunku 1.19 podano trzy główne elementy prostego systemu obsługi. Scharakteryzujemy teraz krótko trzy podstawowe pojęcia, czyli strumień zadań, regulamin obsługi kolejki oraz proces obsługi, które kolejno odpowiadają źródłu zadań, kolejce i stanowisku obsługi z rys. 1.19.

Strumień zadań

Wielkość ta jest określona przez momenty czasu, w których pojawiają się zadania lub przez odstępy czasu miedzy kolejnymi zgłoszeniami zadań $\tau_h = t_h - t_{h-1}$, gdzie t_h jest momentem pojawienia się zadania w systemie (to znaczy w kolejce lub na stanowisku obsługi, gdy jest wolne). Gdy zgłoszenia zadań są losowe, interwały τ_h są realizacjami zmiennych losowych. Informacja o pojawieniu się zadania jest wtedy dana bądź przez dystrybuantę rozkładu prawdopodobieństwa $F(t)$, która ma

interpretację prawdopodobieństwa zdarzenia polegającego na tym, że długość przedziału czasu między zgłoszeniami jest mniejsza niż t, bądź przez odpowiednią gęstość rozkładu prawdopodobieństwa, jeśli taka istnieje. Szczególne znaczenie w teorii masowej obsługi ma rozkład wykładniczy o funkcji gęstości

$$f(t) = \begin{cases} \lambda \exp(-\lambda t), & t \geq 0, \\ 0, & t < 0, \end{cases} \tag{1.74}$$

gdzie $\lambda > 0$ ma interpretację średniego natężenia strumienia zgłaszanych zadań, czyli odwrotności średniej długości przedziału czasu między dwoma sąsiednimi pojawieniami się zadań.

Należy zaznaczyć, że w rozważaniach analitycznych przyjmuje się, że strumień zadań jest stacjonarny, czyli prawdopodobieństwo pojawienia się zadania nie zależy od momentu czasu, w którym ono wystąpiło, pojedynczy, czyli jednocześnie może wystąpić co najwyżej jedno zadanie oraz bez pamięci, czyli nie zależy od liczby i sposobu poprzednich pojawień się zadań.

Regulamin obsługi kolejki

Jest to zbiór reguł decyzyjnych określających kolejność i sposób obsługiwania zadań. Powinien on zapewnić taką obsługę zadań zgłaszanych przez użytkowników, aby czas wykonania zadań spełniał wymagania użytkowników, oraz aby środki techniczne systemu obsługi były wykorzystane możliwie najefektywniej. Podstawowe regulaminy to:

- Regulamin FIFO (ang. *First-In First-Out*). Jest to regulamin naturalny, według którego jako pierwsze jest obsługiwane zadanie najdłużej oczekujące w kolejce.
- Regulamin LIFO (ang. *Last-In First-Out*). Oznacza on obsługę odwrotną do kolejności pojawiania się zadań.
- Regulamin RR (ang. *Round Robin*). Według tego regulaminu obsługa następnego zadania następuje po ustalonym kwancie czasu, w którym było obsługiwane zadanie poprzednie.
- Regulamin RSS (ang. *Random Selection for Service*). Zadanie do obsługi jest określane w drodze losowania.

Oprócz tego jest stosowanych wiele wersji regulaminów z priorytetami, które głównie dotyczą sieci kolejkowych. Umożliwiają wybór dowolnego typu preferencji dla zadań. Mogą na przykład faworyzować zadania krótkie w stosunku do zadań o długich czasach realizacji.

Proces obsługi

Czas obsługi zadania może być również losowy. Jest wtedy opisywany analogicznie jak zadania tworzące strumień zadań, to znaczy przyjmuje się, że jest realizacją zmiennej losowej. Zmienna taka jest scharakteryzowana albo przez funkcję rozkładu prawdopodobieństwa $G(t)$, albo przez funkcję gęstości rozkładu prawdopodobieństwa $g(t)$. Dystrybuanta $G(t)$ ma wtedy interpretację prawdopodobieństwa zdarzenia polegającego na tym, że czas obsługi zadania jest krótszy od wartości t. Często stosowanym rozkładem jest rozkład wykładniczy

$$g(t) = \begin{cases} \mu \exp(-\mu t), & t \geq 0, \\ 0, & t < 0, \end{cases} \tag{1.75}$$

gdzie parametr $\mu > 0$ ma interpretację średniej liczby zadań wykonywanych w jednostce czasu, czyli odwrotności średniego czasu obsługi.

Należy uzupełnić, że oprócz wspomnianych rozkładów prawdopodobieństwa odstępów czasu między pojawieniami się zadań oraz czasu obsługi, a także regulaminów obsługi kolejki, innymi istotnymi wielkościami opisującymi system obsługi są: liczba kanałów obsługi m oraz rozmiar systemu l, czyli maksymalna liczba zadań, które mogą się pomieścić w systemie. Wielkości te są wykorzystywane do opracowywania różnych klasyfikacji systemów obsługi. Jedna z takich klasyfikacji podana przez Lee ma postać

$$\mathrm{X} \mid \mathrm{Y} \mid m \mid \mathrm{d} \mid l ,$$

gdzie X – symbol rozkładu strumienia zgłaszanych zadań,
Y – symbol rozkładu czasu obsługi zadań,
m – liczba kanałów obsługi,
d – kod regulaminu obsługi kolejki,
l – rozmiar systemu.

Do oznaczenia typów rozkładów X oraz Y przyjęto symbole:

M – wykładniczy rozkład odstępów czasu między sąsiednimi pojawieniami się zadań, to znaczy poissonowski rozkład zgłaszających się zadań lub wykładniczy rozkład czasów obsługi. Pojęcie poissonowski rozkład zadań oznacza, że prawdopodobieństwo zdarzenia polegającego na tym, że w przedziale czasu $[0, t)$ w systemie pojawi się k zadań wynosi $p_k(t) = \dfrac{[\lambda(t)]^k}{k!} \exp[-\lambda(t)]$. Jeśli funkcja $\lambda(t) = \lambda t$, to mówimy o jednorodnym, stacjonarnym procesie Poissona,

E_k – rozkład Erlanga *k*-tego rzędu, który może dotyczyć zarówno odstępów czasu między zgłaszającymi się zadaniami, jak i czasu obsługi,
K_n – rozkład χ^2 z *n* stopniami swobody,
C_k – rozkład Coxa rzędu *k,*
D – rozkład regularny (proces pojawiania się zadań jest deterministyczny, stały czas obsługi),
G – rozkład dowolny (brak założeń o procesie pojawiania się zadań, dowolny rozkład czasu obsługi).

Na przykład kod $M \mid E_2 \mid 3 \mid FIFO \mid \infty$ oznacza według tej klasyfikacji system obsługi o poissonowskim strumieniu zadań, erlangowskim rozkładzie czasów obsługi drugiego rzędu, zawierający 3 kanały obsługi pracujące według regulaminu FIFO z nieskończoną liczbą miejsc w systemie.

Jest używana również uproszczona wersja tej klasyfikacji o postaci $X \mid Y \mid m$ sformułowana przez Kendalla.

Analiza systemu $M \mid M \mid 1 \mid FIFO \mid \infty$

Rozważmy teraz jako przykład system o poissonowskim strumieniu zadań, wykładniczym rozkładzie czasu obsługi, jednym stanowisku obsługi oraz kolejce o nieograniczonej pojemności, obsługiwanej według regulaminu FIFO. Aby móc stosować taki opis probabilistyczny, należy założyć, że zadania są generowane przez źródło zadań pojedynczo, wzajemnie niezależnie i niezależnie od przebiegu obsługi. Rozwiązanie problemu analizy umożliwi wyznaczenie m.in. następujących wartości średnich, opisujących system obsługi: czas oczekiwania na obsługę, całkowity czas spędzony w systemie (oczekiwanie w kolejce i obsługa), a także liczby zadań w kolejce i w systemie.

Najistotniejsza z praktycznego punktu widzenia jest analiza systemu obsługi w stanie równowagi statystycznej. Analiza tak zwanych wartości chwilowych nie będzie tu przedstawiana. Niech $X(t)$ oznacza liczbę zadań, które w chwili *t* znajdują się w systemie obsługi. $X(t)$ jest procesem stochastycznym lub inaczej dla ustalonego *t* jego wartość jest realizacją zmiennej losowej. Jeśli $X(t) = 0$, to system jest pusty. Jeśli $X(t) > 1$, to jedno zadanie jest obsługiwane, a pozostałe czekają w kolejce. Niech w ustalonym momencie $t' \geq 0$, $X(t') = j$. Wartość ta może ulec zmianie na $j+1$ w wyniku pojawienia się nowego zadania lub na $j-1$ w wyniku zakończenia obsługi (jeżeli $j > 0$). Rozważmy przedział czasu o długości τ, to znaczy $(t', t' + \tau]$. Dla rozważanego rozkładu wykładniczego prawdopodobieństwo zdarzenia polegającego na wygenerowaniu nowego zadania wynosi $1 - \exp(-\lambda t)$, czyli dla $\tau \to 0$

jest równe $\lambda\tau + o(\tau)$. Symbol o oznacza tu, że $\lim_{\tau\to 0}\frac{1-\exp(-\lambda\tau)-\lambda\tau}{\tau}=0$, czyli $1-\exp(-\lambda\tau)-\lambda\tau = o(\tau)$. Stąd dla $\tau \to 0$ $1-\exp(-\lambda\tau)=\lambda\tau+o(\tau)$. Prawdopodobieństwo zdarzenia przeciwnego, to znaczy, że w danym przedziale nie pojawi się nowe zadanie, wynosi $\exp(-\lambda t)$ lub stosując inny zapis dla $\tau \to 0$, mamy $1-\lambda\tau+o(\tau)$. Wszystkie inne zdarzenia związane z generacją nowych zadań mają prawdopodobieństwo pomijalnie małe, to znaczy $o(\tau)$.

Podobnie można przeanalizować zdarzenia związane ze zmianą liczby zadań w systemie w wyniku ich obsługi. Prawdopodobieństwo zdarzenia polegającego na zakończeniu obsługi zadania w przedziale czasu $(t', t' + \tau]$ wynosi $1-\exp(-\mu t)$, czyli dla $\tau \to 0$ $\mu\tau + o(\tau)$. Prawdopodobieństwo trwania obsługi jest równe $\exp(-\mu\tau)$ lub $1-\mu\tau+o(\tau)$ dla $\tau \to 0$. Ponieważ poszczególne zmienne losowe są wzajemnie niezależne, więc prawdopodobieństwa zdarzeń łącznych obliczamy jako iloczyny odpowiednich prawdopodobieństw. Na przykład prawdopodobieństwo zdarzenia polegającego na wygenerowaniu nowego zadania i ukończeniu obsługi innego zadania wynosi

$$[1-\exp(-\mu\tau)][1-\exp(-\lambda\tau)]=[\mu\tau+o(\tau)][\lambda\tau+o(\tau)]=\mu\lambda\tau^2+o(\tau).$$

Opisywany proces zmiany liczby zadań w systemie obsługi jest tak zwanym dyskretnym procesem Markowa, którego charakterystykami są prawdopodobieństwa przejść zdefiniowane jako

$$p_{j,k}(\tau) \stackrel{\Delta}{=} \mathrm{P}[X(t'+\tau)=k \mid X(t')=j].$$

Opisują one zmiany liczby zadań w systemie z j na k. Prawdopodobieństwa takich zmian w przypadku, gdy j różni się od k o więcej niż 1, są pomijalnie małe. Pozostałe z prawdopodobieństw przejść można wyrazić w postaci następujących równości

$$\begin{aligned}
&p_{0,0}(\tau)=1-\lambda\tau+o(\tau),\\
&p_{0,1}(\tau)=\lambda\tau+o(\tau),\\
&p_{j,j-1}(\tau)=\mu\tau+o(\tau),\\
&p_{j,j}(\tau)=1-(\lambda+\mu)\tau+o(\tau),\\
&p_{j,j+1}(\tau)=\lambda\tau+o(\tau).
\end{aligned} \tag{1.76}$$

Korzystając z równań (1.76), można utworzyć układ równań różniczkowych na prawdopodobieństwa $p_k(t) \stackrel{\Delta}{=} P[X(t)=k]$, to znaczy, że w momencie t w systemie obsługi znajduje się k zadań, a mianowicie

$$\begin{aligned} &\dot{p}_0(t) = -\lambda\, p_0(t) + \mu\, p_1(t), \\ &\dot{p}_k(t) = \lambda\, p_{k-1}(t) - (\lambda+\mu)\, p_k(t) + \mu\, p_{k+1}(t),\ k=1,2,... \end{aligned} \tag{1.77}$$

Układ ten można rozwiązać dla warunku początkowego $p_k(0)$ oraz warunku $\sum_{k=0}^{+\infty} p_k(t) = 1$. Ponieważ, zgodnie z wcześniejszą zapowiedzią, interesują nas własności systemu obsługi w stanie równowagi, tzn. dla $t \to +\infty$, więc szukamy prawdopodobieństw granicznych

$$\lim_{t\to+\infty} p_k(t) = p_k,\ k=0,1,2,...$$

W celu uzyskania takich prawdopodobieństw należy rozwiązać układ równań (1.77) dla lewych stron równych zeru. Po prostych przekształceniach otrzymujemy

$$p_k = p_0 \left(\frac{\lambda}{\mu}\right)^k,\ k=0,1,...$$

Musi być spełniony warunek

$$1 = \sum_{k=0}^{+\infty} p_k = p_0 \sum_{k=0}^{+\infty} \left(\frac{\lambda}{\mu}\right)^k .$$

Szereg ten jest zbieżny wtedy i tylko wtedy, gdy

$$\rho \stackrel{\Delta}{=} \frac{\lambda}{\mu} < 1. \tag{1.78}$$

Wielkość ρ jest tak zwaną intensywnością ruchu i jest ważną charakterystyką systemu obsługi. Z warunku (1.78) wynika następująca własność. Średnia liczba zadań obsługiwanych w jednostce czasu musi być większa niż średnia liczba zadań generowanych w jednostce czasu o tej samej długości, aby długość kolejki nie zdążała do nieskończoności, czyli aby system obsługi osiągnął równowagę. Dla warunku (1.78) otrzymujemy wyrażenie na liczbę zadań w systemie ustabilizowanym

$$p_k = (1-\rho)\,\rho^k,\ k=0,1,...$$

Łatwo można również obliczyć średnią liczbę zadań w systemie jako

$$\sum_{k=1}^{+\infty} k p_k = \rho (1-\rho)^{-1}$$

oraz średnią liczbę zadań w kolejce jako

$$\sum_{k=1}^{+\infty} k p_{k+1} = \rho^2 (1-\rho)^{-1} .$$

Czas oczekiwania zadania w kolejce jest zmienną losową o wartości będącej sumą pełnych czasów obsługi zadań znajdujących się w kolejce oraz resztowego czasu obsługi zadania aktualnie wykonywanego. Wszystkie te czasy, zgodnie z założeniem, są realizacjami zmiennych losowych. Suma takich zmiennych losowych o rozkładach wykładniczych ma rozkład Erlanga, np. [99, 119]. Jej wartość oczekiwana, mająca interpretację średniego czasu przebywania zadania w kolejce, jest równa $\frac{\rho}{\mu(1-\rho)}$. Całkowity czas przebywania zadania w systemie różni się od czasu przebywania w kolejce tylko o czas jego własnej obsługi i również jest realizacją zmiennej losowej o rozkładzie Erlanga. Wartość oczekiwana ma interpretację średniego czasu przebywania zadania w systemie i wynosi $1/[\mu(1 - \rho)]$.

Priorytetowa obsługa zadań

W nawiązaniu do fragmentu dotyczącego regulaminu obsługi kolejki zauważmy, że można wyróżnić dwa typy regulaminów: bezpriorytetowe i priorytetowe. Regulaminy bezpriorytetowe to takie, w których o kolejności obsługi decyduje wyłącznie kolejność pojawiania się zadań. Przykładami są wymienione już wcześniej regulaminy FIFO, LIFO, RR, RSS. W algorytmach priorytetowych o kolejności obsługi decyduje priorytet zadania, który może być ustalony na podstawie porównania pewnych cech systemu obsługi zależnych od typu zadań. Rozważania dotyczące ustalania priorytetów dla zadań zgłaszających się do systemu obsługi ograniczymy do przypadku, gdy zadania te nie są jednorodne. Przyjmiemy mianowicie, że zadania można podzielić na I typów, a procesy zgłoszeń i obsługi są niezależne dla poszczególnych typów zadań. W związku z tym przyjmujemy, że w systemie obsługi występuje I strumieni zgłoszeń i jedno stanowisko obsługi. W ramach jednego strumienia zgłaszają się zadania jednego typu. Określenie strategii obsługi polega w tym przypadku na doborze parametrów dla ustalonego regulaminu obsługi tak, aby optymalizować wybrany wskaźnik jakości działania systemu. Wiąże się to z ustaleniem tak

zwanej liczby poziomów priorytetowych i nadaniem poszczególnym poziomom wartości priorytetowych, np. [49]. Wyznaczanie priorytetów według algorytmu decyzyjnego może się odbywać z wykorzystaniem wielu metod, począwszy od metod symulacyjnych, a skończywszy na metodach analitycznych wykorzystujących modele probabilistyczne. Podejście probabilistyczne polega na wyznaczeniu takich parametrów algorytmu decyzyjnego, które minimalizują probabilistyczny wskaźnik jakości obsługi dla znanych rozkładów prawdopodobieństwa pojawiania się zadań i czasu ich obsługi. Decyzje o przydziale priorytetów podejmuje się na podstawie porównania wybranych, jednakowych cech poszczególnych strumieni zadań. Zespół cech będących podstawą porównania składa się na wielkość, której wartość jest podstawą do określenia ważności każdego strumienia. Wartości wyróżników są więc wskaźnikami ważności strumieni. Porównując te wartości możemy uszeregować strumienie pod względem ich ważności, to znaczy możemy wyznaczyć priorytety. Są one przydzielane kolejno, zgodnie z kierunkiem zmian wartości wyróżników, poszczególnych strumieni. W dalszym ciągu przyjmiemy, że priorytety przyznaje się w kierunku wzrostu wartości wyróżników, to znaczy, że najwyższy priorytet będzie miał strumień o najmniejszej wartości wyróżnika. W przypadku równych wartości wyróżników dla kilku strumieni przydział priorytetów jest niejednoznaczny. Oznaczmy przez $\boldsymbol{e} = [e^{(1)}, e^{(2)}, ..., e^{(I)}]^{\mathrm{T}}$ wektor priorytetów dla wszystkich typów zadań (strumieni). Przyjmuje on wartości w zbiorze utworzonym z wszystkich permutacji ciągu $(1, 2, ..., i, ..., I)$. Wobec losowej natury zjawisk zachodzących w systemach obsługi do oceny efektywności ich działania stosuje się najczęściej probabilistyczne wskaźniki jakości. Ogólnie, wartości takich wskaźników zależą od wektora priorytetów $\boldsymbol{e}$ oraz rozkładów prawdopodobieństw strumieni zadań i procesów obsługi, a najczęściej od parametrów tych rozkładów. Przyjmujemy, że mimo występowania tylko jednego stanowiska obsługi, rozkłady procesów obsługi dla różnych strumieni zadań mogą być różne. Jeśli przyjmiemy, że wśród I strumieni zadań jest M różnych rozkładów prawdopodobieństw charakteryzujących system obsługi, to wektor parametrów tych rozkładów możemy zapisać jako $\boldsymbol{p} = [\boldsymbol{p}^{(1)\,\mathrm{T}}, \boldsymbol{p}^{(2)\,\mathrm{T}}, ..., \boldsymbol{p}^{(m)\,\mathrm{T}}, ..., \boldsymbol{p}^{(M)\,\mathrm{T}}]^{\mathrm{T}}$, gdzie $\boldsymbol{p}^{(m)\,\mathrm{T}}$ – wektor parametrów m-tego rozkładu prawdopodobieństwa i najczęściej $M = 2\,I$. Wówczas ogólna postać wskaźnika jakości Q_{I}, oceniającego efektywność systemu jest funkcją argumentów $\boldsymbol{e}$ i $\boldsymbol{p}$, czyli $Q_{\mathrm{I}} = \Theta(\boldsymbol{e}, \boldsymbol{p})$. Problem wyboru priorytetów polega teraz na wyznaczeniu takiego przydziału priorytetów

$$\Psi^{*}(\boldsymbol{p}) = [\Psi^{*,(1)}(\boldsymbol{p}^{(1)}), \Psi^{*,(2)}(\boldsymbol{p}^{(2)}), ..., \Psi^{*,(i)}(\boldsymbol{p}^{(i)}), ..., \Psi^{*,(I)}(\boldsymbol{p}^{(I)})],$$

gdzie $\boldsymbol{p}^{(i)}$ – podwektor parametrów związanych z i-tym strumieniem, dla którego $\Theta(\boldsymbol{e}, \boldsymbol{p})$ osiąga wartość najmniejszą, to znaczy

$$Q_{\mathrm{I}}^{*}(\boldsymbol{e}^{*}, \boldsymbol{p}) = \min_{\boldsymbol{e}}[\Theta(\boldsymbol{e}, \boldsymbol{p})]. \tag{1.79}$$

Wtedy $Q_{\mathrm{I}}^{*} = \Theta(\boldsymbol{e}^{*}, \boldsymbol{p})$ oraz $\boldsymbol{e}^{*} = \Psi^{*}(\boldsymbol{p})$. Oznaczmy jeszcze przez $w^{(i)}(p^{(i)})$, $i = 1, 2, ..., I$ wartość wyróżnika i-tego strumienia i odpowiedniego procesu obsługi. W dalszym ciągu będziemy rozważać tylko takie przypadki, dla których

$$w^{(1)}(\boldsymbol{p}^{(1)}) \le w^{(2)}(\boldsymbol{p}^{(2)}) \le ... \le w^{(i)}(\boldsymbol{p}^{(i)}) \le ... \le w^{(I)}(\boldsymbol{p}^{(I)}). \tag{1.80}$$

Wzajemne przyporządkowanie wartości wyróżników i priorytetów nie jest w tym przypadku jednoznaczne. Ogólny sposób przyporządkowania jest omówiony w pracy [88]. W przypadku, gdy w zależności (1.80) występują same nierówności ostre, każdy strumień ma inny priorytet oraz

$$e^{(i)} = \Psi^{(i)}(\boldsymbol{p}^{(i)}) = w^{(i)}(\boldsymbol{p}^{(i)}). \tag{1.81}$$

Najczęściej stosowanymi kryteriami oceny efektywności systemów kolejkowych są: opóźnienie realizacji zgłoszonych zadań oraz liczba straconych zadań przed zakończeniem obsługi. W przypadku obsługi priorytetowej, wartości takich wskaźników zależą od wyboru priorytetów. Pierwsze z wymienionych kryteriów dla systemu $\mathrm{M} \mid \mathrm{M} \mid 1 \mid \bullet \mid \infty$ można zapisać następująco

$$Q_{\mathrm{I},1}^{'}(\boldsymbol{e}, \boldsymbol{p}) = \sum_{i=1}^{I} \alpha^{(i)} \lambda^{(i)} \overline{T}_{\mathrm{P}}^{(i)}(\boldsymbol{e}) + \sum_{i=1}^{I} \alpha^{(i)} \mu^{(i)} \overline{T}_{\mathrm{O}}^{(i)}(\boldsymbol{e}), \tag{1.82}$$

gdzie: $\alpha^{(i)}$ – koszt oczekiwania zadania w systemie obsługi przez jednostkę czasu,

$\lambda^{(i)}$ – średnie natężenie i-tego strumienia zadań,

$\mu^{(i)}$ – średnia liczba zadań i-tego strumienia wykonywanych w jednostce czasu,

$\overline{T}_{\mathrm{P}}^{(i)}(\boldsymbol{e})$ – średni czas oczekiwania na obsługę w kolejce związanej z i-tym strumieniem, zależny od wektora priorytetów $\boldsymbol{e}$,

$\overline{T}_{\mathrm{O}}^{(i)}(\boldsymbol{e})$ – średni czas obsługi zadań i-tego strumienia, zależny od wektora priorytetów $\boldsymbol{e}$.

W tym przypadku $\boldsymbol{p}^{(i)} = [\lambda^{(i)}, \mu^{(i)}]^{\mathrm{T}}$ oraz średni czas obsługi nie zależy od priorytetów, dlatego wystarczy rozważać prostszą postać wskaźnika (1.82), a mianowicie

$$Q_{\mathrm{I},1}(\boldsymbol{e}, \boldsymbol{p}) = \sum_{i=1}^{I} \alpha^{(i)} \lambda^{(i)} \overline{T}_{\mathrm{P}}^{(i)}(\boldsymbol{e}), \tag{1.83}$$

gdzie $\boldsymbol{p}^{(i)} = \lambda^{(i)}$. Podamy teraz dwa przykłady optymalnych priorytetów dla systemu $\mathrm{M} \mid \mathrm{M} \mid 1 \mid \bullet \mid \infty$. W obu przypadkach algorytm przydziału priorytetów ψ ma postać (1.81).

Przykład 1.14

Przyjmiemy, że ogólny regulamin obsługi jest następujący. Zgodnie z tym regulaminem, nazywanym też algorytmem z priorytetami bezwzględnymi, do obsługi wybiera się to zadanie, które ma najwyższy priorytet i w swoim strumieniu pojawiło się najwcześniej. W momencie pojawienia się zadania o wyższym priorytecie niż priorytet zadania aktualnie wykonywanego, przerywa się realizację tego zadania i przystępuje się do realizacji zadania o wyższym priorytecie. Po jego zakończeniu kontynuuje się zadanie przerwane od miejsca jego przerwania. Dla tak określonego regulaminu obsługi średni czas oczekiwania na obsługę wynosi

$$\overline{T}_{\mathrm{P}}^{(i)}(\boldsymbol{e}) = \frac{M_I}{(1-\hat{\rho}_{i-1})(1-\hat{\rho}_i)}, \tag{1.84}$$

gdzie

$$M_I = T_{\mathrm{O}}^{(2)} \sum_{i=1}^{I} \lambda^{(i)} \tag{1.85}$$

a $T_{\mathrm{O}}^{(2)}$– moment drugiego rzędu czasu obsługi oraz

$$\hat{\rho}_i = \sum_{l=1}^{i} \lambda^{(l)} T_{\mathrm{O}}^{(l)} = \sum_{l=1}^{i} \rho^{(l)} \tag{1.86}$$

jest sumarycznym obciążeniem systemu obsługą zadań o priorytetach nie niższych niż *i*. Okazuje się [49], że warunkiem koniecznym i wystarczającym optymalnego – w sensie (1.83) ze średnim czasem oczekiwania w postaci (1.84) – przydziału priorytetów jest spełnienie relacji (1.80), gdzie

$$w^{(i)}(\boldsymbol{p}^{(i)}) = \frac{\overline{T}_{\mathrm{O}}^{(i)}}{\alpha^{(i)}}, \quad i = 1, 2, ..., I. \tag{1.87}$$

Oznacza to, że strumień, którego zadaniom przyznano najwyższy priorytet ma najmniejszą wartość wyrażenia (1.87).

■

Przykład 1.15

W tym przykładzie rozważymy priorytetowy regulamin obsługi, w którym zadania są wybierane jak w regulaminie z poprzedniego przykładu, ale z pewną modyfikacją. Polega ona na tym, że zadaniu z *i*-tego strumienia przydziela się odcinek czasu o długości $\Delta\tau_i$, będący dozwolonym czasem jednorazowej obsługi (cyklem obsługi). Jeżeli w tym czasie obsługa zadania nie zakończy się, to jest ona przerywana i rozpoczyna się wykonywanie następnego zadania. Zadanie, którego obsługa została przerwana, jest traktowane jako ostatnie w swoim strumieniu. Po ponownym dojściu do obsługi jego realizacja jest kontynuowana od miejsca przerwania znów przez ten sam odcinek czasu. Postępowanie takie się powtarza do całkowitego zakończenia obsługi zadania. Średni czas oczekiwania wynosi dla wykładniczych rozkładów czasu obsługi

$$\overline{T}_{\mathrm{P}}^{(i)}(\boldsymbol{e}) = \frac{\overline{T}'}{\left(1-\sum_{l=1}^{i}\rho^{(l)}\left[1-\exp\left(-\frac{\Delta\tau_l}{\overline{T}_{\mathrm{O}}^{(l)}}\right)\right]\right)\left(1-\sum_{l=1}^{i-1}\rho^{(l)}\left[1-\exp\left(-\frac{\Delta\tau_l}{\overline{T}_{\mathrm{O}}^{(l)}}\right)\right]\right)}, \quad (1.88)$$

gdzie $\overline{T}'$ jest średnim czasem, jaki upływa od momentu pojawienia się zadania do momentu zakończenia bieżącego cyklu obsługi. W pracy [88] wykazano, że dla przedstawionego regulaminu wskaźnik (1.83) ze średnim czasem oczekiwania w postaci (1.88) osiąga minimum wtedy i tylko wtedy, gdy zachodzi warunek analogiczny do (1.87), to znaczy

$$w^{(i)}(\boldsymbol{p}^{(i)}) = \frac{\overline{T}_{\mathrm{O}}^{(i)}\left[1-\exp\left(-\frac{\Delta\tau_i}{\overline{T}_{\mathrm{O}}^{(i)}}\right)\right]}{\alpha^{(i)}}. \quad (1.89)$$

Tak samo jak w poprzednim przykładzie strumień, którego zadaniom przydzielono najwyższy priorytet ma najmniejszą wartość wyrażenia (1.89).

■

W obu przykładach w celu wyznaczania optymalnych priorytetów należy znać częstotliwości pojawiania się zadań $\lambda^{(i)}$ oraz średni czas obsługi $\overline{T}_{\mathrm{O}}$. W rzeczywistych systemach obsługi informacje takie często są trudne do uzyskania. Dlatego istnieje potrzeba bieżącego ustalania priorytetów na podstawie pomiaru odpowiednich informacji w działającym systemie obsługi. Jedna z metod ustalania optymal-

nych priorytetów, w przypadku braku znajomości parametrów probabilistycznych systemu obsługi, przedstawiona w pracy [88] polegała na oszacowaniu empirycznych parametrów odpowiednich rozkładów prawdopodobieństwa oraz na zastąpieniu otrzymanymi estymatorami nieznanych parametrów. Zaproponowany sposób prowadzi do tak zwanego adaptacyjnego algorytmu obsługi, pozwalającego na iteracyjne wyznaczanie priorytetów.

2. Szeregowanie zadań z uwzględnieniem ruchu realizatorów

2.1. Wprowadzenie

Rozwój problematyki szeregowania zadań jest spowodowany możliwością stosowania bardziej złożonych, a więc efektywniejszych metod i algorytmów rozwiązania. Jest to związane z rozwojem środków informatyki, realizujących te algorytmy. Ponadto, nowa tematyka badawcza jest inspirowana przez nowe możliwości zastosowań praktycznych, między innymi w dyskretnych systemach produkcyjnych, będących ważnym obszarem, w którym istnieje potrzeba stosowania algorytmów szeregowania zadań. W zakresie nowych metod i algorytmów rozwiązania należy wyróżnić zastosowanie różnych technik sztucznej inteligencji, na przykład wybranej logiki rozmytej, algorytmów ewolucyjnych, sztucznych sieci neuronowych. Więcej na ten temat podano w rozdziale 4. Problemy badawcze generowane w związku z nowymi wymaganiami praktycznymi, to przypadki, gdy do wykonywania jednego zadania konieczne jest wykorzystanie więcej niż jednego realizatora, np. [8]. Ważne przypadki dotyczą zagadnień niedeterministycznych. Tematyka badawcza intensywnie rozwijana w tym zakresie dotyczy wykorzystania określonej logiki rozmytej [45]. W konkretnym przypadku zakłada się wówczas, że określone wielkości są zmiennymi lub liczbami rozmytymi.

W niniejszym rozdziale przedstawiono inny nieklasyczny problem szeregowania zadań, który nawiązuje do przypadków występujących w nowoczesnych dyskretnych systemach produkcyjnych jakimi są elastyczne systemy produkcyjne i systemy komputerowo zintegrowanej produkcji. Jest to problem szeregowania zadań z uwzględnieniem ruchu realizatorów. Szersze uzasadnienie tego problemu przedstawiono w następnym rozdziale. Aby wyjaśnić istotę szeregowania z uwzględnieniem ruchu realizatorów (szeregowania z ruchomymi realizatorami), zauważmy, że w przypadku klasycznych problemów szeregowania nie jest istotne miejsce na płaszczyźnie lub w przestrzeni, w którym jest wykonywane zadanie. Wówczas najistotniejszą wielkością charakteryzującą zadanie jest czas jego wykonania. Okazuje

się, że w wielu sytuacjach praktycznych ważna jest informacja o miejscu wykonywania zadania, a dokładniej – informacja o odległościach między miejscami, w których są wykonywane różne zadania. Jest to istotne wtedy, gdy realizatory przemieszczają się w celu wykonania zadań i w konsekwencji czas przemieszczania powinien być uwzględniony w czasie wykonania zadania. W takim przypadku zadanie składa się z dwóch części: dojazdu realizatora do miejsca, w którym ma być wykonana czynność oraz z wykonania tej czynności. W przypadku klasycznym, to znaczy bez uwzględniania ruchu realizatorów, nie ma potrzeby rozróżniania między czynnością a zadaniem, to znaczy oba określenia oznaczają to samo. W rozważanym przypadku zadanie to dojazd i wykonanie czynności. W rezultacie czas wykonania zadania jest sumą czasu dojazdu i wykonania czynności.

Rozważamy pewien przypadek szczególny, który nawiązuje do określonych sytuacji praktycznych. Miejsca, między którymi realizatory się poruszają, nazywamy stanowiskami. Na stanowiskach są umieszczone obiekty, na których realizatory wykonują czynności. Zakładamy, że na jednym stanowisku (obiekcie) jest wykonywana tylko jedna czynność, a więc zbiór stanowisk i zbiór zadań się pokrywają. Ponadto przyjmujemy, że zadania są niezależne i ich wykonania nie można przerywać. Wprowadzamy jeszcze jedno miejsce (stanowisko), z którego wszystkie realizatory wyruszają przed wykonaniem pierwszego przydzielonego zadania, i do którego powracają po zakończeniu wykonywania zadania ostatniego. Na stanowisku tym, zwanym bazą, nie jest wykonywana żadna czynność. Baza uzupełnia zbiór stanowisk, na których są wykonywane czynności, a zbiór zadań jest uzupełniony o zjazdy do bazy, które nie zawierają czynności.

Jeżeli w przypadku klasycznych problemów szeregowania należy określić zbiory lub ciągi zadań do wykonania przez poszczególne realizatory oraz momenty ich rozpoczęcia, to w przypadku szeregowania z ruchomymi realizatorami, dla rozpatrywanego prostego przypadku, rozwiązaniami są trasy przejazdów realizatorów o początkach i końcach w bazie. Potencjalnym obszarem zastosowań rozważanej problematyki może być też wieloagentowy system informatyczny, w którym agent – realizator, będący procedurą (programem komputerowym) powinien dotrzeć do wybranych miejsc w sieci informatycznej w celu wykonania określonej usługi. Mając dany zbiór klientów oczekujących na wykonanie usług oraz zbiór agentów, należy dokonać przyporządkowania elementów jednego zbioru elementom drugiego zbioru tak, aby na przykład czas wykonania wszystkich zadań usługowych był najkrótszy. Wówczas czas transmisji agentów w sieci powinien być brany pod uwagę, a więc należałoby wykorzystać formalizm i algorytm rozwiązujący odpowiedni problem szeregowania z ruchomymi realizatorami.

Opisywany problem szeregowania z ruchomymi realizatorami nawiązuje do klasycznego zagadnienia szeregowania zadań niezależnych i niepodzielnych o równych momentach gotowości na realizatorach dowolnych i jest jego rozszerzeniem. Podstawowa różnica polega na tym, że teraz czasy wykonywania zadań przez realizatory nie są dane a priori, ponieważ przed podaniem rozwiązania, czyli tras przejazdów realizatorów, nie są znane czasy dojazdów. Rozważany problem jest NP-trudnym problemem optymalizacyjnym. Po sformułowaniu odpowiednich problemów optymalizacyjnych dla kryteriów w postaci długości uszeregowania i maksymalnego opóźnienia, w punkcie 2.3 przedstawiono przybliżone algorytmy rozwiązania dla obu kryteriów. Punkt 2.4 poświęcono prezentacji innych ważnych problemów, które zostały opracowane równolegle do zasadniczego zagadnienia, którym było wyznaczanie efektywnych algorytmów przybliżonych, w tym algorytm dokładny dla kryterium w postaci długości uszeregowania. Sformułowany w punkcie 2.2 tak zwany problem wyjściowy jest podstawą do wyznaczania zarówno algorytmów przybliżonych, jak i dokładnych. W obu przypadkach zastosowano odpowiednie dekompozycje problemu wyjściowego. Materiał prezentowany w tym rozdziale jest w większości syntezą problemów przedstawianych w innych pracach [56–64, 66, 69, 73, 75].

2.2. Problem wyjściowy

Rozpocznijmy od wprowadzenia oznaczeń, które będą nawiązywać do tych, które zostały przyjęte w punkcie 1.3. Jak już wspomniano w punkcie 2.1, nie wprowadzamy rozróżnienia między zbiorami zadań i stanowisk. Oba oznaczamy jako $\boldsymbol{H} = \{1, 2, ..., H\}$, gdzie H jest liczbą zadań i stanowisk. Wyróżniamy jedynie bazę i oznaczamy ją jako $h = H + 1$. W konsekwencji zbiór stanowisk, na których są wykonywane czynności, wraz z bazą tworzy zbiór $\overline{\boldsymbol{H}} \stackrel{\Delta}{=} \boldsymbol{H} \cup \{H+1\}$. Analogicznie, $\boldsymbol{R}$ i R są odpowiednio zbiorem realizatorów i liczbą realizatorów. Czas $\tau_{r,h}$ wykonania zadania h przez realizator r jest teraz sumą czasu $\overline{\tau}_{r,h}$ wykonania odpowiedniej czynności na stanowisku h oraz czasu $\hat{\tau}_{r,g,h}$ dojazdu realizatora r do stanowiska h z innego stanowiska oznaczanego jako g, czyli

$$\tau_{r,h} = \overline{\tau}_{r,h} + \hat{\tau}_{r,g,h}, \quad r = 1, 2, ..., R, \quad h = 1, 2, ..., H, \quad g = 1, 2, ..., H+1. \quad (2.1)$$

Ponieważ zakładamy, że po wykonaniu ostatniego przydzielonego zadania każdy realizator musi powrócić do bazy, dodatkowo przez $\hat{\tau}_{r,g,H+1}$, $g = 1, 2, ..., H$ oznaczamy czasy zjazdów realizatorów do bazy. Przyjmujemy też oczywiste założenie, że $\hat{\tau}_{r,h,h} = +\infty$, $h = 1, 2, ..., H+1$. Czasy $\tau_{r,h}$ zdefiniowane w (2.1) tworzą

wektor $\boldsymbol{\tau}_h$ określony przez (1.59). Ponadto możemy zdefiniować macierze $\boldsymbol{\tau} = [\tau_{r,h}]_{\substack{r=1,2,...,R \\ h=1,2,...,H}}$, $\bar{\boldsymbol{\tau}} = [\bar{\tau}_{r,h}]_{\substack{r=1,2,...,R \\ h=1,2,...,H}}$, $\hat{\boldsymbol{\tau}} = [\hat{\tau}_{r,g,h}]_{\substack{r=1,2,...,R \\ g,h=1,2,...,H+1}}$. Problemy wyjściowe będą odrębnie formułowane dla obu rozpatrywanych kryteriów jakości szeregowania.

Długość uszeregowania

Wprowadźmy macierz zmiennych decyzyjnych $\boldsymbol{\gamma}$, która określa przyporządkowanie elementów zbioru $\overline{\boldsymbol{H}}$ do elementów zbioru $\boldsymbol{R}$

$$\boldsymbol{\gamma} = [\gamma_{r,g,h}]_{\substack{r=1,2,...,R \\ g,h=1,2,...,H+1}}, \tag{2.2}$$

gdzie: $\gamma_{r,g,h} = \begin{cases} 1, \text{ jeśli realizator } r \text{ wykonuje zadanie } h \\ \quad \text{po dojeździe ze stanowiska } g, \\ 0, \text{ w przeciwnym przypadku.} \end{cases}$

Nadanie elementowi $\gamma_{r,g,h}$ wartości jeden oznacza, że realizator *r* po zakończeniu wykonywania zadania *g* powinien dojechać do stanowiska *h* i wykonać tam odpowiednią czynność. Postać macierzy $\boldsymbol{\gamma}$ powinna zapewniać wyznaczenie dopuszczalnego uszeregowania w sensie podanym w punkcie 1.3 oraz utworzenie dla każdego realizatora cyklu o początku i końcu w bazie. W celu formalnego przedstawienia tych wymagań wprowadzamy zbiór $\boldsymbol{\Gamma}$ dopuszczanych macierzy $\boldsymbol{\gamma}$, określony przez następujące zależności

$$\gamma_{r,g,h} \in \{0,1\}, \quad r = 1,2,...,R, \quad g,h = 1,2,...,H+1, \tag{2.3}$$

$$\gamma_{r,h,h} = 0, \quad r = 1,2,...,R, \quad h = 1,2,...,H+1, \tag{2.4}$$

$$\sum_{r=1}^{R} \sum_{g=1}^{H+1} \gamma_{r,g,h} = 1, \quad h = 1,2,...,H, \tag{2.5}$$

$$\sum_{g=1}^{H+1} \gamma_{r,g,p} = \sum_{h=1}^{H+1} \gamma_{r,p,h}, \quad r = 1,2,...,R, \quad p = 1,2,...,H+1, \tag{2.6}$$

$$\sum_{h=1}^{H} \gamma_{r,H+1,h} = 1, \quad r = 1,2,...,R, \tag{2.7}$$

$$\boldsymbol{\gamma} \in \boldsymbol{S}, \tag{2.8}$$

gdzie: $\boldsymbol{S} = \left\{ \boldsymbol{\gamma} : \sum_{g \in \overline{\boldsymbol{H}}_S} \sum_{h \in \overline{\boldsymbol{H}}_S} \gamma_{r,g,h} \le \overline{H}_S - 1, \right.$

$\left. \overline{\boldsymbol{H}}_S - \text{dowolny niepusty podzbiór } \overline{\boldsymbol{H}},\ r = 1,2,...,R \right\}$

Definicja 2.1

Zbiór $\boldsymbol{\Gamma} = \{\boldsymbol{\gamma} : \text{zachodzi } (2.3) \wedge (2.4) \wedge (2.5) \wedge (2.6) \wedge (2.7) \wedge (2.8)\}$ nazywamy zbiorem dopuszczalnych macierzy $\boldsymbol{\gamma}$.

■

Oprócz oczywistych wymagań (2.3) i (2.4) pozostałe mają następującą interpretację. Zależność (2.5) zapewnia, że każde zadanie musi być wykonane dokładnie przez jeden realizator po zakończeniu wykonywania innego zadania przez ten realizator lub po dojeździe z bazy. Spełnienie równości (2.6) oznacza, że liczby wjazdów i wyjazdów realizatorów z dowolnego stanowiska są równe. Wówczas jest zachowany warunek ciągłości tras przejazdów realizatorów. Zależności (2.7) i (2.8) zapewniają odpowiednio, że każdy realizator dokładnie jeden raz wyrusza z bazy oraz że trasy dla realizatorów nie tworzą podcykli. Warunkom (2.3)–(2.8) można nadać również interpretację geometryczną, pamiętając, że macierze $\boldsymbol{\gamma}$ są trójwymiarowe. Z zależności (2.5) wynika, że w każdej płaszczyźnie *r*–*g*, tzn. dla ustalonego *h*, musi wystąpić dokładnie jedna wartość 1. Na podstawie analizy zależności (2.6) można określić własności polegające na tym, że w każdej płaszczyźnie *g*–*h*, to znaczy dla ustalonego *r*, liczby elementów o wartościach 1 w kolumnie *p* oraz w wierszu *p* są równe. Następny warunek definiujący $\boldsymbol{\Gamma}$ oznacza, że w płaszczyźnie *r*–*h* dla $g = H + 1$, w każdym wierszu *r*, występuje dokładnie jeden element o wartości 1. Ostatni warunek, zapewniający brak podcykli w trasach realizatorów, można interpretować w ten sposób, że każda podmacierz macierzy $\boldsymbol{\gamma}$ zawiera nie więcej niż $H_{\boldsymbol{S}} - 1$ elementów o wartościach 1, gdzie $H_{\boldsymbol{S}} \le H$.

Macierze decyzyjne $\boldsymbol{\gamma}$ należy wyznaczać tak, aby minimalizować długość uszeregowania $C_{\max}$, oznaczaną dalej jako Q_{C}, w postaci

$$Q_{\mathrm{C}}(\boldsymbol{\gamma}) = \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H+1} \sum_{g=1}^{H+1} \gamma_{r,g,h} (\overline{\tau}_{r,h} + \hat{\tau}_{r,g,h}) \right\}, \tag{2.9}$$

gdzie $\overline{\tau}_{r,H+1} = 0,\ r = 1,2,...,R$, ponieważ w bazie dla realizatorów nie jest wykonywana żadna czynność. W celu zachowania precyzji należy uzupełnić, że w kryterium (2.9) sumowanie względem *g* powinno być wykonywane tylko wtedy, gdy $g \ne h$. Ze względu na swą oczywistość oraz z powodu przyjęcia założenia, że

$\gamma_{r,h,h} = 0$ – w celu uproszczenia zapisu w tym przypadku oraz w analogicznych miejscach w dalszej części pracy warunek $g \neq h$ został pominięty. Sformułowanie problemu szeregowania zadań niezależnych i niepodzielnych o równych momentach gotowości na realizatorach dowolnych, w celu minimalizacji długości uszeregowania jest następujące.

Dla danych: zbioru zadań $\boldsymbol{H}$, zbioru realizatorów $\boldsymbol{R}$, macierzy $\bar{\boldsymbol{\tau}}$ czasów wykonywania czynności na stanowiskach oraz macierzy $\hat{\boldsymbol{\tau}}$ czasów dojazdów realizatorów do stanowisk, należy wyznaczyć wartości elementów macierzy $\boldsymbol{\gamma} \in \boldsymbol{\Gamma}$ tak, aby minimalizować długość uszeregowania (2.9).

Problem ten w dalszym ciągu będzie nazywany problemem wyjściowym PC.

Maksymalne opóźnienie

W przypadku tego kryterium dla każdego zadania *h* jest dany żądany termin zakończenia jego wykonywania d_h. Bez straty ogólności rozważań przyjmijmy, że zadania są uporządkowane w kolejności niemalejących wartości terminów zakończenia, to znaczy

$$d_g \leq d_h, \quad g,h = 1,2,...,H, \quad g < h. \tag{2.10}$$

Wprowadźmy pojęcie przedziału czasu będącego okresem o początku wyznaczonym przez rozpoczęcie procedury szeregowania i końcu określonym przez wartość terminu zakończenia d_l, gdzie l, $l = 1,2,...,H$ jest indeksem przedziału czasu. Macierz zmiennych decyzyjnych $\boldsymbol{\alpha}$ definiujemy następująco

$$\boldsymbol{\alpha} = [\alpha_{r,g,h,l}]_{\substack{r=1,2,...,R \\ g=1,2,...,H+1 \\ h,l=1,2,...,H}} \tag{2.11}$$

gdzie: $\alpha_{r,g,h,l} = \begin{cases} 1, & \text{jeśli wykonanie zadania } h \text{ przez realizator } r \text{ po} \\ & \text{dojeździe ze stanowiska } g \text{ rozpoczyna się przed} \\ & \text{końcem przedziału } l, \\ 0, & \text{w przeciwnym przypadku.} \end{cases}$

Podobnie jak w przypadku minimalizacji długości uszeregowania, postać macierzy $\boldsymbol{\alpha}$ powinna zapewnić określenie dopuszczalnego uszeregowania, czyli ciągłych tras dla realizatorów o początkach w bazie. W tym celu wprowadzamy ograniczenia

$$\alpha_{r,g,h,l} \in \{0,1\}, \quad r = 1,2,...,R, \quad g = 1,2,...,H+1, \quad h,l = 1,2,...,H, \tag{2.12}$$

$$\alpha_{r,h,h,l} = 0, \quad r = 1,2,...,R, \quad h,l = 1,2,...,H, \tag{2.13}$$

$$\sum_{r=1}^{R}\sum_{g=1}^{H+1}\alpha_{r,g,h,l}=1,\quad h=1,2,...,H,\quad l\geq h\,, \tag{2.14}$$

$$\sum_{l=1}^{H}\sum_{g=1}^{H+1}\alpha_{r,g,p,l}=\sum_{l=1}^{H}\sum_{h=1}^{H}\alpha_{r,p,h,l},\quad p=1,2,...,H,\quad p\neq \bar{h}^{r}(H),\ r=1,\,2,\,...,\,R, \tag{2.15}$$

gdzie $\bar{h}^{r}(H)$ jest ostatnim zadaniem wykonywanym przez realizator r,

$$\sum_{l=1}^{H}\sum_{h=1}^{H}\alpha_{r,H+1,h,l}=1,\quad r=1,2,...,R, \tag{2.16}$$

$$\boldsymbol{\alpha}\in\bar{\boldsymbol{S}}\,, \tag{2.17}$$

gdzie: $\bar{\boldsymbol{S}}=\{\boldsymbol{\alpha}:\sum_{l\in\bar{\boldsymbol{H}}_{\bar{S}}}\sum_{g\in\bar{\boldsymbol{H}}_{\bar{S}}}\sum_{h\in\bar{\boldsymbol{H}}_{\bar{S}}}\alpha_{r,g,h,l}\leq\bar{H}_{\bar{S}}-1,$

$\bar{\boldsymbol{H}}_{\bar{S}}$ – dowolny niepusty podzbiór $\bar{\boldsymbol{H}}$, $r=1,2\}$.

Ograniczenia (2.12)–(2.17) mają podobną formę i znaczenie jak ograniczenia (2.3)–(2.8) z tą różnicą, że dodatkowo uwzględniają przedziały czasu, w których są wykonywane zadania. Na przykład zależność (2.14) zapewnia, że każde zadanie będzie wykonane jeden raz i rozpocznie się przed swoim wymaganym terminem zakończenia. Geometryczne interpretacje ograniczeń nie będą tu podawane. Z zależności (2.10) i (2.14) wynika własność

$$\alpha_{r,g,h,l}=1\wedge d_m\geq d_l\Rightarrow\alpha_{r,g,h,m}=1\,. \tag{2.18}$$

Wszystkie ograniczenia nałożone na macierze $\boldsymbol{\alpha}$ determinują zbiór $\boldsymbol{\alpha}$ dopuszczalnych macierzy $\boldsymbol{\alpha}$, określony w definicji 2.2.

Definicja 2.2.

Zbiór $\boldsymbol{\alpha}=[\boldsymbol{\alpha}:\text{zachodzi }\ (2.12)\wedge(2.13)\wedge(2.14)\wedge(2.15)\wedge(2.16)\wedge(2.17)\}$ nazywamy zbiorem dopuszczalnych macierzy $\boldsymbol{\alpha}$.

■

Analityczną postać kryterium jakości szeregowania określamy jako

$$Q_{\mathrm{L}}(\boldsymbol{\alpha})=\max_{r=1,2,...,R}\left\{\max_{l=1,2,...,H}\left\{\sum_{h=1}^{H}\sum_{g=1}^{H+1}\alpha_{r,g,h,l}(\bar{\tau}_{r,h}+\hat{\tau}_{r,g,h})-d_l\right\}\right\}, \tag{2.19}$$

gdzie $Q_{\mathrm{L}}(\boldsymbol{\alpha})$ jest maksymalnym opóźnieniem. Problem wyjściowy dla tego kryterium, oznaczany jako PL, jest następujący.

Dla danych: zbioru zadań $\boldsymbol{H}$, zbioru realizatorów $\boldsymbol{R}$, macierzy $\bar{\tau}$ czasów wykonywania czynności na stanowiskach oraz macierzy $\hat{\tau}$ czasów dojazdów realizatorów do stanowisk, a także terminów zakończenia wykonywania zadań d_h, $h = 1, 2, ..., H$, należy wyznaczyć wartości elementów macierzy $\alpha \in \boldsymbol{\alpha}$ tak, aby minimalizować maksymalne opóźnienie (2.19).

2.3. Algorytmy przybliżone

Własność NP-trudności sformułowanych w punkcie 2.2 problemów wyjściowych PC i PL jest oczywista i formalne dowody nie będą tu przytaczane. Należy jedynie zauważyć, że przyjmując dla każdego h konkretne wartości czasu $\hat{\tau}_{r,g,h}$, to znaczy ustalając g różne niż h – uzyskujemy problem klasyczny, który jest również NP-trudny. Z powodu przytoczonej własności, dla zastosowań praktycznych najistotniejsze jest wyznaczanie algorytmów przybliżonych, czyli takich, których złożoność obliczeniowa jest wielomianowa, a dokładność rozwiązania – oszacowana w sposób analityczny. W kolejnych podpunktach będą zaprezentowane takie algorytmy dla wprowadzonych wcześniej kryteriów w postaci długości uszeregowania i maksymalnego opóźnienia. W obu przypadkach odpowiednie problemy wyjściowe będą przekształcone do prostszej postaci przez zastosowanie dekompozycji. Wykorzystuje ona własność problemów wyjściowych polegającą na tym, że jego rozwiązanie faktycznie polega na określeniu dwóch różnych decyzji, a mianowicie przyporządkowaniu każdemu realizatorowi zbioru zadań do wykonania oraz ustaleniu kolejności ich realizacji. Po dekompozycji obie decyzje są podejmowane oddzielnie. Ulega również zmniejszeniu wymiarowość macierzy decyzyjnych. Podstawową zaletą dekompozycji jest jednak, jak się okaże, możliwość wykorzystania algorytmów rozwiązania znanych problemów. Dla podkreślenia istoty stosowanego przekształcenia, zostało ono nazwane dekompozycją funkcjonalną.

2.3.1. Minimalizacja długości uszeregowania

W punkcie 2.2 podano geometryczną interpretację zależności definiujących zbiór $\boldsymbol{\Gamma}$ dopuszczalnych macierzy decyzyjnych γ. Każda warstwa macierzy γ dla ustalonego r odpowiada innemu realizatorowi, a jej wartości określają ciąg zadań wykonywanych przez ten realizator, a więc również kolejność wykonywania odpowiednich zadań. Procedura dekompozycji problemu PC na prostsze podproblemy optymalizacyjne polega na podziale trójwymiarowych macierzy γ na macierze dwuwymiarowe, będące wspomnianymi warstwami w macierzy γ, odpowiadającymi różnym realizatorom ze zbioru $\boldsymbol{R}$. Idea takiej dekompozycji polega na tym, aby pro-

blem rozwiązywać osobno dla różnych realizatorów. Z prostej analizy zależności określających zbiór $\boldsymbol{\Gamma}$ można jednak wnioskować, że rozważane warstwy macierzy $\boldsymbol{\gamma}$ są ze sobą powiązane i bez straty optymalności nie można wyznaczać osobno (niezależnie) elementów wspomnianych macierzy dwuwymiarowych. Wystarczy bowiem spostrzec, że warstwy macierzy $\boldsymbol{\gamma}$, leżące w płaszczyźnie *g–h* dla różnych *r,* są ze sobą powiązane – wynika to z równości (2.5). Przejdziemy teraz do formalnego przedstawienia procedury dekompozycji. Wprowadźmy nowe macierze decyzyjne wynikające z macierzy $\boldsymbol{\gamma}$. Macierze

$$\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}} = [\tilde{\gamma}^{r}_{g,h}]_{g,h=1,2,\ldots,H+1}, \quad r = 1,2,\ldots,R, \tag{2.20}$$

gdzie: $\tilde{\gamma}^{r}_{g,h} = \begin{cases} 1, & \text{jeśli realizator } r \text{ wykonuje zadanie } h \\ & \text{po dojeździe ze stanowiska } g, \\ 0, & \text{w przeciwnym przypadku.} \end{cases}$

zawierają informacje o trasach przejazdów poszczególnych realizatorów. Mogą one być traktowane jako niezależne warstwy macierzy $\boldsymbol{\gamma}$. Na macierze $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}}$ są nałożone cztery warunki w celu zapewnienia ich dopuszczalności, a mianowicie

$$\tilde{\gamma}^{r}_{h,h} = 0, \quad h = 1,2,\ldots,H+1, \tag{2.21}$$

$$\sum_{g=1}^{H+1} \tilde{\gamma}^{r}_{g,h} \le 1, \quad h = 1,2,\ldots,H+1, \tag{2.22}$$

$$\sum_{g=1}^{H+1} \tilde{\gamma}^{r}_{g,p} = \sum_{h=1}^{H+1} \tilde{\gamma}^{r}_{p,h}, \quad p = 1,2,\ldots,H+1, \tag{2.23}$$

$$\tilde{\boldsymbol{\gamma}}^{r} \in \boldsymbol{S}_{r}, \tag{2.24}$$

gdzie: $\boldsymbol{S}_{r} = \left\{ \tilde{\boldsymbol{\gamma}}^{r} : \sum_{g \in \overline{\boldsymbol{H}}^{r}_{S}} \sum_{h \in \overline{\boldsymbol{H}}^{r}_{S}} \tilde{\gamma}^{r}_{g,h} \le \overline{H}^{r}_{S} - 1 \right\}$ i $\overline{\boldsymbol{H}}^{r}_{S}$ jest dowolnym niepustym pod-zbiorem $\overline{\boldsymbol{H}}$.

Warunki te umożliwiają zdefiniowanie zbioru $\tilde{\boldsymbol{\Gamma}}^{r}$ dopuszczalnych macierzy $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}}$.

Definicja 2.3

Zbiór $\tilde{\boldsymbol{\Gamma}}^{r} = \{\tilde{\boldsymbol{\gamma}}^{r} \in \boldsymbol{\Gamma}^{r} : \text{zachodzi } (2.21) \wedge (2.22) \wedge (2.23) \wedge (2.24)\}$ nazywamy zbiorem dopuszczalnych macierzy $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}}$, gdzie

$$\boldsymbol{\Gamma}^r = \{\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}} = [\tilde{\gamma}^r_{g,h}] : \tilde{\gamma}^r_{g,h} \in \{0,1\},\ g,h = 1,2,...,H+1\}.$$

■

Nierówność (2.22) zapewnia, że każde zadanie może być wykonane co najwyżej jeden raz. Jeśli dwa kolejne warunki są spełnione, to trasy przejazdów realizatorów są ciągłe i nie zawierają podcykli. Macierze $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}} \in \boldsymbol{\Gamma}^r$ dla różnych r nie są niezależne i nie mogą być wyznaczane osobno. Dlatego wprowadzamy nową macierz $\boldsymbol{c}$, która wprawdzie nie wnosi nowych informacji o trasach przejazdów realizatorów, ale koordynuje niezależnie wyznaczane macierze $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}}$. Ma ona postać

$$\boldsymbol{c} = [c_{r,h}]_{\substack{r=1,2,...,R \\ h=1,2,...,H+1}}, \tag{2.25}$$

gdzie: $c_{r,h} = \begin{cases} 1, \text{ jeśli realizator } r \text{ wykonuje zadanie } h, \\ 0, \text{ w przeciwnym przypadku.} \end{cases}$

Zakładamy, że

$$\sum_{r=1}^{R} c_{r,h} = 1,\ \ h = 1,2,...,H \tag{2.26}$$

oraz

$$c_{r,H+1} = 1,\ \ r = 1,2,...,H. \tag{2.27}$$

Oznacza to, że każde zadanie musi być wykonane oraz że każdy realizator musi dojechać do bazy. Wymagania (2.26) i (2.27) umożliwiają zdefiniowanie zbioru $\tilde{\boldsymbol{C}}$ dopuszczalnych macierzy $\boldsymbol{c}$.

Definicja 2.4

Zbiór $\tilde{\boldsymbol{C}} = \{\boldsymbol{c} \in \boldsymbol{C} : \text{zachodzi } (2.26) \wedge (2.27)\}$ nazywamy zbiorem dopuszczalnych macierzy $\boldsymbol{c}$, gdzie

$$\boldsymbol{C} = \{\boldsymbol{c} = [c_{r,h}] : c_{r,h} \in \{0,1\},\ r = 1,2,...,R,\ h = 1,2,...,H+1\}.$$

■

Związek między macierzą $\boldsymbol{\gamma}$ a macierzami $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}}$ i $\boldsymbol{c}$ jest określony zależnością

$$\gamma_{r,g,h} = c_{r,h}\,\tilde{\gamma}^r_{g,h},\ \ g,h = 1,2,...,H+1,\ \ r = 1,2,...,R. \tag{2.28}$$

Na macierze $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{r}}$ wprowadzamy wspólne oznaczenie

$$\tilde{\boldsymbol{\gamma}} \stackrel{\Delta}{=} [\tilde{\boldsymbol{\gamma}}^1 | \tilde{\boldsymbol{\gamma}}^2 | \cdots | \tilde{\boldsymbol{\gamma}}^R]. \tag{2.29}$$

Okazuje się, że nowy problem optymalizacyjny powstały w wyniku zastąpienia trójwymiarowej macierzy decyzyjnej $\boldsymbol{\gamma}$ zestawem dwuwymiarowych macierzy decyzyjnych tworzących $\tilde{\boldsymbol{\gamma}}$ oraz macierzy $\boldsymbol{c}$ jest równoważny problemowi wyjściowemu PC. Jest to treścią twierdzenia 2.1.

Twierdzenie 2.1

Jeżeli związek między macierzami zmiennych decyzyjnych $\boldsymbol{\gamma}$ oraz $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$ jest określony przez (2.28), to problem wyjściowy PC jest równoważny w sensie kryteriów jakości oraz ograniczeń minimalizacji względem $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$ kryterium jakości

$$\tilde{Q}_{\mathrm{C}}(\boldsymbol{c}, \tilde{\boldsymbol{\gamma}}) = \max_{r=1,2,\ldots,R} \left\{ \sum_{h=1}^{H+1} c_{r,h} \left(\bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} \cdot \hat{\tau}_{r,g,h} \right) \right\} \tag{2.30}$$

z ograniczeniami

$$\boldsymbol{c} \in \tilde{\boldsymbol{C}}, \tag{2.31}$$

$$\tilde{\gamma}_{h,h}^{r} = 0, \text{ dla } c_{r,h} = 1, \quad h = 1, 2, \ldots, H+1, \quad r = 1, 2, \ldots, R, \tag{2.32}$$

$$\sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} = 1, \text{ dla } c_{r,h} = 1, \quad h = 1, 2, \ldots, H+1, \quad r = 1, 2, \ldots, R, \tag{2.33}$$

$$\begin{aligned} &\sum_{g=1}^{H+1} \tilde{\gamma}_{g,p}^{r} = \sum_{h=1}^{H+1} \tilde{\gamma}_{p,h}^{r}, \quad p = 1, 2, \ldots, H+1, \\ &\text{dla } c_{r,h} = 1, \quad h = 1, 2, \ldots, H+1, \quad r = 1, 2, \ldots, R, \end{aligned} \tag{2.34}$$

$$\tilde{\boldsymbol{\gamma}}^{r} \in \boldsymbol{S}_r, \quad r = 1, 2, \ldots, R, \tag{2.35}$$

gdzie zbiory $\boldsymbol{S}_r$ są określone w (2.24), przy czym dla występujących tam indeksów g i h jest prawdziwa równość $c_{r,g} = c_{r,h} = 1$.

Dowód

Biorąc pod uwagę zależności (2.28) i (2.33), kryterium jakości dla PC można przekształcić do postaci (2.30)

$$Q_{\mathrm{C}}(\boldsymbol{\gamma}) = \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H+1} \sum_{g=1}^{H+1} \gamma_{r,g,h} \left(\bar{\tau}_{r,h} + \hat{\tau}_{r,g,h} \right) \right\}$$

$$= \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H+1} \sum_{g=1}^{H+1} c_{r,h} \tilde{\gamma}_{g,h}^{r} \left(\bar{\tau}_{r,h} + \hat{\tau}_{r,g,h} \right) \right\}$$

$$= \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H+1} c_{r,h} \left(\bar{\tau}_{r,h} \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} + \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} \hat{\tau}_{r,g,h} \right) \right\}$$

$$= \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H+1} c_{r,h} \left(\bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} \hat{\tau}_{r,g,h} \right) \right\} = \tilde{Q}_{\mathrm{C}}(\boldsymbol{c}, \tilde{\boldsymbol{\gamma}}).$$

Ograniczenia (2.3)–(2.8), które definiują zbiór $\boldsymbol{\Gamma}$, mogą być również przekształcone tak, aby uzyskać (2.31)–(2.35). Zauważmy, że wszystkie macierze decyzyjne są binarne i w obu problemach optymalizacyjnych przejazd między tym samym stanowiskiem nie jest możliwy, to znaczy $\gamma_{r,h,h} = 0$ i $\tilde{\gamma}_{h,h}^{r} = 0$, $r = 1, 2, ..., R$, $h = 1, 2, ..., H+1$. W obu problemach optymalizacyjnych jest spełnione również wymaganie, aby każde zadanie było wykonywane przez dokładnie jeden realizator. W problemie PC jest to zapewnione przez (2.5), natomiast w problemie po dekompozycji funkcjonalnej – przez (2.26). Ponadto (2.5) oraz (2.26) i (2.33) są równoważne. Z zależności (2.5) wynika wprost (2.26) oraz

$$\sum_{r=1}^{R} \sum_{g=1}^{H+1} \gamma_{r,g,h} = \sum_{r=1}^{R} \sum_{g=1}^{H+1} c_{r,h} \tilde{\gamma}_{g,h}^{r} = \sum_{r=1}^{R} c_{r,h} \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} = \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} = 1.$$

Warunki (2.6) i (2.34) zapewniają, że liczby wjazdów i wyjazdów realizatora z dowolnego stanowiska są równe. Równoważność obu warunków można wykazać, kładąc w (2.6) dla $r = 1, 2, ..., R$ oraz $p = 1, 2, ..., H+1$ iloczyny $c_{r,p} \tilde{\gamma}_{g,p}^{r}$ w miejsce $\gamma_{r,p,h}$ oraz wykorzystując fakt, że w takim przypadku $c_{r,p} = c_{r,h} = 1$. Po wstawieniu (2.28) do (2.6) dla $p = H + 1$ otrzymujemy

$$\sum_{g=1}^{H+1} c_{r,H+1} \tilde{\gamma}_{g,H+1}^{r} = \sum_{h=1}^{H+1} c_{r,h} \tilde{\gamma}_{H+1,h}^{r}, \quad r = 1, 2, ..., R. \tag{2.36}$$

Po uwzględnieniu (2.27) i w konsekwencji (2.33) dla $h = H + 1$ stwierdzamy, że lewa strona zależności (2.36) jest równa 1 i może być zapisana jako

$$\sum_{h=1}^{H+1} \tilde{\gamma}_{H+1,h}^{r} = 1 \text{ dla } c_{r,h} = 1, \quad r = 1, 2, ..., R.$$

Oznacza to, że każdy realizator dokładnie jeden raz wyrusza z bazy, a więc ograniczenia dla problemu po dekompozycji funkcjonalnej zapewniają spełnienie wymagania, które w problemie PC zostało wyrażone w postaci równości (2.7). Zbiór $\boldsymbol{S}$ z zależności (2.8) przekształcamy zgodnie z (2.28) i otrzymujemy

$$\boldsymbol{S} = \left\{ \left[c_{r,h}\, \tilde{\gamma}_{g,h}^{r} \right] : \sum_{h \in \overline{\boldsymbol{H}}_S} c_{r,h} \sum_{g \in \overline{\boldsymbol{H}}_S} \tilde{\gamma}_{g,h}^{r} \le \overline{H}_S - 1, \quad r = 1, 2, ..., R \right\},$$

co równoważnie można zapisać w postaci

$$\boldsymbol{S}_r = \left\{ \left[\tilde{\gamma}_{g,h}^{r} \right] : \sum_{g \in \overline{\boldsymbol{H}}_S^r} \sum_{h \in \overline{\boldsymbol{H}}_S^r} \tilde{\gamma}_{g,h}^{r} \le \overline{H}_S^r - 1 \right\}, \quad r = 1, 2, ..., R,$$

gdzie $\overline{\boldsymbol{H}}_S^r$ jest dowolnym niepustym podzbiorem $\overline{\boldsymbol{H}}$, dla którego elementów zachodzi $c_{r,h} = 1$. Tak więc ograniczenia w obu rozważanych problemach optymalizacyjnych zawierają wymaganie, aby trasa żadnego realizatora nie zawierała podcykli. Ostatecznie wykazaliśmy równoważność kryteriów jakości oraz wymagań zawartych w ograniczeniach.

■

Z prostej analizy ograniczeń (2.32), (2.33) i (2.34), (2.35) wynika, że macierze decyzyjne $\boldsymbol{c}$ i $\tilde{\gamma}$ są ze sobą powiązane. Oznacza to m.in., że interesują nas tylko takie macierze $\tilde{\boldsymbol{\gamma}}^r$, które odpowiadają zadaniom wykonywanym przez realizator r dla $r = 1, 2, ..., R$, czyli tym zadaniom, dla których $c_{r,h} = 1$.

Rozpatrzymy teraz to zagadnienie bardziej szczegółowo. Zmodyfikujemy wpierw definicję 2.3 zbioru $\tilde{\boldsymbol{\Gamma}}^r$. W tym celu wprowadzamy oznaczenie $\overline{\boldsymbol{H}}^r(\boldsymbol{c})$ na zbiór zadań wykonywanych przez realizator r, $r = 1, 2, ..., R$, to znaczy

$$\overline{\boldsymbol{H}}^r(c) \triangleq \left\{ h \in \overline{\boldsymbol{H}} : c_{r,h} = 1 \right\}, \quad r = 1, 2, ..., R. \tag{2.37}$$

Ponadto $\overline{H}^r(\boldsymbol{c})$ oznacza liczność zbioru $\overline{\boldsymbol{H}}^r(\boldsymbol{c})$. Wtedy możemy wykorzystać warunki (2.21)–(2.24) definiujące zbiór $\tilde{\boldsymbol{\Gamma}}^r$ i otrzymać:

$$\tilde{\boldsymbol{\gamma}}_{h,h}^{r} = 0, \quad h \in \overline{\boldsymbol{H}}^r(\boldsymbol{c}), \quad r = 1, 2, ..., R, \tag{2.38}$$

$$\sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} = 1,\ h \in \overline{\boldsymbol{H}}^{r}(\boldsymbol{c}),\ r = 1,2,...,R, \tag{2.39}$$

$$\sum_{g=1}^{H+1} \tilde{\gamma}_{g,p}^{r} = \sum_{h=1}^{H+1} \tilde{\gamma}_{p,h}^{r},\ \ p \in \overline{\boldsymbol{H}}^{r}(\boldsymbol{c}),\ r = 1,2,...,R, \tag{2.40}$$

$$[\tilde{\gamma}_{g,h}^{r}] \in \boldsymbol{S}_{r},\ \ g,h \in \overline{\boldsymbol{H}}^{r}(\boldsymbol{c}),\ \ r = 1,2,...,R \tag{2.41}$$

oraz zdefiniować zbiór $\tilde{\Gamma}(\boldsymbol{c})$ dopuszczalnych macierzy $\tilde{\boldsymbol{\gamma}}^{r}$ zależny od macierzy $\boldsymbol{c}$, $r = 1,2,...,R$.

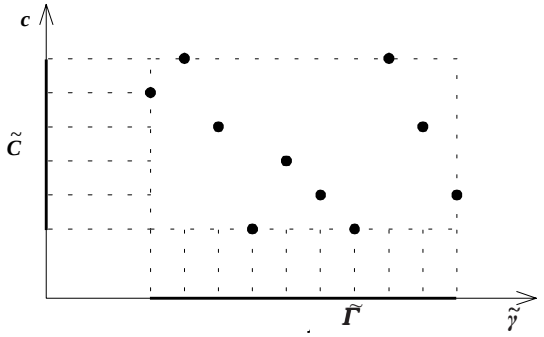
Definicja 2.5

Zbiór $\tilde{\Gamma}^{r}(\boldsymbol{c}) = \{\tilde{\boldsymbol{\gamma}}^{r} \in \Gamma^{r} : \text{zachodzi}\,(2.38) \wedge (2.39) \wedge (2.40) \wedge (2.41)\}$ nazywamy zbiorem dopuszczalnych macierzy $\tilde{\boldsymbol{\gamma}}^{r}$, zależnym od macierzy $\boldsymbol{c}$.

Związek między $\boldsymbol{c}$ oraz $\tilde{\boldsymbol{\gamma}}$ może być wyrażony w formie następującej relacji $\boldsymbol{W}_1$

$$\begin{aligned} &\boldsymbol{W}_1(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}^{1},\tilde{\boldsymbol{\gamma}}^{2},...,\tilde{\boldsymbol{\gamma}}^{R}) \ \stackrel{\Delta}{=} \boldsymbol{W}_1(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}) \\ &= \{(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}^{1},\tilde{\boldsymbol{\gamma}}^{2},...,\tilde{\boldsymbol{\gamma}}^{R}) \in \boldsymbol{C} \times \boldsymbol{\Gamma}^{1} \times \boldsymbol{\Gamma}^{2} \times \cdots \times \boldsymbol{\Gamma}^{R} \stackrel{\Delta}{=} (\boldsymbol{c},\tilde{\boldsymbol{\gamma}}) \in \boldsymbol{C} \times \boldsymbol{\Gamma} : \\ &\text{zachodzi}\,(2.26) \wedge (2.27) \wedge (2.38) \wedge (2.39) \wedge (2.40) \wedge (2.41)\}. \end{aligned} \tag{2.42}$$

Ilustracją tej zależności jest rysunek 2.1, dla $\tilde{\boldsymbol{\Gamma}} \subseteq \boldsymbol{\Gamma}$, gdzie $\tilde{\Gamma} = \tilde{\Gamma}^{1}(\boldsymbol{c}) \times \tilde{\Gamma}^{2}(\boldsymbol{c}) \times ... \times \tilde{\Gamma}^{R}(\boldsymbol{c})$. Znajomość macierzy $\tilde{\boldsymbol{\gamma}}$ w sposób jednoznaczny określa $\boldsymbol{c}$, ale nie odwrotnie. Jest to zrozumiałe, ponieważ elementy macierzy $\tilde{\boldsymbol{\gamma}}$ determinują trasy przejazdów realizatorów, a więc również przyporządkowanie realizatorów do zadań i w konsekwencji wartości elementów macierzy $\boldsymbol{c}$. Ze znajomości przyporządkowania realizatorów do zadań nie wynikają natomiast w sposób jednoznaczny ich trasy



Rys. 2.1. Ilustracja zależności między $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$ dla $\tilde{\boldsymbol{\gamma}} \in \tilde{\boldsymbol{\Gamma}}$

przejazdów. W podsumowaniu stwierdzamy, że zmienne decyzyjne – macierze $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$ są ze sobą powiązane i może to być wyrażone w postaci relacji (2.42) oraz że powiązanie to jest niesymetryczne, to znaczy ze znajomości macierzy $\tilde{\boldsymbol{\gamma}}$ wynika znajomość macierzy $\boldsymbol{c}$ – ale nie odwrotnie.

Możemy teraz ostatecznie sformułować problem szeregowania z ruchomymi realizatorami po dekompozycji funkcjonalnej.

Dane są takie same jak w problemie wyjściowym PC, podanym w punkcie 2.2. Należy wyznaczyć wartości elementów macierzy $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$, należących odpowiednio do zbioru $\tilde{\boldsymbol{C}}$ i do zbioru $\tilde{\boldsymbol{\Gamma}}(\boldsymbol{c})$ tak, aby minimalizować długość uszeregowania (2.30).

Istnienie powiązania między zmiennymi decyzyjnymi $\boldsymbol{c}$ oraz $\tilde{\boldsymbol{\gamma}}^1,\tilde{\boldsymbol{\gamma}}^2,...,\tilde{\boldsymbol{\gamma}}^R$ w formie relacji uniemożliwia zmniejszenie wymiarowości problemu optymalizacyjnego uzyskane w wyniku dekompozycji, a polegające na oddzielnym wyznaczaniu macierzy $\boldsymbol{c}$ oraz $\tilde{\boldsymbol{\gamma}}^1,\tilde{\boldsymbol{\gamma}}^2,...,\tilde{\boldsymbol{\gamma}}^R$. Dzieje się tak, ponieważ w celu wyznaczenia macierzy $\boldsymbol{c}$ są potrzebne pewne informacje zawarte w $\tilde{\boldsymbol{\gamma}}^r$, r = 1, 2, ..., R. Nie jest konieczna znajomość $\tilde{\boldsymbol{\gamma}}^r$, ponieważ $\tilde{\boldsymbol{\gamma}}^r$ jednoznacznie określa $\boldsymbol{c}$, ale jest potrzebna informacja o stanowisku, z którego realizator podąża w celu wykonania bieżącego zadania. Traktując sprawę formalnie, mówimy o takiej sytuacji, w której spośród wszystkich $\tilde{\gamma}^r_{g,h}$ dla $g = 1,2,...,H+1$ dokładnie jeden element ma wartość równą 1. Własność taką mają macierze należące do określonego w definicji 2.6 zbioru $\hat{\boldsymbol{\Gamma}}^r$ początkowych macierzy $\tilde{\boldsymbol{\gamma}}^r$, r = 1, 2, ..., R.

Definicja 2.6

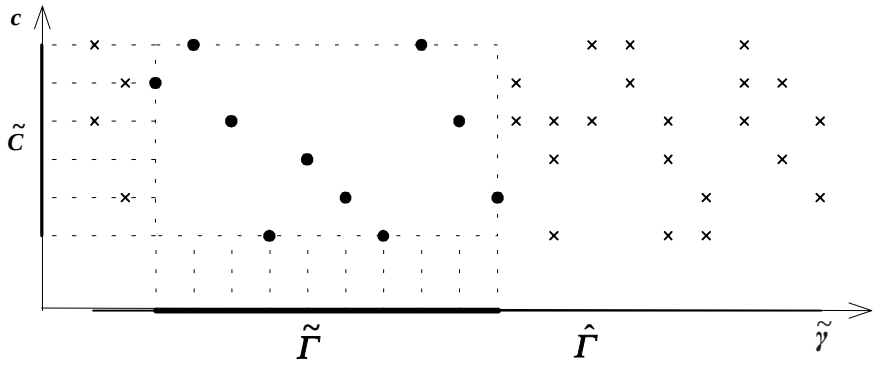
Zbiór $\hat{\boldsymbol{\Gamma}}^r = \left\{\tilde{\boldsymbol{\gamma}}^r \in \boldsymbol{\Gamma}^r : \tilde{\gamma}^r_{h,h} = 0,\ h = 1,2,...,H+1 \text{ oraz } \sum_{g=1}^{H+1} \tilde{\gamma}^r_{g,h} = 1,\ h = 1,2,...,H+1\right\}$

nazywamy zbiorem początkowych postaci macierzy $\tilde{\boldsymbol{\gamma}}^r$.

■

Proponowany algorytm rozwiązania polega na wstępnym ustaleniu początkowych wartości macierzy $\tilde{\boldsymbol{\gamma}}^r$ ze zbiorów $\hat{\boldsymbol{\Gamma}}^r$, a następnie na kolejnym wyznaczaniu $\boldsymbol{c}$ oraz $\tilde{\boldsymbol{\gamma}}^r$. Macierz $\boldsymbol{c}$ oraz macierz $\tilde{\boldsymbol{\gamma}}^r$ należąca do $\hat{\boldsymbol{\Gamma}}^r$ też są zależne, ale inaczej niż w przypadku wyrażonym przez relację (2.42). W celu określenia rozpatrywanej współzależności macierzy wprowadzimy inną relację, a mianowicie

$$\boldsymbol{W}_2(c,\tilde{\boldsymbol{\gamma}}) = \{(c,\boldsymbol{\gamma}) \in \boldsymbol{C}\times\boldsymbol{\Gamma} : \text{zachodzi } (2.5) \wedge (2.6) \wedge \tilde{\gamma}^r_{h,h} = 0,$$
$$h = 1,2,...,H+1 \wedge \sum_{g=1}^{H+1} \tilde{\gamma}^r_{g,h} = 1,\ \ h = 1,2,...,H+1\}. \qquad (2.43)$$



Rys. 2.2. Ilustracja zależności między $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$ dla $\tilde{\boldsymbol{\gamma}} \in \tilde{\boldsymbol{\Gamma}}$ i $\tilde{\boldsymbol{\gamma}} \in \hat{\boldsymbol{\Gamma}}$

W wyniku porównania (2.42) i (2.43) stwierdzamy, że $\boldsymbol{W}_1(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}) \subseteq \boldsymbol{W}_2(\boldsymbol{c},\tilde{\boldsymbol{\gamma}})$ oraz że znajomość $\tilde{\boldsymbol{\gamma}}^r \in \hat{\boldsymbol{\Gamma}}^r$ nie określa jednoznacznie $\boldsymbol{c}$. Rysunek 2.2 jest poglądową interpretacją tych własności. Przedstawione rozważania wykorzystujemy w prezentowanej metodzie rozwiązania. Składa się ona z trzech następujących kroków.

1. Ustalamy $\tilde{\boldsymbol{\gamma}}^r = \tilde{\boldsymbol{\gamma}}_0^r \in \hat{\boldsymbol{\Gamma}}^r$, $r = 1,2,...,R$, ogólnie $\tilde{\boldsymbol{\gamma}} = \tilde{\boldsymbol{\gamma}}_0 \in \hat{\Gamma} \stackrel{\Delta}{=} \hat{\Gamma}^1 \times \hat{\Gamma}^2 \times ... \hat{\Gamma}^R$.

2. Minimalizujemy $\tilde{Q}_{\mathrm{C}}(\boldsymbol{c};\tilde{\boldsymbol{\gamma}})$ względem $\boldsymbol{c} \in \tilde{\boldsymbol{C}}$. Wtedy oczywiście $(\boldsymbol{c},\tilde{\boldsymbol{\gamma}} = \tilde{\boldsymbol{\gamma}}_0) \in \boldsymbol{W}_2(\boldsymbol{c},\tilde{\boldsymbol{\gamma}})$. W wyniku tej minimalizacji otrzymujemy postać optymalną macierzy $\boldsymbol{c}$ zależną od $\tilde{\boldsymbol{\gamma}}_0$, czyli $\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0)$, oraz kryterium (2.30) również zależne od $\tilde{\boldsymbol{\gamma}}_0$, tj. $\tilde{Q}_{\mathrm{C}}(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0),\tilde{\boldsymbol{\gamma}}) \stackrel{\Delta}{=} \overline{Q}_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}};\tilde{\boldsymbol{\gamma}}_0)$.

3. Minimalizujemy $\overline{Q}_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}};\tilde{\boldsymbol{\gamma}}_0)$ względem $\tilde{\boldsymbol{\gamma}} \in \tilde{\Gamma}(\boldsymbol{c}*(\tilde{\boldsymbol{\gamma}}_0))$ i otrzymujemy $\tilde{\boldsymbol{\gamma}}^*(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0)) \stackrel{\Delta}{=} \tilde{\boldsymbol{\gamma}}^*(\tilde{\boldsymbol{\gamma}}_0)$ oraz $\overline{Q}_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}}^*(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0));\tilde{\boldsymbol{\gamma}}_0) \stackrel{\Delta}{=} \tilde{Q}_{\mathrm{C}}{}^*(\tilde{\boldsymbol{\gamma}}_0)$.

Tak więc, ostatecznym rezultatem działania metody są optymalne postaci macierzy $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$, tj. odpowiednio $\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0)$ i $\tilde{\boldsymbol{\gamma}}*(\tilde{\boldsymbol{\gamma}}_0) = [\tilde{\boldsymbol{\gamma}}^{1,*}(\tilde{\boldsymbol{\gamma}}_0) | \tilde{\boldsymbol{\gamma}}^{2,*}(\tilde{\boldsymbol{\gamma}}_0) | ... | \tilde{\boldsymbol{\gamma}}^{R,*}(\tilde{\boldsymbol{\gamma}}_0)]$ oraz optymalna wartość kryterium jakości $\tilde{Q}_{\mathrm{C}}(\boldsymbol{c},\tilde{\boldsymbol{\gamma}})$, czyli $\tilde{Q}_{\mathrm{C}}^*(\tilde{\boldsymbol{\gamma}}_0)$. Otrzymane wyniki zależą od początkowej postaci macierzy $\tilde{\boldsymbol{\gamma}}$, czyli od ustalonego $\tilde{\boldsymbol{\gamma}}_0$, co zostało zaznaczone w punktach 1–3 przez podanie $\tilde{\boldsymbol{\gamma}}_0$ jako argumentu w poszczególnych wyrażeniach. Zapis ten należy rozumieć w ten sposób, że dla różnych $\tilde{\boldsymbol{\gamma}}_0$ możemy otrzymać różne wyniki liczbowe – nie jest to jednak zależność funkcyjna.

Omówimy teraz bardziej dokładnie problemy minimalizacji z kroków 2. i 3. Do rozwiązania problemu optymalizacyjnego w kroku 2. zastosujemy metodę rozwiązywania problemu szeregowania zadań niepodzielnych i niezależnych z równymi momentami gotowości na realizatorach dowolnych. Wobec ustalonej w kroku 1.

postaci macierzy $\tilde{\gamma} = \tilde{\gamma}_0 \in \hat{\Gamma}$ problem minimalizacji $\tilde{Q}_{\mathrm{C}}(\boldsymbol{c};\tilde{\gamma})$ względem $\boldsymbol{c} \in \tilde{\boldsymbol{C}}$ zapisujemy w postaci

$$\min_{\boldsymbol{c}} \tilde{Q}_{\mathrm{C}}(\boldsymbol{c};\tilde{\gamma}) = \min_{\boldsymbol{c}} \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H+1} c_{r,h} \left(\bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{0,g,h}^{r} \hat{\tau}_{r,g,h} \right) \right\} = \overline{Q}_{\mathrm{C}}(\tilde{\gamma};\tilde{\gamma}_0) \quad (2.44)$$

z ograniczeniami

$$c_{r,h} \in \{0,1\}, \quad r = 1,2,...,R, \quad h = 1,2,...,H, \quad (2.45)$$

$$c_{r,H+1} = 1, \quad r = 1,2,...,R, \quad (2.46)$$

$$\sum_{r=1}^{R} c_{r,h} = 1, \quad h = 1,2,...,H. \quad (2.47)$$

Zgodnie z ograniczeniem (2.46) zadania zjazdu do bazy nie podlegają szeregowaniu – każdy realizator musi jeden raz zjechać do bazy. Dlatego modyfikujemy (2.44) i dalej rozwiązujemy następujący problem minimalizacji

$$\min_{\boldsymbol{c}} \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H} c_{r,h} \left(\bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{0,g,h}^{r} \hat{\tau}_{r,g,h} \right) \right\} = \min_{\boldsymbol{c}} \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H} c_{r,h} T_{r,h}(\tilde{\gamma}_0^r) \right\}, \quad (2.48)$$

gdzie $T_{r,h}(\tilde{\gamma}_0^r) = \bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{0,g,h}^{r} \hat{\tau}_{r,g,h}$, $r = 1,2,...,R$, $h = 1,2,...,H$ jest stałą wartością zależną od $\tilde{\gamma}_0^r$, z ograniczeniami (2.45) i (2.47). Wyrażenie (2.48) z ograniczeniami (2.45) i (2.47) jest formalnym zapisem szeregowania *H* zadań niepodzielnych i niezależnych z równymi momentami gotowości na *R* realizatorach dowolnych. Wynikiem rozwiązania są: macierz $\boldsymbol{c}^*(\tilde{\gamma}_0)$ oraz postać $\overline{Q}_{\mathrm{C}}(\tilde{\gamma};\tilde{\gamma}_0)$ kryterium (2.30), którą zgodnie z (2.44) przedstawiono w następujący sposób

$$\overline{Q}_{\mathrm{C}}(\tilde{\gamma};\tilde{\gamma}_0) = \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H} c_{r,h}^{*}(\tilde{\gamma}_0) \left(\bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} \hat{\tau}_{r,g,h} \right) + \sum_{g=1}^{H+1} \tilde{\gamma}_{g,H+1}^{r} \hat{\tau}_{r,g,H+1} \right\}. \quad (2.49)$$

Otrzymana macierz $\boldsymbol{c}^*(\tilde{\gamma}_0)$ określa jednoznacznie zbiory zadań, przeznaczone do wykonywania przez realizatory. W określeniu takich zbiorów uwzględniamy zadania zjazdów realizatorów do bazy, a mianowicie

$$\overline{\boldsymbol{H}}^{r,*}(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0)) \stackrel{\Delta}{=} \overline{\boldsymbol{H}}_c^{r,*} = \left\{h \in \boldsymbol{H} : c_{r,h}^*(\tilde{\boldsymbol{\gamma}}_0) = 1\right\} \cup \{H+1\}, \quad r = 1,2,...,R. \quad (2.50)$$

Wtedy kontynuujemy (2.49) jako

$$\overline{Q}_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}};\tilde{\boldsymbol{\gamma}}_0) = \max_{r=1,2,...,R}\left\{\sum_{h=1}^{\overline{H}_c^{r,*}}\left(\bar{\tau}_{r,h} + \sum_{g=1}^{\overline{H}_c^{r,*}}\tilde{\gamma}_{g,h}^r\hat{\tau}_{r,g,h}\right)\right\}, \quad (2.51)$$

gdzie $\overline{H}_c^{r,*}$ jest licznością zbioru $\overline{\boldsymbol{H}}_c^{r,*}$.

Wyrażenie (2.51) jest podstawą minimalizacji w kroku 3. prezentowanego algorytmu. Jak łatwo zauważyć, występujące w (2.51) argumenty funkcji maksimum dla różnych r są niezależne. Taką samą własność mają zbiory $\tilde{\boldsymbol{\Gamma}}^r(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0))$. Dlatego możemy osobno wyznaczać optymalne wartości $\tilde{\boldsymbol{\gamma}}^r(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0)) \stackrel{\Delta}{=} \tilde{\boldsymbol{\gamma}}^r(\tilde{\boldsymbol{\gamma}}_0)$ (dalej oznaczane w skrócie jako $\tilde{\boldsymbol{\gamma}}^r$) przez rozwiązywanie problemów minimalizacji kryteriów cząstkowych

$$\overline{Q}_{\mathrm{C}}^r(\tilde{\boldsymbol{\gamma}}^r;\tilde{\boldsymbol{\gamma}}_0) \stackrel{\Delta}{=} \sum_{h=1}^{\overline{H}_c^{r,*}}\left(\bar{\tau}_{r,h} + \sum_{g=1}^{\overline{H}_c^{r,*}}\tilde{\gamma}_{g,h}^r\hat{\tau}_{r,g,h}\right) \quad (2.52)$$

względem $\tilde{\boldsymbol{\gamma}}^r \in \tilde{\boldsymbol{\Gamma}}^r(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0))$, $r = 1,2,...,R$, a następnie wyznaczać optymalną wartość kryterium dla wszystkich realizatorów jako wartość maksymalnego kryterium cząstkowego. Formalnie zastosowaną dekompozycję problemu optymalizacyjnego w kroku 3 możemy przedstawić następująco

$$\min_{\tilde{\boldsymbol{\gamma}}^r \in \tilde{\boldsymbol{\Gamma}}^r,\ r=1,2,...,R} \overline{Q}_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}};\tilde{\boldsymbol{\gamma}}_0) = \min_{\tilde{\boldsymbol{\gamma}}^r \in \tilde{\boldsymbol{\Gamma}}^r,\ r=1,2,...,R} \max_{r=1,2,...,R}\left\{\overline{Q}_{\mathrm{C}}^r(\tilde{\boldsymbol{\gamma}}^r;\tilde{\boldsymbol{\gamma}}_0)\right\}$$

$$= \max_{r=1,2,...,R}\left\{\min_{\tilde{\boldsymbol{\gamma}}^r \in \tilde{\boldsymbol{\Gamma}}^r} \overline{Q}_{\mathrm{C}}^r(\tilde{\boldsymbol{\gamma}}^r;\tilde{\boldsymbol{\gamma}}_0)\right\} = \max_{r=1,2,...,R}\left\{\tilde{Q}_{\mathrm{C}}^{r,*}(\tilde{\boldsymbol{\gamma}}_0)\right\} = \tilde{Q}_{\mathrm{C}}^*(\tilde{\boldsymbol{\gamma}}_0). \quad (2.53)$$

W wyrażeniu (2.53) zbiory $\tilde{\boldsymbol{\Gamma}}^r(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0))$ w skrócie zapisywaliśmy jako $\tilde{\boldsymbol{\Gamma}}^r$. Wskaźnik jakości (2.52) możemy równoważnie przedstawić w postaci

$$\overline{Q}_{\mathrm{C}}^r(\tilde{\boldsymbol{\gamma}}^r;\tilde{\boldsymbol{\gamma}}_0) = \sum_{h=1}^{\overline{H}_c^{r,*}}\bar{\tau}_{r,h} + \sum_{h=1}^{\overline{H}_c^{r,*}}\sum_{g=1}^{\overline{H}_c^{r,*}}\tilde{\gamma}_{g,h}^r\hat{\tau}_{r,g,h}. \quad (2.54)$$

Pierwszy składnik w zależności (2.54) jest nieistotny w minimalizacji względem $\tilde{\boldsymbol{\gamma}}^r$, natomiast minimalizacja drugiego składnika z ograniczeniami od (2.38) do (2.41), które określają zbiór $\tilde{\boldsymbol{\Gamma}}^r(\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0))$, jest formalnym zapisem problemu komi-

wojażera. W ten sposób problem optymalizacyjny z kroku 3. algorytmu sprowadziliśmy do R problemów komiwojażera ze wspólną bazą. Metodę rozwiązania problemu po dekompozycji funkcjonalnej możemy jeszcze raz skrótowo przedstawić w postaci następującej zależności

$$Q_{\mathrm{C}}^{*} = \tilde{Q}_{\mathrm{C}}^{*}(\boldsymbol{c}^{*}, \tilde{\boldsymbol{\gamma}}^{*}) = \min_{\substack{\boldsymbol{c}\in\tilde{\boldsymbol{C}} \\ \tilde{\boldsymbol{\gamma}}\in\tilde{\boldsymbol{\Gamma}}}} [\tilde{Q}_{\mathrm{C}}(\boldsymbol{c}, \tilde{\boldsymbol{\gamma}})] \le \min_{\tilde{\boldsymbol{\gamma}}\in\tilde{\boldsymbol{\Gamma}}} \min_{\substack{\boldsymbol{c}\in\tilde{\boldsymbol{C}} \\ \tilde{\boldsymbol{\gamma}}=\tilde{\boldsymbol{\gamma}}_0\in\hat{\boldsymbol{\Gamma}}}} \tilde{Q}_{\mathrm{C}}(\boldsymbol{c}; \tilde{\boldsymbol{\gamma}})$$

$$\le \min_{\tilde{\boldsymbol{\gamma}}\in\tilde{\boldsymbol{\Gamma}}(\boldsymbol{c}^{*}(\tilde{\boldsymbol{\gamma}}_0))} [\overline{Q}_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}}; \tilde{\boldsymbol{\gamma}}_0)] = \max_{r=1,2,\ldots,R} \left\{ \min_{\tilde{\boldsymbol{\gamma}}^{r}\in\tilde{\boldsymbol{\Gamma}}^{r}(\boldsymbol{c}^{*}(\tilde{\boldsymbol{\gamma}}_0))} \overline{Q}_{\mathrm{C}}^{r}(\tilde{\boldsymbol{\gamma}}^{r}; \tilde{\boldsymbol{\gamma}}_0^{r}) \right\}$$

$$= \max_{r=1,2,\ldots,R} \{\tilde{Q}_{\mathrm{C}}^{r,*}(\tilde{\boldsymbol{\gamma}}_0)\} = \tilde{Q}_{\mathrm{C}}^{*}(\tilde{\boldsymbol{\gamma}}_0), \tag{2.55}$$

gdzie Q_{C}^{*} jest optymalną wartością kryterium jakości (2.9) lub w formie opisowej.

1. Ustalamy $\tilde{\boldsymbol{\gamma}} = \tilde{\boldsymbol{\gamma}}_0 \in \hat{\boldsymbol{\Gamma}}$.

2. Rozwiązujemy problem optymalizacyjny (2.48) z ograniczeniami (2.45) i (2.47) (tzn. problem szeregowania) i uzyskujemy $\boldsymbol{c}^{*}(\tilde{\boldsymbol{\gamma}}_0)$, a następnie wyznaczamy zbiory $\overline{\boldsymbol{H}}_{c}^{r,*}$, $r = 1, 2, ..., R$ oraz kryterium jakości w postaci $\overline{Q}_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}}; \tilde{\boldsymbol{\gamma}}_0)$, czyli (2.51).

3. Rozwiązujemy R problemów optymalizacyjnych (2.54) (tzn. R niezależnych problemów komiwojażera) i otrzymujemy $\tilde{\boldsymbol{\gamma}}^{r,*}(\tilde{\boldsymbol{\gamma}}_0)$, $r = 1, 2, ..., R$, a następnie obliczamy wartość kryterium $Q_{\mathrm{C}}^{*}(\tilde{\boldsymbol{\gamma}}_0)$, czyli zależności (2.53).

Zauważmy jeszcze, że pierwsza równość w (2.55) wynika z twierdzenia 2.1, a górne oszacowanie, wchodzących w jej skład wartości, jest spowodowane zastosowaną metodą rozwiązania.

Dotychczasowe rozważania wyróżniają dwie przyczyny powodujące pogorszenie jakości rozwiązania rozpatrywanego problemu. Pierwsza z nich to ustalanie początkowej postaci macierzy $\tilde{\boldsymbol{\gamma}} = \tilde{\boldsymbol{\gamma}}_0$, spowodowane potrzebą uproszczenia problemu optymalizacyjnego z wzajemnie powiązanymi macierzami $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$ – jako zmiennymi optymalizacyjnymi. Przyjęcie konkretnej postaci $\tilde{\boldsymbol{\gamma}}_0$ umożliwia kolejne wyznaczanie optymalnych wartości elementów macierzy $\boldsymbol{c}$ i $\tilde{\boldsymbol{\gamma}}$. Uproszczenie takie – jak się okaże – nie pozostaje bez wpływu na jakość rozwiązania. Druga przyczyna wiąże się również z uproszczeniem, tym razem w procedurze określania macierzy $\boldsymbol{c}$ w drugim kroku metody. Do wyznaczania tej macierzy zaproponowaliśmy metodę rozwiązywania określonego problemu szeregowania, ale dla zadań z pominięciem zjazdów realizatorów do bazy. Zadania zjazdów do bazy występują jednak w ogólnym sformułowaniu problemu optymalizacyjnego (2.44), a ich pominięcie jest pewnym

uproszczeniem pogarszającym jakość rozwiązania. Należy podkreślić, że obie przyczyny pogorszenia jakości rozwiązania, czyli ustalenie początkowej postaci macierzy $\tilde{\gamma} = \tilde{\gamma}_0$ oraz uproszczenie zadania minimalizacji (2.44), są niezależne, a dokładniej – w sposób niezależny wpływają na omawiane pogorszenie jakości. Uzasadnienie podajemy w dowodzie twierdzenia 2.2, które zawiera ilościową ocenę omawianego pogorszenia jakości rozwiązania.

Twierdzenie 2.2

Niech $Q_{\mathrm{C}}^{*}(\tilde{\gamma}_0)$ będzie optymalną wartością kryterium jakości dla problemu po dekompozycji funkcjonalnej. Wtedy

$$\tilde{Q}_{\mathrm{C}}^{*}(\tilde{\gamma}_0) - Q_{\mathrm{C}}^{*} \le \varepsilon_{\gamma} + \varepsilon_{c}, \tag{2.56}$$

gdzie

$$\varepsilon_{\gamma} = \sum_{h=1}^{H+1} \max_{r=1,2,\ldots,R} \left\{ \max_{\substack{g=1,2,\ldots,H+1 \\ g \neq h}} \{\hat{\tau}_{r,g,h}\} - \min_{\substack{g=1,2,\ldots,H+1 \\ g \neq h}} \{\hat{\tau}_{r,g,h}\} \right\}, \tag{2.57}$$

$$\varepsilon_{c} = \max_{r=1,2,\ldots,R} \left\{ \sum_{\substack{g=1 \\ g \neq H+1}}^{H+1} \tilde{\gamma}_{0,g,H+1}^{r} \hat{\tau}_{r,g,H+1} \right\} - \min_{r=1,2,\ldots,R} \left\{ \sum_{\substack{g=1 \\ g \neq H+1}}^{H+1} \tilde{\gamma}_{0,g,H+1}^{r} \hat{\tau}_{r,g,H+1} \right\}. \tag{2.58}$$

Dowód

Dowód będzie się składał z trzech części. W pierwszej części wykażemy, że przyjęcie $\tilde{\gamma} = \tilde{\gamma}_0$ pogarsza jakość rozwiązania nie więcej niż o wartość ε_{γ}, określoną przez (2.57). W drugiej części uzasadnimy prawdziwość równości (2.58) związanej z pogorszeniem jakości na skutek przybliżonego rozwiązania problemu optymalizacyjnego (2.44). Na koniec wykażemy nierówność (2.56).

I. Uzasadnienie wyrażenia (2.57).

Nieoptymalność proponowanej metody rozwiązania, tylko w rozpatrywanym tu aspekcie, wynika faktycznie z możliwości nieprawidłowego doboru czasów dojazdu realizatorów $\hat{\tau}_{r,g,h}$, które są potrzebne do rozwiązania problemu optymalizacyjnego w drugim kroku metody. Doboru tego dokonujemy przez ustalanie macierzy $\tilde{\gamma}_0$. Niech $\Delta\tau_{r,g,h}$ oznacza różnicę czasu dojazdu realizatora r do wykonania zadania h, przyjętego w ocenianej metodzie rozwiązania problemu po dekompozy-

cji funkcjonalnej oznaczanego jako $\hat{\tau}_{r,g,h}$ oraz takiego samego czasu w problemie PC oznaczanego jako $\hat{\tau}^*_{r,g,h}$, czyli

$$\Delta\tau_{r,g,h} \stackrel{\Delta}{=} \hat{\tau}_{r,g,h} - \hat{\tau}^*_{r,g,h}.$$

Oczywiste jest następujące górne oszacowanie $\Delta\tau_{r,g,h}$

$$\Delta\tau_{r,g,h} \le \max_{\substack{g=1,2,...,H+1 \\ g\neq h}} \{\hat{\tau}_{r,g,h}\} - \min_{\substack{g=1,2,...,H+1 \\ g\neq h}} \{\hat{\tau}_{r,g,h}\} \stackrel{\Delta}{=} \overline{\Delta}\tau_{r,g,h}. \tag{2.59}$$

Wtedy różnica odpowiedniego czasu wykonywania zadania *h* przez realizator *r* ma takie samo górne ograniczenie, to znaczy

$$\Delta\tau_{r,h} = \tau_{r,h} - \tau^*_{r,h} = (\overline{\tau}_{r,h} + \hat{\tau}_{r,g,h}) - (\overline{\tau}_{r,h} + \hat{\tau}^*_{r,g,h}) = \Delta\tau_{r,g,h} \le \overline{\Delta}\tau_{r,g,h}. \tag{2.60}$$

Niech $\Delta Q^r_{\mathrm{C},\gamma}$ oznacza zwiększenie czasu pracy realizatora *r* w stosunku do odpowiedniego czasu uzyskanego przez algorytm optymalny, rozwiązujący problem PC, czyli różnicę czasu pracy realizatora *r* dla problemu po dekompozycji funkcjonalnej $\tilde{Q}^{r,*}_{\mathrm{C}}(\tilde{\gamma}_0)$ oraz czasu pracy tego realizatora dla PC, tj. $Q^{r,*}$. Po uwzględnieniu (2.59) i (2.60) oraz oznaczeniu przez $\overline{\boldsymbol{H}}^{r,*}$ zbioru zadań wykonywanych przez realizator *r* wyznaczonego przez algorytm optymalny dla PC, różnicę tę zapisujemy jako

$$\Delta Q^r_{\mathrm{C},\gamma} = \sum_{h\in\overline{\boldsymbol{H}}^{r,*}} \Delta\tau_{r,h} = \sum_{h\in\overline{\boldsymbol{H}}^{r,*}} \Delta\tau_{r,g,h} \le \sum_{h\in\overline{\boldsymbol{H}}^{r,*}} \overline{\Delta}\tau_{r,g,h}. \tag{2.61}$$

Analogicznie szacujemy maksymalną wartość $\Delta Q^r_{\mathrm{C},\gamma}$

$$\Delta Q_{\mathrm{C},\gamma} \stackrel{\Delta}{=} \max_{r=1,2,...,R} \{\Delta Q^r_{\mathrm{C},\gamma}\} \le \max_{r=1,2,...,R} \left\{ \sum_{h\in\overline{\boldsymbol{H}}^{r,*}} \overline{\Delta}\tau_{r,g,h} \right\}. \tag{2.62}$$

Uwzględniamy fakt, że zbiory $\overline{\boldsymbol{H}}^{r,*}$, $r = 1,2,...,R$ są rozłączne, a ich suma jest zbiorem wszystkich zadań wraz ze zjazdem realizatorów do bazy, i kontynuujemy (2.62)

$$\Delta Q_{\mathrm{C},\gamma} \le \sum_{h=1}^{H+1} \max_{r=1,2,...,R} \left\{ \overline{\Delta}\tau_{r,g,h} \right\}$$

$$= \sum_{h=1}^{H+1} \max_{r=1,2,...,R} \left\{ \max_{\substack{g=1,2,...,H+1 \\ g\neq h}} \{\hat{\tau}_{r,g,h}\} - \min_{\substack{g=1,2,...,H+1 \\ g\neq h}} \{\hat{\tau}_{r,g,h}\} \right\} = \varepsilon_\gamma. \tag{2.63}$$

II. Uzasadnienie wyrażenia (2.58).

Rozpatrzmy problem optymalizacyjny (2.44). Wykorzystujemy zapis z (2.48) i szacujemy od góry kryterium $\overline{Q}_{\mathrm{C}}(\tilde{\gamma};\tilde{\gamma}_0)$

$$\overline{Q}_{\mathrm{C}}(\tilde{\gamma};\tilde{\gamma}_0) \le \min_{c}\left[\max_{r=1,2,\ldots,R}\left\{\sum_{g=1}^{H+1}\tilde{\gamma}^{r}_{0,g,h}\hat{\tau}_{r,g,H+1}\right\} + \max_{r=1,2,\ldots,R}\left\{\sum_{h=1}^{H} c_{r,h}T_{r,h}(\tilde{\gamma}^{r}_{0})\right\}\right]. \quad (2.64)$$

Pierwszy składnik, występujący po prawej stronie w (2.64) nie zależy od $\boldsymbol{c}$, a więc nie ma wpływu na minimalizację. Problem minimalizacji drugiego składnika już omówiono. Jakość uzyskanego rozwiązania może być gorsza niż jakość rozwiązania problemu pierwotnego (2.44). Wielkość takiego pogorszenia jakości jest określona przez wartość pierwszego składnika po prawej stronie w (2.64). Jeśli jednak przyjmiemy, że problem minimalizacji długości uszeregowania (2.48) zaczniemy rozwiązywać w momencie czasu równym najkrótszemu czasowi zjazdu realizatora do bazy określonemu przez wartość

$$\min_{r=1,2,\ldots,R}\left\{\sum_{g=1}^{H+1}\tilde{\gamma}^{r}_{0,g,H+1}\hat{\tau}_{r,g,H+1}\right\}, \quad (2.65)$$

to o tę wartość możemy zmniejszyć rozpatrywane pogorszenie jakości rozwiązania, które możemy interpretować jako zwiększenie maksymalnego czasu działania realizatorów. Oznaczamy je przez $\Delta Q_{\mathrm{C},c}$ i po wykorzystaniu (2.64) oraz (2.65) możemy przedstawić w postaci

$$\Delta Q_{\mathrm{C},c} = \max_{r=1,2,\ldots,R}\left\{\sum_{g=1}^{H+1}\tilde{\gamma}^{r}_{0,g,h}\hat{\tau}_{r,g,H+1}\right\} - \min_{r=1,2,\ldots,R}\left\{\sum_{g=1}^{H+1}\tilde{\gamma}^{r}_{0,g,h}\hat{\tau}_{r,g,H+1}\right\} = \varepsilon_c. \quad (2.66)$$

III. Uzasadnienie zależności (2.56).

Wartości liczbowe ε_γ i ε_c pogorszenia jakości rozwiązania wyznaczyliśmy dla założenia, że ich przyczyny są niezależne. Niezależność tę można uzasadnić następująco: Konkretna postać $\tilde{\gamma}_0$ ma oczywiście wpływ na procedurę szeregowania w tym sensie, że ustala dane dla szeregowania, to znaczy czas dojazdu realizatorów do stanowisk, nie ma natomiast wpływu na sam algorytm szeregowania. Jednak wartość ε_c jest możliwie największym pogorszeniem jakości, spowodowanym przybliżonym rozwiązaniem problemu (2.44), i w tym sensie nie zależy od macierzy $\tilde{\gamma}_0$. Wartość ε_c zależy jedynie od danych problemu. Brak zależności w drugą stronę jest oczywisty. Na podstawie tego można określić wartość całkowitego pogorszenia jakości rozwiązania ΔQ_{C} jako sumę (2.63) i (2.66), tj.

$$\Delta Q_{\mathrm{C}} = \Delta Q_{\mathrm{C},\gamma} + \Delta Q_{\mathrm{C},c} \le \varepsilon_\gamma + \varepsilon_c .$$

Wtedy $\Delta Q_{\mathrm{C}} \overset{\Delta}{=} \tilde{Q}^*_{\mathrm{C}}(\tilde{\gamma}_0) - Q^*_{\mathrm{C}} \le \varepsilon_\gamma + \varepsilon_c$.

■

Warto zauważyć, że jeśli czas dojazdu jest taki sam, tj.

$$\hat{\tau}_{r,g,h} \overset{\Delta}{=} \hat{\tau},\ r = 1,2,...,R,\ \ g,h = 1,2,...,H+1,\ \ g \ne h,\ \text{ to } \tilde{Q}^*_{\mathrm{C}}(\tilde{\gamma}_0) = Q^*_{\mathrm{C}}.$$

Rozważmy elementarny przykład obliczeniowy w celu ilustracji przedstawionego algorytmu rozwiązania.

Przykład 2.1

Niech $H = 5$, $R = 2$, a macierze $\bar{\tau}$ i $\hat{\tau}$ są dane w tab. 2.1, 2.2 i 2.3. Przed rozpoczęciem obliczeń rozpatrzmy dokładniej ustalanie macierzy $\tilde{\gamma}_0 \in \hat{\Gamma}$ w pierwszym kroku algorytmu. Zgodnie definicją 2.6, każda macierz $\tilde{\gamma}^r_0$ odznacza się tym, że w każdej kolumnie, to znaczy dla ustalonego h, występuje dokładnie jeden element równy 1. Element taki jednoznacznie wskazuje na stanowisko g, z którego realizator r dojeżdża w celu wykonania zadania h. Wybierając taki element w każdej kolumnie macierzy $\tilde{\gamma}^r_0$, ustalamy dla każdego zadania związany z jego wykonaniem czas dojazdu realizatora. Możemy wyróżnić następujące przykładowe postaci macierzy $\tilde{\gamma}^r_0$, $r = 1,2,...,R$.

$$\tilde{\gamma}^r_{0,g,h} = \begin{cases} 1, \text{ jeśli } \hat{\tau}_{r,g,h} = \max\limits_{\substack{p=1,2,...,H+1 \\ p \ne h}} \{\hat{\tau}_{r,p,h}\} \\ 0, \text{ w przeciwnym razie} \end{cases} \quad g,h = 1,2,...,H+1, \tag{2.67}$$

$$\tilde{\gamma}^r_{0,g,h} = \begin{cases} 1, \text{ jeśli } \hat{\tau}_{r,g,h} = \min\limits_{\substack{p=1,2,...,H+1 \\ p \ne h}} \{\hat{\tau}_{r,p,h}\} \\ 0, \text{ w przeciwnym razie} \end{cases} \quad g,h = 1,2,...,H+1. \tag{2.68}$$

Tabela 2.1. Czasy wykonywania czynności $\bar{\tau}_{r,h}$

r \ h	1	2	3	4	5
1	57	64	60	54	56
2	62	52	53	59	58

Tabela 2.2. Czasy dojazdu $\hat{\tau}_{1,g,h}$

g \ h	1	2	3	4	5	6
1	∞	52	58	64	53	57
2	57	∞	55	53	61	59
3	69	54	∞	52	58	55
4	56	58	57	∞	54	62
5	51	63	61	53	∞	65
6	54	59	62	56	58	∞

Tabela 2.3. Czasy dojazdu $\hat{\tau}_{2,g,h}$

g \ h	1	2	3	4	5	6
1	∞	55	62	58	57	60
2	64	∞	65	53	64	52
3	62	57	∞	52	52	62
4	52	58	61	∞	55	63
5	53	59	54	57	∞	54
6	57	53	57	60	53	∞

Ogólnie, element $\tilde{\gamma}^r_{0,g,h}$ jest równy 1, jeśli g jest k-tym w kolejności stanowiskiem czasowo najbliższym stanowisku h, gdzie $k = 1, 2, ..., H$ oraz jest równy 0 – w przeciwnym przypadku. W wyrażeniu (2.67) $k = H$, a w wyrażeniu (2.68) $k = 1$.

Analiza rozpatrywanego problemu oraz metody jego rozwiązywania dowodzi, że możemy uogólnić definicję zbioru $\hat{\boldsymbol{\Gamma}}^r$ początkowych macierzy $\tilde{\boldsymbol{\gamma}}^r$. Z definicji tej wynika, że elementy macierzy $\tilde{\boldsymbol{\gamma}}^r_0$, podobnie zresztą jak elementy macierzy $\tilde{\boldsymbol{\gamma}}^r$, należą do zbioru $\boldsymbol{\Gamma}^r$, czyli są binarne. Jednak ze względu na stosowaną metodę rozwiązania nie jest to konieczne. Wystarczy, aby macierz $\tilde{\boldsymbol{\gamma}}^r_0$ w sposób jednoznaczny określała rozwiązywany dalej problem szeregowania. W tym celu potrzebne jest tylko spełnienie dwóch warunków podanych w definicji 2.6, a mianowicie $\tilde{\gamma}^r_{h,h} = 0$, $h = 1,2,..., H+1$ oraz $\sum_{g=1}^{H+1} \tilde{\gamma}^r_{g,h} = 1,\ h = 1,2,..., H+1$. Przykładem macierzy $\tilde{\boldsymbol{\gamma}}^r_0$, należącej do tak zmodyfikowanego zbioru $\hat{\boldsymbol{\Gamma}}^r$, jest taka jej postać, w której

$$\tilde{\gamma}^r_{0,g,h} = \begin{cases} \dfrac{\hat{\tau}_{r,g,h}}{\sum_{\substack{p=1 \\ p \neq h}}^{H+1} \hat{\tau}_{r,p,h}}, & g \neq h \\ 0, & g = h \end{cases} . \tag{2.69}$$

W definicji 2.6 podaliśmy niezmodyfikowaną postać zbiorów $\hat{\boldsymbol{\Gamma}}^r$, aby bez wyraźnej potrzeby nie komplikować zapisu.

Przykład będziemy rozwiązywać dla dwóch postaci macierzy $\tilde{\boldsymbol{\gamma}}_0$, a mianowicie (2.67) – przypadek 1 i (2.68) – przypadek 2. Macierz $\tilde{\boldsymbol{\gamma}}_0$ umożliwia obliczenie czasów $T_{r,h}(\tilde{\boldsymbol{\gamma}}_0^r)$, określonych w zależności (2.48). Czasy te podano w tab. 2.4 i 2.5. Następnym krokiem metody rozwiązania – po ustaleniu $\tilde{\boldsymbol{\gamma}}_0$ – jest rozwiązanie problemu szeregowania. Wyniki, po zastosowaniu w tym prostym przypadku przeglądu zupełnego, przedstawiono w tab. 2.6 i 2.7.

Tabela 2.4. Czasy $T_{r,h}(\tilde{\boldsymbol{\gamma}}_0^r)$ dla przypadku 1

r \ h	1	2	3	4	5
1	126	127	122	118	117
2	126	111	118	119	122

Tabela 2.5. Czasy $T_{r,h}(\tilde{\boldsymbol{\gamma}}_0^r)$ dla przypadku 2

r \ h	1	2	3	4	5
1	108	116	115	106	109
2	114	105	107	111	110

Tabela 2.6. Macierz $\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0)$ dla przypadku 1

r \ h	1	2	3	4	5	6
1	1	0	0	0	1	1
2	0	1	1	1	0	1

Tabela 2.7. Macierz $\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0)$ dla przypadku 2

r \ h	1	2	3	4	5	6
1	1	0	0	1	0	1
2	0	1	1	0	1	1

Jak łatwo sprawdzić, $\boldsymbol{c}^*(\tilde{\boldsymbol{\gamma}}_0) \in \tilde{\boldsymbol{C}}$, czyli są spełnione (2.26) i (2.27). Równoważna postać rozwiązania, czyli zbiory $\overline{\boldsymbol{H}}_c^{r,*}$, jest następująca: $\overline{\boldsymbol{H}}_c^{1,*} = \{1,5,6\}$ i $\overline{\boldsymbol{H}}_c^{2,*} = \{2,3,4,6\}$ dla przypadku 1. oraz $\overline{\boldsymbol{H}}_c^{1,*} = \{1,4,6\}$ i $\overline{\boldsymbol{H}}_c^{2,*} = \{2,3,5,6\}$ dla przypadku 2. Realizacja kolejnego kroku metody polega na rozwiązaniu dwóch problemów komiwojażera (dla $r = 1$ i $r = 2$). Potrzebne dane, uzyskane z tab. 2.2 i 2.3 dla obu rozpatrywanych przypadków, przedstawiono w tab. 2.8–2.11.

Rozwiązania, czyli macierze $\tilde{\boldsymbol{\gamma}}^{r,*}(\tilde{\boldsymbol{\gamma}}_0)$, przedstawione w tab. 2.12–2.15, uzyskano również metodą przeglądu zupełnego. Macierze te określają trasy, wzdłuż których poruszają się realizatory. Dla przypadku 1:

$$\text{realizator } r = 1 \;-\; \text{trasa}: 6 \to 5 \to 1 \to 6,$$
$$\text{realizator } r = 2 \;-\; \text{trasa}: 6 \to 3 \to 4 \to 2 \to 6.$$

Tabela 2.8. Czasy $\hat{\tau}_{1,g,h}$ dla przypadku 1

g \ h	1	5	6
1	∞	53	57
5	51	∞	65
6	54	58	∞

Tabela 2.9. Czasy $\hat{\tau}_{2,g,h}$ dla przypadku 1

g \ h	2	3	4	6
2	∞	65	53	52
3	57	∞	52	62
4	58	61	∞	63
6	53	57	60	∞

Tabela 2.10. Czasy $\hat{\tau}_{1,g,h}$ dla przypadku 2

g \ h	1	4	6
1	∞	64	57
4	56	∞	62
6	54	56	∞

Tabela 2.11. Czasy $\hat{\tau}_{2,g,h}$ dla przypadku 2

g \ h	2	3	5	6
2	∞	65	64	52
3	57	∞	52	62
5	59	54	∞	54
6	53	57	53	∞

Dla przypadku 2:

$$\text{realizator } r = 1 \;-\; \text{trasa}: 6 \to 4 \to 1 \to 6,$$
$$\text{realizator } r = 2 \;-\; \text{trasa}: 6 \to 5 \to 3 \to 2 \to 6.$$

Otrzymane rozwiązania są dopuszczalne, ponieważ każda z macierzy przedstawionych w tab. 2.12–2.15 należy do zbioru $\tilde{\Gamma}^r(\boldsymbol{c}^*(\tilde{\gamma}_0))$ – są spełnione bowiem zależności od (2.38) do (2.41). Kolejność wykonywania zadań jest przedstawiona również na rys. 2.3 i 2.4 w postaci wykresów Gantta (ostatnie pola oznaczają zjazdy do bazy, a w nawiasach podano czasy wykonywania zadań).

Obliczymy teraz wartości kryterium jakości dla obu przypadków.

Tabela 2.12. Macierz $\tilde{\gamma}^{1,*}(\tilde{\gamma}_0)$ dla przypadku 1

g \ h	1	2	3	4	5	6
1	0	0	0	0	0	1
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	1	0	0	0	0	0
6	0	0	0	0	1	0

Tabela 2.13. Macierz $\tilde{\gamma}^{2,*}(\tilde{\gamma}_0)$ dla przypadku 1

g \ h	1	2	3	4	5	6
1	0	0	0	0	0	0
2	0	0	0	0	0	1
3	0	0	0	1	0	0
4	0	1	0	0	0	0
5	0	0	0	0	0	0
6	0	0	1	0	0	0

Tabela 2.14. Macierz $\tilde{\gamma}^{1,*}(\tilde{\gamma}_0)$ dla przypadku 2

g \ h	1	2	3	4	5	6
1	0	0	0	0	0	1
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	1	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	1	0	0

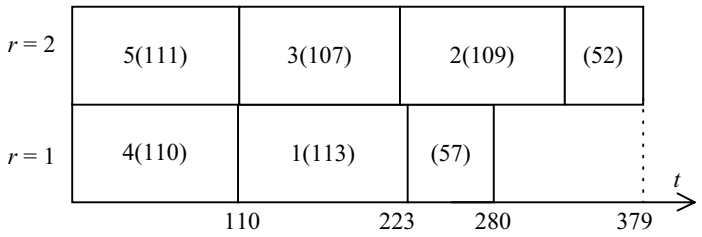
Tabela 2.15. Macierz $\tilde{\gamma}^{2,*}(\tilde{\gamma}_0)$ dla przypadku 2

g \ h	1	2	3	4	5	6
1	0	0	0	0	0	0
2	0	0	0	0	0	1
3	0	1	0	0	0	0
4	0	0	0	0	0	0
5	0	0	1	0	0	0
6	0	0	0	0	1	0

Dla przypadku 1:

$$\tilde{Q}_{\mathrm{C}}^{*}(\tilde{\gamma}_0) = \max\{\hat{\tau}_{1,6,5} + \bar{\tau}_{1,5} + \hat{\tau}_{1,5,1} + \bar{\tau}_{1,1} + \hat{\tau}_{1,1,6}, \hat{\tau}_{2,6,3} + \bar{\tau}_{2,3} + \hat{\tau}_{2,3,4} + \bar{\tau}_{2,4} + \hat{\tau}_{2,4,2} + \bar{\tau}_{2,2} + \hat{\tau}_{2,2,6}\} = \max\{58+56+51+57+57{,}57 + 53+52+59+58+52+52\} = \max\{279{,}383\} = 383.$$

Rys. 2.3. Kolejność wykonywania zadań dla przypadku 1 w przykładzie 2.1



Rys. 2.4. Kolejność wykonywania zadań dla przypadku 2 w przykładzie 2.1

Dla przypadku 2:

$$\tilde{\tilde{Q}}^{*}_{\mathrm{C}}(\tilde{\gamma}_0) = \max\{\hat{\tau}_{1,6,4} + \bar{\tau}_{1,4} + \hat{\tau}_{1,4,1} + \bar{\tau}_{1,1} + \hat{\tau}_{1,1,6,}\hat{\tau}_{2,6,5} + \bar{\tau}_{2,5} + \hat{\tau}_{2,5,3} + \bar{\tau}_{2,3} + \hat{\tau}_{2,3,2} + \bar{\tau}_{2,2} + \hat{\tau}_{2,2,6}\} = \max\{56+54+56+57+57{,}53 + 58+54+53+57+52+52\} = \max\{280{,}379\} = 379.$$

Tak więc stwierdzamy, że w rozpatrywanym przykładzie wybór $\tilde{\gamma}_0$ miał wpływ na osiągnięty rezultat, ponieważ zarówno wyznaczone trasy przejazdów realizatorów, jak i wartości kryterium jakości, dla obu przypadków są różne. W przypadku 2. uzyskaliśmy nieznacznie mniejszą wartość kryterium jakości.

Oszacujemy teraz pogorszenie jakości spowodowane zastosowaną metodą rozwiązania, czyli obliczymy wartości

$$\varepsilon_{\gamma} = \sum_{h=1}^{6} \max_{r=1,2} \left\{ \max_{\substack{g=1,2,\ldots,6 \\ g \neq h}} \{\hat{\tau}_{r,g,h}\} - \min_{\substack{g=1,2,\ldots,6 \\ g \neq h}} \{\hat{\tau}_{r,g,h}\} \right\}$$

$$= 18+11+11+12+12+11 = 75$$

oraz dla przypadku 1:

$$\varepsilon_c = \max_{r=1,2}\left\{\sum_{\substack{g=1\\ g\neq 6}}^{6} \tilde{\gamma}^r_{0,g,6}\,\hat{\tau}_{r,g,6}\right\} - \min_{r=1,2}\left\{\sum_{\substack{g=1\\ g\neq 6}}^{6} \tilde{\gamma}^r_{0,g,6}\,\hat{\tau}_{r,g,6}\right\}$$

$$= \max_{r=1,2}\left\{\max_{\substack{p=1,2,\ldots,6\\ p\neq 6}}\{\hat{\tau}_{r,p,6}\}\right\} - \min_{r=1,2}\left\{\max_{\substack{p=1,2,\ldots,6\\ p\neq 6}}\{\hat{\tau}_{r,p,6}\}\right\}$$

$$= \max\{65, 63\} - \min\{65, 63\} = 65 - 63 = 2$$

i dla przypadku 2:

$$\varepsilon_c = \max_{r=1,2}\left\{\sum_{\substack{g=1\\ g\neq 6}}^{6} \tilde{\gamma}^r_{0,g,6}\hat{\tau}_{r,g,6}\right\} - \min_{r=1,2}\left\{\sum_{\substack{g=1\\ g\neq 6}}^{6} \tilde{\gamma}^r_{0,g,6}\hat{\tau}_{r,g,6}\right\}$$

$$= \max_{r=1,2}\left\{\min_{\substack{p=1,2,\ldots,6\\ p\neq 6}}\{\hat{\tau}_{r,p,6}\}\right\} - \min_{r=1,2}\left\{\min_{\substack{p=1,2,\ldots,6\\ p\neq 6}}\{\hat{\tau}_{r,p,6}\}\right\}$$

$$= \max\{55, 52\} - \min\{55, 52\} = 55 - 52 = 3.$$

Wartość ε_γ dla obu przypadków jest taka sama – wyraża ona maksymalne pogorszenie jakości rozwiązania, spowodowane niewłaściwym doborem $\tilde{\gamma}_0$, wartość ε_c zależy natomiast od $\tilde{\gamma}_0$ w sposób określony w dowodzie twierdzenia 2.2. Ostatecznie maksymalne pogorszenie jakości rozwiązania, spowodowane zastosowaną metodą, dla przypadku 1. wynosi 77, a dla przypadku 2. jest równe 78.

■

2.3.2. Minimalizacja maksymalnego opóźnienia

Prezentację tego przypadku będziemy prowadzić podobnie jak poprzednio, to znaczy dla kryterium w postaci długości uszeregowania. Dlatego wyjaśnianie kwestii

analogicznych będzie prowadzone w sposób bardziej skrótowy. Szczególną uwagę zwrócimy na zagadnienia oryginalne dla rozważanego kryterium. Pierwszą istotną różnicą jest fakt, że decyzje dla zadań dotyczą nie tylko realizatorów, ale również przedziałów czasu, w których te zadania mają być wykonane. Okazuje się, że w tym przypadku również można zastosować dekompozycję funkcjonalną problemu PL. Wprowadźmy nowe macierze decyzyjne o mniejszych rozmiarach niż $\boldsymbol{\alpha}$, to znaczy

$$\tilde{\boldsymbol{\alpha}}^{r,l} = [\tilde{\alpha}_{g,h}^{r,l}]_{\substack{g=1,2,...,H+1 \\ h=1,2,...,H}}, \quad r = 1,2,...,R, \; l = 1,2,...,H, \tag{2.70}$$

gdzie: $\tilde{\alpha}_{g,h}^{r,l} = \begin{cases} 1, & \text{jeśli wykonywanie zadania } h \text{ przez realizator } r \text{ po dojeździe} \\ & \text{ze stanowiska } g \text{ rozpoczyna się przed zakonczeniem przedziału } l, \\ 0, & \text{w przeciwnym przypadku.} \end{cases}$

Macierze $\tilde{\boldsymbol{\alpha}}^{r,l}$ tworzą macierz $\tilde{\boldsymbol{\alpha}}$

$$\tilde{\boldsymbol{\alpha}} = \begin{bmatrix} \tilde{\boldsymbol{\alpha}}^{1,1}, & \cdots, & \tilde{\boldsymbol{\alpha}}^{1,H} \\ \cdots, & \cdots, & \cdots \\ \tilde{\boldsymbol{\alpha}}^{R,1}, & \cdots, & \tilde{\boldsymbol{\alpha}}^{R,H} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{\alpha}}^{1} \\ \cdots \\ \tilde{\boldsymbol{\alpha}}^{R} \end{bmatrix},$$

gdzie $\tilde{\boldsymbol{\alpha}}^{r} = [\tilde{\boldsymbol{\alpha}}^{r,1}, \tilde{\boldsymbol{\alpha}}^{r,2}, ..., \tilde{\boldsymbol{\alpha}}^{r,H}]$. Na macierze $\tilde{\boldsymbol{\alpha}}^{r,l}$ nakładamy ograniczenia

$$\tilde{\alpha}_{h,h}^{r,l} = 0, \; h = 1,2,...,H, \tag{2.71}$$

$$\sum_{g=1}^{H+1} \tilde{\alpha}_{g,h}^{r,l} \le 1, \; h = 1,2,...,H, \tag{2.72}$$

$$\sum_{g=1}^{H+1} \tilde{\alpha}_{g,p}^{r,l} = \sum_{h=1}^{H} \tilde{\alpha}_{p,h}^{r,H}, \; p = 1,2,...,H, \; p \neq \bar{h}^{r}(H), \tag{2.73}$$

$$\sum_{h=1}^{H} \tilde{\alpha}_{H+1,h}^{r,l} \le 1, \tag{2.74}$$

$$\tilde{\alpha}^{r,l} \in \bar{\boldsymbol{S}}_{r,l}, \tag{2.75}$$

gdzie $\bar{\boldsymbol{S}}_{r,l} = \{[\tilde{\alpha}_{g,h}^{r,l}]: \sum_{g \in \boldsymbol{H}_{\bar{S}}^{r,l}} \sum_{h \in \boldsymbol{H}_{\bar{S}}^{r,l}} \tilde{\alpha}_{g,h}^{r,l} \le H_{\bar{S}}^{r,l} - 1\}$, natomiast $\boldsymbol{H}_{\bar{S}}^{r,l}$ i $\boldsymbol{H}^{r,l}$ są odpowiednio dowolnym niepustym podzbiorem $\boldsymbol{H}^{r,l}$ i zbiorem zadań wykonywanych przez realizator *r* w przedziale *l*,

$$\sum_{g \in \boldsymbol{H}^{r,l-1}} \sum_{h \in \boldsymbol{H}^{r,l}} \tilde{\alpha}_{g,h}^{r,l} = 1, \; l = 2,3,...,H. \tag{2.76}$$

Umożliwiają one zdefiniowanie zbioru $\tilde{\boldsymbol{\alpha}}^{r,l}$ dopuszczalnych macierzy $\tilde{\boldsymbol{\alpha}}^{r,l}$.

Definicja 2.7

Zbiór

$$\tilde{\boldsymbol{\alpha}}^{r,l} = \{\tilde{\boldsymbol{\alpha}}^{r,l} \in \boldsymbol{\alpha}^{r,l} : \text{zachodzi } (2.71) \wedge (2.72) \wedge (2.73) \wedge (2.74) \wedge (2.75) \wedge (2.76)\}$$

nazywamy zbiorem dopuszczalnych macierzy $\tilde{\boldsymbol{\alpha}}^{r,l}$, gdzie

$$\boldsymbol{\alpha}^{r,l} = \{\tilde{\boldsymbol{\alpha}}^{r,l} = [\tilde{\alpha}_{g,h}^{r,l}]: \tilde{\alpha}_{g,h}^{r,l} \in \{0,1\}, \; g = 1,2,...,H+1, \; h = 1,2,...,H\}.$$

■

Ponadto $\tilde{\boldsymbol{\alpha}} = \tilde{\boldsymbol{\alpha}}^{1,1} \times \cdots \times \tilde{\boldsymbol{\alpha}}^{1,H} \times \cdots \times \tilde{\boldsymbol{\alpha}}^{R,1} \times \cdots \times \tilde{\boldsymbol{\alpha}}^{R,H}$. Warunki (2.71)–(2.75) mają analogiczne znaczenie do tych, które definiują zbiory $\tilde{\boldsymbol{\Gamma}}^{r}$ dla przypadku rozpatrywanego w podpunkcie 2.3.1. Dodatkowy warunek (2.76) zapewnia istnienie jednego połączenia między częściami tras przejazdów realizatorów wykonywanych w sąsiednich przedziałach czasu. Podobnie jak w p. 2.3.1 macierze decyzyjne $\tilde{\boldsymbol{\alpha}}^{r,l}$ dla różnych *r* i *l* nie są niezależne. Aby zapewnić ich koordynację, wprowadzamy macierz

$$\boldsymbol{a} = [a_{r,h,l}]_{\substack{r=1,2,...,R \\ h,l=1,2,...,H}}, \tag{2.77}$$

gdzie: $a_{r,h,l} = \begin{cases} 1, & \text{jeśli wykonywanie zadania } h \text{ przez realizator } r \\ & \text{rozpoczyna się przed zakończeniem przedzialu } l, \\ 0, & \text{w przeciwnym przypadku} \end{cases}$

z ograniczeniami

$$\sum_{r=1}^{R} a_{r,h,l} = 1, \quad h = 1,2,...,H, \quad l \ge h, \tag{2.78}$$

które są wykorzystywane do definicji zbioru $\tilde{\boldsymbol{A}}$ dopuszczalnych macierzy $\boldsymbol{a}$.

Definicja 2.8

Zbiór $\tilde{\boldsymbol{A}} = \{\boldsymbol{a} \in \boldsymbol{A} : \text{zachodzi (2.78)}\}$ nazywamy zbiorem dopuszczalnych macierzy $\boldsymbol{a}$, gdzie $\boldsymbol{A} = \{\boldsymbol{a} = [a_{r,h,l}] : a_{r,h,l} \in \{0, 1\},\ r = 1,2, ..., R,\ h, l = 1, 2, ..., H\}$.

■

W równości (2.78) zawarte jest wymaganie, że każde zadanie ma być wykonane jeden raz. Z definicji macierzy $\boldsymbol{a}$ wynika własność

$$a_{r,h,l} = 1 \rightarrow a_{r,h,m} = 1, \text{ dla } m = 2,...,H,\ m > l. \tag{2.79}$$

Wprowadzenie macierzy $\boldsymbol{a}$ umożliwia jeszcze lepsze określenie macierzy $\tilde{\boldsymbol{\alpha}}$, a mianowicie zawarte w macierzach $\tilde{\boldsymbol{\alpha}}^{r,l}$ informacje o zadaniach wykonywanych przez realizator *r* w przedziale *l* są istotne tylko do tych zadań, dla których $a_{r,h,l} = 1$. Oznaczmy zbiór takich zadań jako

$$\boldsymbol{H}^{r,l}(\boldsymbol{a}) = \{h \in \boldsymbol{H} : a_{r,h,l} = 1\},\ r = 1,2,...,R,\ l = 1,2,...,H. \tag{2.80}$$

Wtedy ograniczenia (2.71)–(2.76) zapisujemy jako

$$\tilde{\alpha}_{h,h}^{r,l} = 0,\ h \in \boldsymbol{H}^{r,l}(\boldsymbol{a}), \tag{2.81}$$

$$\sum_{g=1}^{H+1} \tilde{a}_{g,h}^{r,l} = 1,\ h \in \boldsymbol{H}^{r,l}(\boldsymbol{a}), \tag{2.82}$$

$$\sum_{g=1}^{H+1} \tilde{\alpha}_{g,p}^{r,l} = \sum_{h=1}^{H} \tilde{\alpha}_{p,h}^{r,l},\ p \in \boldsymbol{H}^{r,l}(\boldsymbol{a}),\ p \neq \bar{h}^{r}(l), \tag{2.83}$$

gdzie $\bar{h}^{r}(l)$ jest ostatnim zadaniem wykonywanym przez realizator *r* w przedziale czasu *l*

$$\sum_{h \in \boldsymbol{H}^{r,l}(\boldsymbol{a})} \tilde{\alpha}_{H+1,h}^{r,l} = 1, \tag{2.84}$$

$$[\tilde{\alpha}_{g,h}^{r,l}] \in \bar{\boldsymbol{S}}_{r,l},\ g, h \in \boldsymbol{H}^{r,l}(\boldsymbol{a}), \tag{2.85}$$

$$\sum_{g \in \boldsymbol{H}^{r,l-1}(\boldsymbol{a})} \sum_{h \in \boldsymbol{H}^{r,l}(\boldsymbol{a})} \tilde{\alpha}_{g,h}^{r,l} = 1,\ l = 2,3,...,H. \tag{2.86}$$

Uściśleniu ulegają również zbiory $\tilde{\boldsymbol{\alpha}}^{r,l}$.

Definicja 2.9

Zbiór

$$\tilde{\boldsymbol{\alpha}}^{r,l}(\boldsymbol{a}) = \{\tilde{\alpha}^{r,l} \in \boldsymbol{\alpha}^{r,l} : \text{zachodzi } (2.81) \wedge (2.82) \wedge (2.83) \wedge (2.84) \wedge (2.85) \wedge (2.86)\}$$

nazywamy zbiorem dopuszczalnych macierzy $\tilde{\alpha}^{r,l}$ zależnych od $\boldsymbol{a}$.

■

Dodatkowo oznaczmy

$$\tilde{\boldsymbol{\alpha}}(\boldsymbol{a}) = \tilde{\boldsymbol{\alpha}}^{1,1}(\boldsymbol{a}) \times \cdots \times \tilde{\boldsymbol{\alpha}}^{1,H}(\boldsymbol{a}) \times \cdots \times \tilde{\boldsymbol{\alpha}}^{R,1}(\boldsymbol{a}) \times \cdots \times \tilde{\boldsymbol{\alpha}}^{R,H}(\boldsymbol{a}) .$$

Zauważmy, że w przedstawionym opisie macierzy decyzyjnych oraz ograniczeń, pominięto zadania zjazdów realizatorów do bazy, ponieważ – jak to już wcześniej podano – nie mają one wpływu na wartość maksymalnego opóźnienia. Dla zadań takich nie określono terminów zakończenia. Fakt ten został uwzględniony w postaci kryterium jakości podanym w twierdzeniu 2.3, gdzie są określone warunki równoważności problemu wyjściowego PL i problemu optymalizacyjnego uzyskanego w wyniku zastosowanej dekompozycji funkcjonalnej.

Twierdzenie 2.3

Jeżeli związek między macierzami $\boldsymbol{a}$, $\tilde{\alpha}$ oraz macierzą $\boldsymbol{\alpha}$ jest dany w postaci równań

$$\alpha_{r,g,h,l} = a_{r,h,l}\tilde{\alpha}^{r,l}_{g,h}, \quad g = 1,2,...,H+1, \quad h,l = 1,2,...,H, \quad r = 1,2,...,R, \quad (2.87)$$

to problem wyjściowy PL jest równoważny w sensie kryterium jakości i ograniczeń problemowi optymalizacyjnemu polegającemu na minimalizacji względem $\boldsymbol{a} \in \tilde{\boldsymbol{A}}$ i $\tilde{\alpha} \in \tilde{\boldsymbol{\alpha}}(\boldsymbol{a})$ kryterium jakości

$$\tilde{Q}_{\mathrm{L}}(\boldsymbol{a}, \tilde{\boldsymbol{\alpha}}) = \max_{r=1,2,...,R} \left\{ \max_{l=1,2,...,H} \left\{ \sum_{h=1}^{H} a_{r,h,l} \left(\bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\alpha}^{r,l}_{g,h} \cdot \hat{\tau}_{r,g,h} \right) - d_l \right\} \right\}. \quad (2.88)$$

Dowód

W celu wykazania równoważności kryteriów jakości wystarczy porównać wyrażenia (2.19) i (2.88), wykorzystując przy tym własności (2.87) i (2.82). Prawdziwy jest ciąg przekształceń

$$Q_{\mathrm{L}}(\boldsymbol{\alpha}) = \max_{r=1,2,\ldots,R}\left\{\max_{l=1,2,\ldots,H}\left\{\sum_{h=1}^{H}\sum_{g=1}^{H+1}\alpha_{r,g,h,l}\left(\bar{\tau}_{r,h}+\hat{\tau}_{r,g,h}\right)-d_l\right\}\right\}$$

$$= \max_{r=1,2,\ldots,R}\left\{\max_{l=1,2,\ldots,H}\left\{\sum_{h=1}^{H}\sum_{g=1}^{H+1}a_{r,h,l}\tilde{\alpha}_{g,h}^{r,l}\left(\bar{\tau}_{r,h}+\hat{\tau}_{r,g,h}\right)-d_l\right\}\right\}$$

$$= \max_{r=1,2,\ldots,R}\left\{\max_{l=1,2,\ldots,H}\left\{\sum_{h=1}^{H}a_{r,h,l}\left(\bar{\tau}_{r,h}\sum_{g=1}^{H+1}\tilde{\alpha}_{g,h}^{r,l}+\sum_{g=1}^{H+1}\tilde{\alpha}_{g,h}^{r,l}\hat{\tau}_{r,g,h}\right)-d_l\right\}\right\}$$

$$= \max_{r=1,2,\ldots,R}\left\{\max_{l=1,2,\ldots,H}\left\{\sum_{h=1}^{H}a_{r,h,l}\left(\bar{\tau}_{r,h}+\sum_{g=1}^{H+1}\tilde{\alpha}_{g,h}^{r,l}\hat{\tau}_{r,g,h}\right)-d_l\right\}\right\} = \tilde{Q}_{\mathrm{L}}(\boldsymbol{a},\tilde{\boldsymbol{\alpha}}).$$

Równoważność ograniczeń (2.13) i (2.81) jest oczywista. Wymaganie, aby każde zadanie było wykonane jeden raz jest wyrażone równością (2.14) dla problemu PL oraz zależnościami (2.78) i (2.82). Ich równoważność wynika z następującego przekształcenia

$$\sum_{r=1}^{R}\sum_{g=1}^{H+1}\alpha_{r,g,h,l} = \sum_{r=1}^{R}\sum_{g=1}^{H+1}a_{r,h,l}\tilde{\alpha}_{g,h}^{r,l} = \sum_{r=1}^{R}\left(a_{r,h,l}\sum_{g=1}^{H+1}\tilde{\alpha}_{g,h}^{r,l}\right) = \sum_{r=1}^{R}a_{r,h,l} = 1.$$

Porównujemy ograniczenia (2.15) i (2.83). W pierwszym z nich zastępujemy $\alpha_{r,g,p,l}$ przez $a_{r,p,l}\tilde{\alpha}_{g,p}^{r,l}$ oraz $\alpha_{r,p,h,l}$ przez $a_{r,h,l}\tilde{\alpha}_{p,h}^{r,l}$ dla $p \neq \bar{h}^{r}(l)$ i otrzymujemy

$$\sum_{l=1}^{H}\sum_{g=1}^{H}a_{r,p,l}\tilde{\alpha}_{g,p}^{r,l} = \sum_{l=1}^{H}\sum_{h=1}^{H}a_{r,h,l}\tilde{\alpha}_{p,h}^{r,l}.$$

Jest to zapis ograniczenia (2.83), ponieważ $a_{r,p,l} = a_{r,h,l} = 1$. Równoważności równości (2.16) i (2.84) nie ma potrzeby dodatkowo uzasadniać. Na koniec przekształćmy zbiór $\bar{\boldsymbol{S}}$ z (2.17), kładąc $a_{r,h,l}\tilde{\alpha}_{g,h}^{r,l}$ w miejsce $\alpha_{r,g,h,l}$. Otrzymujemy

$$\bar{\boldsymbol{S}} = \left\{\left[a_{r,h,l}\tilde{\alpha}_{g,h}^{r,l}\right]: \sum_{l\in\bar{\boldsymbol{H}}_{\bar{S}}}\sum_{h\in\bar{\boldsymbol{H}}_{\bar{S}}}a_{r,h,l}\sum_{g\in\bar{\boldsymbol{H}}_{\bar{S}}}\tilde{\alpha}_{g,h}^{r,l} \leq \bar{H}_{\bar{S}}-1\right\},\quad r=1,2,\ldots,R. \tag{2.89}$$

Biorąc pod uwagę (2.86) możemy ten zbiór zapisać jako

$$\overline{\boldsymbol{S}}_{r,l} = \left\{ \left[\tilde{\alpha}_{g,h}^{r,l} \right] : \sum_{g \in \overline{\boldsymbol{H}}_{\tilde{S}}^{r,l}} \sum_{h \in \overline{\boldsymbol{H}}_{\tilde{S}}^{r,l}} \tilde{\alpha}_{g,h}^{r,l} \le \overline{H}_{\tilde{S}}^{r,l} - 1 \right\},\ r = 1, 2, ..., R. \tag{2.90}$$

W wyrażeniu (2.89) jest zapisane wymaganie, że trasa żadnego realizatora nie zawiera podcykli. Podobne wymaganie, tylko dla bieżącego przedziału *l*, jest przedstawione w formie wyrażenia (2.90). Ponadto wykorzystany warunek (2.86) zapewnia, że fragmenty tras realizatorów dla dwóch sąsiednich przedziałów czasu mają dokładnie jedno połączenie, a więc nie zawierają podcykli. Dlatego zbiór (2.90) razem z warunkiem (2.86) jest równoważny zbiorowi (2.89). Tak więc zarówno kryteria jakości jak i ograniczenia dla obu rozpatrywanych problemów optymalizacyjnych są równoważne.

■

Powiązanie między macierzami $\boldsymbol{a}$ i $\tilde{\boldsymbol{\alpha}}$ można, tak jak poprzednio, wyrazić w formie relacji

$$\begin{aligned} &\overline{\boldsymbol{W}}_1(\boldsymbol{a}, \tilde{\boldsymbol{\alpha}}) = \{(\boldsymbol{a}, \tilde{\boldsymbol{\alpha}}) \in \boldsymbol{A} \times \boldsymbol{\alpha} : \\ &\text{zachodzi}\,(2.78) \wedge (2.81) \wedge (2.82) \wedge (2.83) \wedge (2.84) \wedge (2.85) \wedge (2.86)\}. \end{aligned} \tag{2.91}$$

■

Ostatecznie, po dokonaniu dekompozycji funkcjonalnej, należy rozwiązać następujący problem optymalizacyjny.

Dla danych takich samych jak w problemie wyjściowym PL_z punktu 2.2 należy wyznaczyć wartości elementów macierzy $\boldsymbol{a} \in \tilde{\boldsymbol{A}}$ oraz $\tilde{\boldsymbol{\alpha}}(\boldsymbol{a}) \in \tilde{\boldsymbol{\alpha}}(\boldsymbol{a})$ tak, aby minimalizować kryterium jakości szeregowania (2.88).

Tak jak w przypadku minimalizacji długości uszeregowania, będziemy proponować oddzielne wyznaczanie macierzy $\boldsymbol{a}$ i $\tilde{\boldsymbol{\alpha}}$. Aby móc zastosować taki sposób rozwiązania, uwzgledniwszy powiązanie obu macierzy, należy określić początkową postać macierzy $\tilde{\boldsymbol{\alpha}}$, czyli macierz oznaczaną dalej jako $\tilde{\boldsymbol{\alpha}}_0$, należącą do zbioru

$$\hat{\boldsymbol{\alpha}} = \hat{\boldsymbol{\alpha}}^{1,1} \times \cdots \times \hat{\boldsymbol{\alpha}}^{1,H} \times \cdots \times \hat{\boldsymbol{\alpha}}^{R,1} \times \cdots \times \hat{\boldsymbol{\alpha}}^{R,H}.$$

Bieżący zbiór $\hat{\boldsymbol{\alpha}}^{r,l}$ jest określony w definicji 2.10.

Definicja 2.10

Zbiór $\hat{\boldsymbol{\alpha}}^{r,l} = \{\tilde{\boldsymbol{\alpha}}^{r,l} \in \boldsymbol{\alpha}^{r,l} : \tilde{\alpha}_{h,h}^{r,l} = 0,\ \text{i}\ \sum_{g=1}^{H+1} \tilde{\alpha}_{g,h}^{r,l} = 1,\ h = 1, 2, ..., H\}$ nazywamy zbiorem początkowych postaci macierzy $\tilde{\boldsymbol{\alpha}}^{r,l}$.

■

Możemy teraz podać bardziej ogólną postać relacji wiążącej macierze $\boldsymbol{a}$ i $\tilde{\boldsymbol{\alpha}}$, to znaczy

$$\overline{\boldsymbol{W}}_2(\boldsymbol{a},\tilde{\boldsymbol{\alpha}}) = \{(\boldsymbol{a},\tilde{\boldsymbol{\alpha}}) \in \boldsymbol{A}\times\boldsymbol{\alpha} : \text{zachodzi } (2.78) \wedge \tilde{\alpha}_{h,h}^{r,l} = 0 \ \wedge \ \sum_{g=1}^{H+1} \tilde{\alpha}_{g,h}^{r,l} = 1, \ h = 1,2,...,H\}.$$

Wtedy

$$\overline{\boldsymbol{W}}_1(\boldsymbol{a},\tilde{\boldsymbol{\alpha}}) \subseteq \overline{\boldsymbol{W}}_2(\boldsymbol{a},\tilde{\boldsymbol{\alpha}})$$

i postać macierzy $\tilde{\boldsymbol{\alpha}} = \tilde{\boldsymbol{\alpha}}_0 \in \hat{\boldsymbol{\alpha}}$ nie determinuje postaci macierzy $\boldsymbol{\alpha}$. Po przyjęciu macierzy $\tilde{\boldsymbol{\alpha}}$ jako $\tilde{\boldsymbol{\alpha}}_0 \in \hat{\boldsymbol{\alpha}}$ otrzymujemy problem minimalizacji kryterium

$$\min_{r=1,2,...,R}\left\{\min_{l=1,2,...,H}\left\{\sum_{h=1}^{H} a_{r,h,l} B_{r,h,l}\left(\tilde{\alpha}_0^{r,l}\right) - d_l\right\}\right\}, \tag{2.92}$$

gdzie

$$B_{r,h,l}(\tilde{\alpha}_0^{r,l}) = \bar{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\alpha}_{0,g,h}^{r,l} \hat{\tau}_{r,g,h}$$

z macierzą $\boldsymbol{a}$ jako zmienną optymalizacyjną oraz z ograniczeniami (2.78). Jest to analityczny zapis problemu szeregowania zadań niezależnych i niepodzielnych z równymi momentami gotowości na realizatorach dowolnych w celu minimalizacji maksymalnego opóźnienia. Rozwiązaniem jest optymalna postać macierzy $\boldsymbol{a}^*(\tilde{\boldsymbol{\alpha}}_0)$, zależna od $\tilde{\boldsymbol{\alpha}}_0$ i kryterium jakości szeregowania w postaci

$$\overline{Q}_{\mathrm{L}}(\tilde{\boldsymbol{\alpha}};\tilde{\boldsymbol{\alpha}}_0) = \max_{r=1,2,...,R}\left\{\max_{l=1,2,...,H}\left\{\sum_{h=1}^{H} a_{r,h,l}^*(\tilde{\boldsymbol{\alpha}}_0)(\tilde{\tau}_{r,h} + \sum_{g=1}^{H+1} \tilde{\alpha}_{g,h}^{r,l} \hat{\tau}_{r,g,h}) - d_l\right\}\right\} \tag{2.93}$$

Inną, równoważną w stosunku do macierzy $\boldsymbol{a}^*(\tilde{\boldsymbol{\alpha}}_0)$, postacią rozwiązania są zbiory

$$\boldsymbol{H}_a^{r,l,*} = \{h \in \boldsymbol{H} : a_{r,h,l}^*(\tilde{\boldsymbol{\alpha}}_0) = 1\}, \ r = 1,2,...,R, \ l = 1,2,...,H. \tag{2.94}$$

Kryterium (2.93) zapisujemy jako

$$\begin{aligned}\overline{Q}_{\mathrm{L}}(\tilde{\boldsymbol{\alpha}};\tilde{\boldsymbol{\alpha}}_0) &= \max_{r=1,2,...,R}\{\max_{l=1,2,...,H}\{\sum_{h=1}^{H_a^{r,l,*}} (\bar{\tau}_{r,h}\tilde{\alpha}_{H+1,h}^{r,l}\hat{\tau}_{r,H+1,h} + \sum_{g=1}^{H_a^{r,l,*}} \tilde{\alpha}_{g,h}^{r,l}\hat{\tau}_{r,g,h}) - d_l\}\} \\ &= \max_{r=1,2,...,R}\{\overline{Q}_{\mathrm{L}}^r(\tilde{\boldsymbol{\alpha}}^r;\tilde{\boldsymbol{a}}_0)\}.\end{aligned} \tag{2.95}$$

Funkcje $\overline{Q}_{\mathrm{L}}^{r}(\tilde{\boldsymbol{\alpha}}^{r};\tilde{\boldsymbol{\alpha}}_0)$ i zbiory $\tilde{\boldsymbol{\alpha}}^{r}(\boldsymbol{a}^{*}(\tilde{\boldsymbol{\alpha}}_0))$ dla różnych r są niezależne, gdzie

$$\tilde{\boldsymbol{\alpha}}(\boldsymbol{a}^{*}(\tilde{\boldsymbol{\alpha}}_0)) \stackrel{\Delta}{=} \tilde{\boldsymbol{\alpha}}^{r,1}(\boldsymbol{a}^{*}(\tilde{\boldsymbol{\alpha}}_0))\times\tilde{\boldsymbol{\alpha}}^{r,2}(\boldsymbol{a}^{*}(\tilde{\boldsymbol{\alpha}}_0))\times\cdots\times\tilde{\boldsymbol{\alpha}}^{r,H}(\boldsymbol{a}^{*}(\tilde{\boldsymbol{\alpha}}_0))$$

jest zbiorem macierzy $\tilde{\boldsymbol{\alpha}}^{r}$. Dlatego jest możliwe niezależne wyznaczanie tras przejazdów poszczególnych realizatorów. Wspomnianą dekompozycję można formalnie zapisać jako

$$\min_{\substack{\tilde{\boldsymbol{\alpha}}^{r,l}\in\tilde{\boldsymbol{\alpha}}^{r,l}(\boldsymbol{a}^{*}(\boldsymbol{\alpha}_0))\\ r=1,2,\ldots,R\\ l=1,2,\ldots,H}} \max_{r=1,2,\ldots,R}\{\overline{Q}_{\mathrm{L}}^{r}(\tilde{\boldsymbol{\alpha}}^{r};\tilde{\boldsymbol{\alpha}}_0)\} = \max_{r=1,2,\ldots,R}\left\{\min_{\tilde{\boldsymbol{\alpha}}^{r}\in\tilde{\boldsymbol{\alpha}}^{r}(a^{*}(\tilde{\alpha}_0))} \overline{Q}_{\mathrm{L}}^{r}(\tilde{\boldsymbol{\alpha}}^{r};\tilde{\boldsymbol{\alpha}}_0)\right\}, \quad (2.96)$$

po której kryterium (2.95) przyjmuje postać

$$\begin{aligned}\overline{Q}_{\mathrm{L}}^{r}(\tilde{\boldsymbol{\alpha}}^{r};\tilde{\boldsymbol{\alpha}}_0) &= \max_{l=1,2,\ldots,H}\left\{\sum_{h=1}^{H_a^{r,l,*}}\sum_{g=1}^{\overline{H}_a^{r,l,*}}\tilde{\alpha}_{g,h}^{r,l}\hat{\tau}_{r,g,h}+\sum_{h=1}^{H_a^{r,l,*}}\overline{\tau}_{r,h}-d_l\right]\right\}\\ &= \max_{l=1,2,\ldots,H}\left\{\overline{Q}_{\mathrm{L}}^{r,l}(\tilde{\boldsymbol{\alpha}}^{r,l};\tilde{\boldsymbol{\alpha}}_0)\right\}.\end{aligned} \quad (2.97)$$

Wówczas wyznaczenie trasy przejazdu realizatora r wymaga rozwiązania problemu optymalizacyjnego

$$\min_{\substack{\tilde{\boldsymbol{\alpha}}^{r,l}\\ l=1,2,\ldots,H}}\left[\max_{l=1,2,\ldots,H}\{\overline{Q}_{\mathrm{L}}^{r,l}(\tilde{\boldsymbol{\alpha}}^{r,l};\tilde{\boldsymbol{\alpha}}_0)\}\right] \quad (2.98)$$

z ograniczeniami (2.81)–2.86) dla ustalonego r. Odmiennie niż w przypadku omówionym w poprzednim podpunkcie, nie jest to zapis problemu komiwojażera. Jest to spowodowane ograniczeniem (2.86) oraz występowaniem terminów zakończenia zadań d_l. Dlatego dalej jest proponowany przybliżony algorytm rozwiązania, polegający na wyznaczaniu fragmentów tras przejazdów realizatorów niezależnie dla poszczególnych przedziałów czasu l. Oznaczmy przez $\underline{h}^{r}(l)$ i $\overline{h}^{r}(l)$ dla $l=1,2,\ldots,H$ odpowiednio pierwsze i ostatnie zadanie wykonywane przez realizator r w przedziale l. Zgodnie z (2.86), dla $\overline{h}^{r}(l-1)$ i $\underline{h}^{r}(l)$, gdzie $l=2,3,\ldots,H-1$ musi być spełniony warunek $\tilde{\alpha}_{\overline{h}^{r}(l-1),\underline{h}^{r}(l)}^{r,l}=1$. Oczywiście $\underline{h}^{r}(1)=H+1$, $r=1,2,\ldots,R$, ponieważ każdy realizator startuje z bazy. Możemy sformułować H następujących niezależnych problemów optymalizacyjnych

$$\min_{\tilde{\boldsymbol{\alpha}}^{r,l}} \overline{Q}_{\mathrm{L}}^{r,l}(\tilde{\boldsymbol{\alpha}}^{r,l};\tilde{\boldsymbol{\alpha}}_0) \tag{2.99}$$

z ograniczeniami

$$\tilde{\alpha}_{h,h}^{r} = 0,\ h \in \boldsymbol{H}_a^{r,l,*}, \tag{2.100}$$

$$\sum_{g=1}^{H_a^{r,l,*}} \tilde{\alpha}_{g,h}^{r,l} = 1,\ h \in \boldsymbol{H}_a^{r,l,*}, \tag{2.101}$$

$$\sum_{g=1}^{H_a^{r,l,*}} \tilde{\alpha}_{g,p}^{r,l} = \sum_{h=1}^{H_a^{r,l,*}} \tilde{\alpha}_{p,h}^{r,l},\quad p \in \boldsymbol{H}_a^{r,l,*},\ p \neq \overline{h}^{r}(l), \tag{2.102}$$

$$\sum_{h=1}^{H_a^{r,l,*}} \tilde{\alpha}_{\underline{h}(l),h}^{r,l} = 1, \tag{2.103}$$

$$[\tilde{\alpha}_{g,h}^{r,l}] \in \overline{\boldsymbol{S}}_{r,l},\quad g,h \in \boldsymbol{H}_a^{r,l,*}. \tag{2.104}$$

Ograniczenia (2.100)–(2.104), w których występują stanowiska (zadania) $\underline{h}^{r}(l)$ i $\overline{h}^{r}(l)$, zastępują teraz ograniczenia (2.81)–(2.85). Ograniczenie (2.86) nie jest już aktywne, ponieważ wprowadzone problemy optymalizacyjne dotyczą pojedynczych przedziałów czasu. Problem optymalizacyjny (2.99) z ograniczeniami (2.100)––(2.104) jest analitycznym zapisem zagadnienia komiwojażera polegającego na wyznaczaniu drogi Hamiltona. Rozwiązaniem każdego z nich są macierze $\tilde{\boldsymbol{\alpha}}^{r,l,*}(\tilde{\boldsymbol{\alpha}}_0)$ i wartość kryterium jakości szeregowania.

Algorytm rozwiązania przedstawimy teraz w zwartej postaci, szacując kryterium jakości szeregowania

$$Q_{\mathrm{L}}^{*} = \tilde{Q}_{\mathrm{L}}^{*}(\boldsymbol{a}^{*};\tilde{\boldsymbol{\alpha}}^{*}) = \min_{\substack{\boldsymbol{a}\in\tilde{\boldsymbol{A}} \\ \tilde{\alpha}\in\tilde{\boldsymbol{\alpha}}(\boldsymbol{a})}} \tilde{Q}_{\mathrm{L}}(\boldsymbol{a},\tilde{\boldsymbol{\alpha}}) \leq \min_{\tilde{\alpha}\in\tilde{\boldsymbol{\alpha}}(\boldsymbol{a})} \min_{\substack{\boldsymbol{a}\in\tilde{\boldsymbol{A}} \\ \tilde{\alpha}=\boldsymbol{a}_0\in\hat{\boldsymbol{\alpha}}}} \tilde{Q}_{\mathrm{L}}(\boldsymbol{a},\tilde{\boldsymbol{\alpha}})$$

$$= \min_{\tilde{\alpha}\in\tilde{\boldsymbol{\alpha}}(\boldsymbol{a}^{*}(\tilde{\boldsymbol{\alpha}}_0))} \overline{Q}_{\mathrm{L}}(\tilde{\boldsymbol{\alpha}};\tilde{\boldsymbol{\alpha}}_0) = \max_{r=1,2,\ldots,R} \left\{ \min_{\tilde{\alpha}^{r}\in\boldsymbol{\alpha}^{r}(\boldsymbol{a}^{*}(\tilde{\alpha}))} \overline{Q}_{\mathrm{L}}^{r}(\tilde{\boldsymbol{\alpha}}^{r};\tilde{\boldsymbol{\alpha}}_0) \right\}$$

$$\leq \max_{r=1,2,\ldots,R} \left\{ \max_{l=1,2,..,R} \left\{ \min_{\tilde{\alpha}^{r,l}\in\boldsymbol{\alpha}^{r,l}(\boldsymbol{a}^{*}(\boldsymbol{\alpha}_0))} \overline{Q}_{\mathrm{L}}^{r,l}(\tilde{\boldsymbol{\alpha}}^{r,l};\tilde{\boldsymbol{\alpha}}_0) \right\} \right\} = \tilde{Q}_{\mathrm{L}}^{*}(\tilde{\boldsymbol{\alpha}}_0), \tag{2.105}$$

gdzie Q_{L}^{*} jest wartością optymalną wskaźnika jakości (2.19) oraz $\breve{\boldsymbol{\alpha}}^{r,l}(\boldsymbol{a}^{*}(\boldsymbol{\alpha}_0))$ jest zbiorem macierzy $\tilde{\boldsymbol{\alpha}}^{r,l}$, dla których obowiązują ograniczenia (2.100) – (2.104) oraz w sposób opisowy

1. Ustalamy $\tilde{\boldsymbol{\alpha}} = \tilde{\boldsymbol{\alpha}}_0 \in \hat{\boldsymbol{\alpha}}$

2. Minimalizujemy (2.92) względem $\boldsymbol{a}$ z ograniczeniami (2.78) (rozwiązujemy problem szeregowania). W wyniku optymalizacji otrzymujemy macierz $\boldsymbol{a}^{*}(\tilde{\boldsymbol{\alpha}}_0)$ lub równoważnie zbiory $\boldsymbol{H}^{r,l,*}$, a także kryterium jakości szeregowania w postaci $\bar{Q}_{\mathrm{L}}(\tilde{\boldsymbol{\alpha}};\tilde{\boldsymbol{\alpha}}_0)$.

3. Rozwiązujemy R problemów optymalizacyjnych (2.98) z ograniczeniami (2.81)–(2.86). Każdy z tych problemów rozwiązujemy niezależnie dla poszczególnych realizatorów w dwóch następujących krokach.

3.1. Wyznaczamy zadania (stanowiska) $\underline{h}^{r}(l)$ i $\bar{h}^{r}(l)$, $l = 1, 2, ..., H-1$ oraz $\underline{h}^{r}(H)$ tak, aby był spełniony warunek (2.86).

3.2. Rozwiązujemy H niezależnych problemów optymalizacyjnych (2.99) z ograniczeniami (2.100)–(2.104), czyli H niezależnych problemów komiwojażera w celu uzyskania drogi Hamiltona. Wynikiem optymalizacji w poszczególnych przedziałach są macierze $\tilde{\boldsymbol{\alpha}}^{r,l,*}(\tilde{\boldsymbol{\alpha}}_0)$, l = 1, 2, ..., H, które w sposób jednoznaczny tworzą trasy przejazdów realizatorów. Możemy również obliczyć wartość maksymalnego opóźnienia $\tilde{Q}_{\mathrm{L}}^{*}(\tilde{\boldsymbol{\alpha}}_0)$.

Proponujemy dwa sposoby realizacji kroków 3.1. i 3.2.

Sposób 1

Ustalamy l = 1 oraz $\underline{h}^{r}(l) = H+1$, a następnie w sposób iteracyjny w odwrotnej kolejności wykonujemy krok 3.1 oraz 3.2, to znaczy

A. (krok 3.2). Wyznaczamy drogę Hamiltona z ustalonym początkiem $\underline{h}^{r}(l)$.

B. (krok 3.1) Jeśli $l < H$, to przyjmujemy koniec drogi Hamiltona uzyskany w kroku A. jako początek drogi Hamiltona dla następnego kroku, to znaczy $\underline{h}^{r}(l+1) = \bar{h}^{r}(l)$, podstawiamy $l = l + 1$ i powtarzamy krok A. W przeciwnym przypadku kończymy działanie procedury.

Sposób 2

Kroki 3.1 i 3.2 są wykonywane kolejno i niezależnie. Wynikiem wykonania kroku 3.1 są początki i końce dróg Hamiltona w przedziałach l = 1, 2, ..., $H-1$ oraz tylko początek tej drogi w przedziale $l = H$. Realizujemy to w sposób następujący.

3.1.1. Ustalamy l = 1 oraz $\underline{h}^{r}(l) = H+1$.

3.1.2. Szukamy takich stanowisk $g^{*} \in \boldsymbol{H}_a^{r,l,*}$ oraz $h^{*} \in \boldsymbol{H}_a^{r,l+1,*}$ w sąsiednich przedziałach, między którymi czas przejazdu jest najkrótszy, to znaczy dla których jest spełniony warunek:

$$\hat{\tau}_{r,g^*,h^*} = \min_{\substack{g \in \boldsymbol{H}_a^{r,l,*} \\ h \in \boldsymbol{H}_a^{r,l+1,*}}} \hat{\tau}_{r,g,h}. \tag{2.106}$$

Następnie przyjmujemy $\overline{h}^r(l) = g^*$ oraz $\underline{h}^r(l+1) = h^*$.

3.1.3. Jeśli $l < H + 1$, podstawiamy $l = l + 1$ i powtarzamy krok 3.1.2. W przeciwnym przypadku kończymy krok 3.1. i rozpoczynamy realizację kroku 3.2.

Krok 3.2 polega na rozwiązaniu problemu komiwojażera w celu uzyskania dróg Hamiltona z ustalonymi początkami i końcami, czyli $\underline{h}^r(l)$ i $\overline{h}^r(l)$, $l = 1, 2, ..., H-1$ oraz ustalonym początkiem $\underline{h}^r(H)$ dla $l = H$.

Oba sposoby realizacji kroku 3. algorytmu rozwiązania mogą prowadzić oczywiście do różnych wyników. Do sprawy tej powrócimy jeszcze w podpunkcie 2.4.2. Podamy teraz analityczne oszacowanie wyniku, który można uzyskać stosując podany algorytm – niezależnie od wyboru sposobu realizacji kroku 3.

Twierdzenie 2.4

Niech Q_{L}^* będzie optymalną wartością kryterium jakości dla problemu PL. Wtedy

$$\widetilde{Q}_{\mathrm{L}}^*(\tilde{\boldsymbol{\alpha}}_0) - Q_{\mathrm{L}}^* \le \varepsilon_{\mathrm{I}} + \varepsilon_{\mathrm{II}}, \tag{2.107}$$

gdzie

$$\varepsilon_{\mathrm{I}} = \sum_{h=1}^{H} \max_{r=1,2,...,R} \left\{ \overline{\Delta} \hat{\tau}_{r,g,h} \right\}, \tag{2.108}$$

$$\varepsilon_{\mathrm{II}} = \max_{r=1,2,...,R} \left\{ \sum_{l=1}^{H} \hat{\tau}'_{r,h^{r,l,*}(1),h^{r,l,*}_{\mathrm{OPT}}(1)} \right\}, \tag{2.109}$$

przy czym

$$\hat{\tau}'_{r,h^{r,l,*}(1),h^{r,l,*}_{\mathrm{OPT}}(1)} = \begin{cases} \hat{\tau}_{r,h^{r,l,*}(1),h^{r,l,*}_{\mathrm{OPT}}(1)}, \text{ dla } h^{r,l,*}(1) \neq h^{r,l,*}_{\mathrm{OPT}}(1) \\ 0, \text{ dla } h^{r,l,*}(1) = h^{r,l,*}_{\mathrm{OPT}}(1) \end{cases}.$$

Dowód

Nieoptymalność zaproponowanego algorytmu rozwiązania jest spowodowana dwoma niezależnymi czynnikami: arbitralnym przyjęciem początkowej postaci macierzy $\tilde{\boldsymbol{\alpha}}$ oraz przybliżonym sposobem rozwiązywania problemu optymalizacyjnego (2.98). Rozpocznijmy od wykazania równości (2.108). Wartość ε_{I} jest oszacowaniem jakości rozwiązania związanym z przyjęciem macierzy $\tilde{\boldsymbol{\alpha}}_0$. Nieprawidłowe ustalenie tej macierzy może spowodować nieoptymalne określenie czasów dojazdu

$\hat{\tau}_{r,g,h}$. Znajomość wartości tych czasów jest niezbędna do rozwiązania problemu optymalizacyjnego z drugiego punktu algorytmu. Niech $\Delta\hat{\tau}_{r,g,h} \stackrel{\Delta}{=} \hat{\tau}_{r,g,h} - \hat{\tau}^{*}_{r,g,h}$, gdzie $\hat{\tau}^{*}_{r,g,h}$ jest czasem dojazdu dla rozwiązania optymalnego. Wtedy

$$\Delta\hat{\tau}_{r,g,h} \le \overline{\Delta}\,\hat{\tau}_{r,g,h} \stackrel{\Delta}{=} \max_{\substack{g=1,2,\ldots,H+1 \\ g\neq h}}\{\hat{\tau}_{r,g,h}\} - \min_{\substack{g=1,2,\ldots,H+1 \\ g\neq h}}\{\hat{\tau}_{r,g,h}\},$$

a także $\Delta\tau_{r,h} \le \overline{\Delta}\,\hat{\tau}_{r,g,h}$. Niech $\Delta Q^{r,l}_{\tilde{\alpha}_0}$ będzie wzrostem czasu działania realizatora *r* w przedziale *l* w stosunku do czasu optymalnego. Wartość ta może być oszacowana w sposób następujący

$$\Delta Q^{r,l}_{\tilde{\alpha}_0} = \sum_{h\in\overline{\boldsymbol{H}}^{r,l,*}} \Delta\tau_{r,h} \le \sum_{h\in\overline{\boldsymbol{H}}^{r,l,*}} \overline{\Delta}\,\hat{\tau}_{r,g,h}\,.$$

Stąd

$$\Delta Q_{\tilde{\alpha}_0} \stackrel{\Delta}{=} \max_{r=1,2,\ldots,R}\left\{\max_{l=1,2,\ldots,H}\left\{\Delta Q^{r,l}_{\alpha}\right\}\right\} \le \max_{r=1,2,\ldots,R}\left\{\max_{l=1,2,\ldots,H}\left\{\sum_{h\in\overline{\boldsymbol{H}}^{r,l,*}} \overline{\Delta}\,\hat{\tau}_{r,g,h}\right\}\right\}.$$

Biorąc pod uwagę fakt, że zbiory $\overline{\boldsymbol{H}}^{r,l,*}$ są rozłączne, a czasy $\hat{\tau}_{r,g,h}$ – niezależne od *l* otrzymujemy

$$\Delta Q_{\tilde{\alpha}_0} \le \sum_{h=1}^{H} \max_{r=1,2,\ldots,R}\{\overline{\Delta}\,\hat{\tau}_{r,g,h}\} = \varepsilon_{\mathrm{I}}\,.$$

Rozważmy teraz pogorszenie jakości rozwiązania spowodowane tylko nieoptymalnym sposobem rozwiązania problemu optymalizacyjnego (2.98), a dokładniej rozwiązywaniem w zamian problemu optymalizacyjnego (2.99). Niech $h^{r,l,*}(i)$ i $h^{r,l,*}_{\mathrm{OPT}}(i)$ oznaczają zadania będące *i*-tymi elementami dróg Hamiltona otrzymane w wyniku działania odpowiednio ocenianego algorytmu i algorytmu optymalnego. Wówczas, ogólnie $h^{r,l,*}(1) \neq h^{r,l,*}_{\mathrm{OPT}}(1)$. Bez straty ogólności możemy przyjąć, że $h^{r,l,*}(1) = h^{r,l,*}_{\mathrm{OPT}}(j)$. Wtedy w najgorszym przypadku do optymalnej drogi Hamiltona wystarczy dodać połączenie między stanowiskami $h^{r,l,*}_{\mathrm{OPT}}(j)$ i $h^{r,l,*}_{\mathrm{OPT}}(1)$ oraz pominąć stanowisko $h^{r,l,*}_{\mathrm{OPT}}(j)$, łącząc bezpośrednio $h^{r,l,*}_{\mathrm{OPT}}(j-1)$ z $h^{r,l,*}_{\mathrm{OPT}}(j+1)$. W rezultacie wartość kryterium jakości $\overline{Q}^{r,l}_{\mathrm{L}}\left(\tilde{\boldsymbol{\alpha}}^{r,l};\tilde{\boldsymbol{\alpha}}_0\right)$ może wzrosnąć i być oszacowana jako

$$\begin{aligned}\Delta Q_{\tilde{\alpha}} = {} & \hat{\tau}_{r,h^{r,l,*}(1),\,h^{r,j,*}_{\mathrm{OPT}}(1)} + \hat{\tau}_{r,h^{r,l,*}_{\mathrm{OPT}}(j-1),h^{r,l,*}_{\mathrm{OPT}}(j+1)} - \hat{\tau}_{r,h^{r,l,*}_{\mathrm{OPT}}(j-1),h^{r,l,*}_{\mathrm{OPT}}(j)} \\ & - \hat{\tau}_{r,h^{r,l,*}_{\mathrm{OPT}}(j),h^{r,l,*}_{\mathrm{OPT}}(j+1)} \le \hat{\tau}'_{r,h^{r,l,*}(1),h^{r,l,*}_{\mathrm{OPT}}(1)}. \end{aligned} \tag{2.110}$$

Nierówność w zależności (2.110) wynika z faktu, że suma trzech ostatnich składników jest niedodatnia ze względu na nierówność trójkąta. Wyznaczanie dróg Hamiltona dla różnych realizatorów jest niezależne. W zwiazku z tym z (2.110) wprost otrzymujemy równość (2.109).

Na koniec należy uzasadnić niezależność obu przyczyn powodujących pogorszenie jakości rozwiązania. Brak wpływu drugiej przyczyny na pierwszą jest oczywisty. Z kolei postać macierzy $\tilde{\boldsymbol{\alpha}}_0$ ma wpływ jedynie na dane problemu optymalizacji (2.98), a nie na samą procedurę optymalizacji. Ponadto oszacowanie (2.110) jest dokonane dla najgorszego przypadku i w tym sensie również nie zależy od wartości macierzy $\tilde{\boldsymbol{\alpha}}_0$. Tak więc wartość $\varepsilon_{\mathrm{II}}$ zależy jedynie od danych problemu. W konsekwencji

$$\Delta Q = \tilde{Q}_{\mathrm{L}}^{*}(\tilde{\boldsymbol{\alpha}}_0) - Q_{\mathrm{L}}^{*} = \Delta Q_{\tilde{\alpha}_0} + \Delta Q_{\tilde{\alpha}} \leq \varepsilon_{\mathrm{I}} + \varepsilon_{\mathrm{II}} .$$

■

Dla ilustracji dotychczasowych rozważań przedstawimy elementarny przykład obliczeniowy.

Przykład 2.2

Przyjmujemy dane jak w przykładzie 2.1 oraz $d_1 = 220$, $d_2 = 140$, $d_3 = 230$, $d_4 = 220$, $d_5 = 350$. Dla problemu optymalizacyjnego (2.92) nie jest konieczne określanie różnych macierzy początkowych dla różnych przedziałów czasu l. Wystarczy na przykład przyjąć

$$\tilde{\alpha}_{0,g,h}^{r,l} = \alpha_{g,h}^{r,H}, \; g = 1,2,...,H+1, \quad h,l = 1,2,...,H , \tag{2.111}$$

gdzie

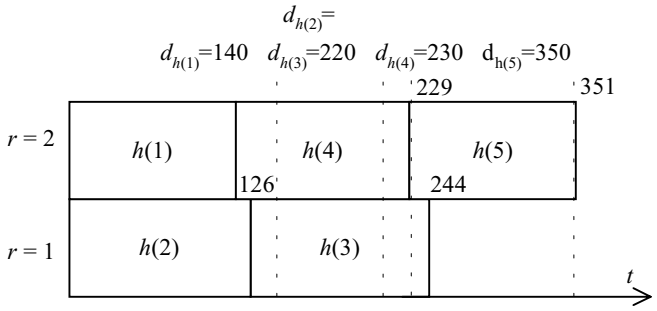
$$\tilde{\alpha}_{g,h}^{r,H} = \begin{cases} 1, \text{ jeśli } \hat{\tau}_{r,g,h} = \max\limits_{\substack{p=1,2,...,H+1 \\ p \neq h}} \{\hat{\tau}_{r,p,h}\} \\ 0, \text{ w przeciwnym przypadku} \end{cases}, \; g = 1,2,...,H+1, \; h = 1,2,...,H . \tag{2.112}$$

Postać macierzy w (2.112) jest analogiczna do postaci macierzy $\tilde{\boldsymbol{\gamma}}_0$ w (2.67). Można również ustalać $\tilde{\boldsymbol{\alpha}}_0$ analogicznie do macierzy $\tilde{\boldsymbol{\gamma}}_0$ w (2.68) lub (2.69). Wtedy czasy $B_{r,h,l}(\tilde{\alpha}_0^{r,l})$ są niezależne od l, czyli

$$B_{r,h,l}(\tilde{\alpha}_{0,g,h}^{r,l}) = B_{r,h}(\alpha_{g,h}^{r,H}), \; r = 1,2,...,R, \; g = 1,2,...,H+1, \; h,l = 1,2,...,H .$$

Biorąc pod uwagę (2.48) i (2.92) oraz porównując $\tilde{\boldsymbol{\alpha}}_0$ i $\tilde{\boldsymbol{\gamma}}_0$, zauważamy, że czasy $B_{r,h}$ są takie same jak czasy $T_{r,h}$, przedstawione w tab. 2.4. Zamieniamy numerację zadań tak, aby był spełniony warunek (2.10). Nowymi indeksami zadań będą

zmienne $h(k)$, $k = 1,2,...,5$, które odpowiadają starym indeksom w następujący sposób: $h(1) = 2$, $h(2) = 1$, $h(3) = 4$, $h(4) = 3$, $h(5) = 5$. Wówczas $d_{h(1)} = d_2 = 140$, $d_{h(2)} = d_1 = 220$, $d_{h(3)} = d_4 = 220$, $d_{h(4)} = d_3 = 230$, $d_{h(5)} = d_5 = 350$. Analogiczna zmiana indeksów powinna być dokonana dla czasów $\bar{\tau}_{r,h}$ i $\hat{\tau}_{r,g,h}$ (tab. 2.1, 2.2, 2.3). Nowe formy tych tabel nie są podawane. Rozwiązanie problemu optymalizacyjnego z drugiego kroku algorytmu, czyli macierze $\boldsymbol{a}^*(\boldsymbol{\alpha}_0)$ przedstawiono w tab. 2.16 dla $r = 1$ i w tab. 2.17 dla $r = 2$. Wartość wskaźnika jakości $\overline{Q}_{\mathrm{L}}(\tilde{\boldsymbol{\alpha}}; \tilde{\boldsymbol{\alpha}}_0)$ jest równa 24. Rozwiązanie można również przedstawić na wykresie Gantta (rys. 2.5) lub w postaci zbiorów $\boldsymbol{H}_a^{1,l,*} = \{h(2), h(3)\}$, $l = 1, 2, ..., 5$, $\boldsymbol{H}_a^{2,1,*} = \boldsymbol{H}_a^{2,2,*} = \{h(1), h(4)\}$,



Rys. 2.5. Kolejność wykonywania zadań w przykładzie 2.2
po wykonaniu drugiego kroku algorytmu

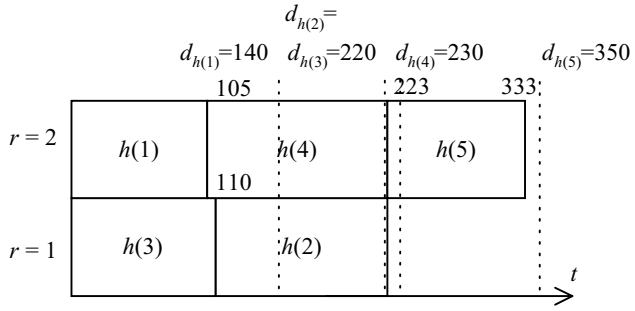
Tabela 2.16. Macierz $\boldsymbol{a}^*_{1,h(k),l}(\tilde{\boldsymbol{\alpha}}_0)$

$h(k)$ \ l	1	2	3	4	5
$h(1)$	0	0	0	0	0
$h(2)$	1	1	1	1	1
$h(3)$	1	1	1	1	1
$h(4)$	0	0	0	0	0
$h(5)$	0	0	0	0	0

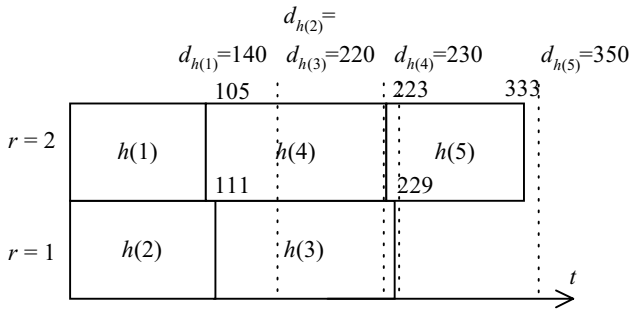
Tabela 2.17. Macierz $\boldsymbol{a}^*_{2,h(k),l}(\tilde{\boldsymbol{\alpha}}_0)$

$h(k)$ \ l	1	2	3	4	5
$h(1)$	1	1	1	1	1
$h(2)$	0	0	0	0	0
$h(3)$	0	0	0	0	0
$h(4)$	1	1	1	1	1
$h(5)$	0	0	1	1	1

$\boldsymbol{H}_a^{2,3,*} = \boldsymbol{H}_a^{2,4,*} = \boldsymbol{H}_a^{2,5,*} = \{h(1), h(4), h(5)\}$. W trzecim kroku algorytmu należy wyznaczyć macierz $\tilde{\boldsymbol{\alpha}}$. W tym celu dla ustalonych r oraz l należy określić kolejność wykonywanych zadań. Dla rozważanego prostego przykładu, więcej niż jedno zadanie jest wykonywane tylko przez realizator $r = 1$ w przedziałach $l = 1$ i $l = 2$. Okazuje się, że jeśli zadanie $h(3)$ poprzedza zadanie $h(2)$, uzyskujemy mniejszą wartość kryterium, to znaczy $\overline{Q}_{\mathrm{L}}^{*} = 3$ (rys. 2.6). W przeciwnym przypadku, gdy $h(2)$ poprzedza, $h(3)$ $\overline{Q}_{\mathrm{L}}^{*} = 9$ (rys. 2.7).



Rys. 2.6. Kolejność wykonywania zadań w przykładzie 2.2 dla przypadku $\overline{Q}_{\mathrm{L}}^{*} = 3$



Rys. 2.7. Kolejność wykonywania zadań w przykładzie 2.2 dla przypadku $\overline{Q}_{\mathrm{L}}^{*} = 9$

Obliczamy pogorszenie jakości spowodowane zastosowanym algorytmem rozwiązania, czyli sumę wartości ε_{I} i $\varepsilon_{\mathrm{II}}$. Wartość ε_{I} jest równa wartości ε_{γ} z przykładu 2.1, czyli 75. Obliczenie $\varepsilon_{\mathrm{II}}$ wymaga znajomości rozwiązania optymalnego. Do uzyskania takiego rozwiązania zastosowano przegląd zupełny. Okazało się, że optymalne trasy mają postać

$$\text{realizator } r = 1 - \text{trasa} : 6 \to 2 \to 4 \to 5,$$
$$\text{realizator } r = 2 - \text{trasa} : 6 \to 1 \to 3,$$

a optymalna wartość kryterium jakości szeregowania wynosi –1. Wówczas $\varepsilon_{II} = 56$ oraz maksymalne pogorszenie jakości $\varepsilon_I + \varepsilon_{II} = 131$.

■

2.4. Inne problemy

Omówiono trzy różne problemy, które dotyczą zagadnienia szeregowania zadań z uwzględnieniem ruchu realizatorów. Pierwszy przedstawiono algorytm dokładny rozwiązania dla kryterium w postaci długości uszeregowania. Opracowanie tego algorytmu wymagało potraktowania rozważanego kompleksu operacji jako systemu zdarzeniowo-dynamicznego.

Kolejny problem nawiązuje do rozważań z poprzedniego punktu, w którym zaprezentowano przybliżone algorytmy rozwiązania. Podano tam analityczne oszacowania jakości algorytmów. Nie umożliwiają one pełnej oceny algorytmów, między innymi zbadania wpływu danych problemu szeregowania na uzyskiwane wyniki. Dlatego ważną rolę pełnią komputerowe badania symulacyjne. Wybrane wyniki takich badań przedstawiono w punkcie 2.4.2. Bardziej istotne są jednak zawarte tam podstawy do badań symulacyjnych, które mają zastosowanie nie tylko do rozpatrywanego problemu szeregowania.

Ostatni problem dotyczy przypadku probabilistycznego dla kryterium w postaci długości uszeregowania. W rozważaniach przyjęto założenie, że czasy wykonywania czynności na stanowiskach i czasy dojazdów są realizacjami zmiennych losowych.

2.4.1. Algorytm dokładny

Algorytm dokładny omówimy dla przypadku minimalizacji długości uszeregowania. Podstawą do wyznaczania algorytmu jest problem wyjściowy PC. Dokonujemy tak zwanej dekompozycji czasowej tego problemu i uzyskujemy inny równoważny problem optymalizacyjny. Do jego rozwiązania stosujemy zasadę podziału i ograniczeń. Algorytm ten został po raz pierwszy przedstawiony w [58].

Zasada proponowanej dekompozycji czasowej problemu PC polega na podziale horyzontu czasowego, w którym są wykonywane wszystkie zadania ze zbioru $\boldsymbol{H}$ oraz zjazdy realizatorów do bazy, na mniejsze przedziały czasu i na podejmowaniu decyzji o wykonywaniu zadań i ruchu realizatorów w takich przedziałach, z uwzględnieniem skutków decyzji podjętych wcześniej. Pojęcie przedziału czasu jest w tym podejściu bardzo istotne. Wiąże się ono z innym ważnym pojęciem, a mianowicie ze zdarzeniem. Przez zdarzenie rozumiemy zakończenie wykonywania co naj-

mniej jednego zadania lub zjazdu realizatora do bazy. Moment czasu, w którym powstaje zdarzenie jest tą chwilą, w której kończy się przedział czasu z nim związany oraz kiedy musimy podjąć kolejną decyzję, dotyczącą realizowanego kompleksu operacji. Decyzje podejmowane są więc w dyskretnych momentach czasu. Inaczej możemy również powiedzieć, że decyzje są podejmowane w taktach, a długości takich taktów, czyli przedziałów czasu, są zmienne. W dalszym ciągu n będzie oznaczać kolejne zdarzenie lub kolejny takt, w którym należy podjąć decyzję, N jest natomiast całkowitą liczbą zdarzeń (taktów).

Model kompleksu operacji

Zarysowana tu koncepcja zastosowania dekompozycji czasowej polega więc na potraktowaniu kompleksu operacji, będącego obiektem podejmowania decyzji, jako systemu zdarzeniowo-dynamicznego (ewentystyczno-dynamicznego). Opis tak traktowanego obiektu przedstawimy w przestrzeni stanu. Stanem w takcie n, $n = 0,1, ..., N$ jest macierz

$$\boldsymbol{z}(n) = [z_{r,h}(n)]_{\substack{r=1,2,...,R \\ h=1,2,...,H+2}}. \tag{2.113}$$

Elementy macierzy $\boldsymbol{z}(n)$ nie mają jednolitej interpretacji. Zbiór wartości tych elementów, które leżą w pierwszych $H + 1$ kolumnach, jest następujący

$$\boldsymbol{Z}_{r,h} \stackrel{\Delta}{=} [0, \tilde{\tau}_{r,h}] \cup \{-1,-2,...,-(H+R)\}, \quad r = 1,2,...,R, \quad h = 1,2,...,H+1, \tag{2.114}$$

gdzie

$$\tilde{\tau}_{r,h} = \bar{\tau}_{r,h} + \max_{\substack{g=1,2,...,H+1 \\ g \neq h}} \{\hat{\tau}_{r,g,h}\}.$$

Wtedy, $z_{r,h}(n) \geq 0$, $r = 1,2,...,R$, $h = 1,2,...,H+1$ oznacza czas, który w takcie n pozostał do zakończenia wykonywania zadania h przez realizator r, $z_{r,h}(n) < 0$ jest natomiast wskaźnikiem informującym o kolejności zadań już wykonanych przez realizator r. Jeśli $z_{r,h_1}(n) < z_{r,h_2}(n) < 0$, gdzie $h_1, h_2 \in \boldsymbol{H}$ i $h_1 \neq h_2$, to zadanie h_1 zostało przez realizator r wykonane wcześniej niż zadanie h_2. Zbiór wartości tych elementów macierzy $\boldsymbol{z}(n)$, które leżą w ostatniej kolumnie, jest dyskretny i skończony, to znaczy

$$\boldsymbol{Z}_{H+2} \stackrel{\Delta}{=} \{1,2,...,H+1\}, \; r = 1,2,...,R. \tag{2.115}$$

Wartość elementu $z_{r,H+2}(n) \in \boldsymbol{Z}_{H+2}$, $r = 1,2,...,R$ oznacza numer stanowiska, z którego realizator r wyjechał w celu wykonania zadania realizowanego w takcie n.

Ze wzoru (2.114) wynika, że zbiory wartości dla $z_{r,h}(n)$, gdzie $h \le H+1$ mogą być różne ze względu na różne czasy $\hat{\tau}_{r,g,h}$ dojazdu realizatorów do stanowisk. Ponadto, zbiory te są ciągło-dyskretne, ale dzięki temu pozwalają one na uwzględnienie w $z_{r,h}(n)$ dla $r = 1, 2, ..., R$, $h = 1, 2, ..., H+1$ dwóch różnych informacji: o bieżących czasach realizacji zadań przez realizatory oraz o trasach pokonanych przez realizatory do taktu n włącznie. Przestrzeń stanu $\boldsymbol{Z}$ dla $\boldsymbol{z}(n)$ zapisujemy jako iloczyn kartezjański zbiorów wartości wszystkich elementów macierzy stanu, tj.

$$\begin{aligned}\boldsymbol{Z} &= \boldsymbol{Z}_{1,1} \times \boldsymbol{Z}_{1,2} \times ... \times \boldsymbol{Z}_{1,H+1} \times \boldsymbol{Z}_{H+2} \times ... \\ &... \times \boldsymbol{Z}_{R,1} \times \boldsymbol{Z}_{R,2} \times ... \times \boldsymbol{Z}_{R,H+1} \times \boldsymbol{Z}_{H+2}.\end{aligned} \tag{2.116}$$

Postać stanu początkowego $\boldsymbol{z}(0)$ wynika z warunków problemu, a zwłaszcza z założenia, że wszystkie realizatory wyruszają ze wspólnej bazy, czyli

$$\boldsymbol{z}(0) = [z_{r,h}(0)]_{\substack{r=1,2,...,R \\ h=1,2,...,H+2}}, \tag{2.117}$$

gdzie

$$z_{r,h}(0) = \begin{cases} \tilde{\tau}_{r,h}, & \text{dla } r = 1,2,...,R,\ h = 1,2,...,H+1, \\ H+1, & \text{dla } r = 1,2,...,R,\ h = H+2. \end{cases} \tag{2.118}$$

Zdefiniujmy zbiór stanów początkowych.

Definicja 2.11

Zbiór $\boldsymbol{D}_z(0) = \{\boldsymbol{z}(0) \in \boldsymbol{Z} : \text{zachodzi (2.118)}\}$ nazywamy zbiorem stanów początkowych.

■

Oczywiście jest to zbiór jednoelementowy. Z warunków problemu wynika również definicja stanu końcowego $\boldsymbol{z}(N)$. Stanem końcowym jest macierz

$$\boldsymbol{z}(N) = [z_{r,h}(N)]_{\substack{r=1,2,...,R \\ h=1,2,...,H+2}}. \tag{2.119}$$

Na elementy tej macierzy są narzucone następujące ograniczenia

$$\sum_{r=1}^{R} \boldsymbol{I}(-z_{r,h}(N)) = 1,\ h = 1,2,...,H, \tag{2.120}$$

gdzie $\boldsymbol{I}(\cdot)$ jest funkcją przyjmującą wartości 1 lub 0, to znaczy

$$\mathbf{1}(x) \stackrel{\Delta}{=} \begin{cases} 1, & \text{dla } x \geq 0, \\ 0, & \text{dla } x < 0, \end{cases} \tag{2.121}$$

$$z_{r,H+1}(N) \leq 0, \; r = 1, 2, ..., R, \tag{2.122}$$

$$z_{r,H+2}(N) = H + 1, \; r = 1, 2, ..., R. \tag{2.123}$$

Warunek (2.120) zapewnia, że każde zadanie będzie wykonane. Z sytuacją taką mamy do czynienia wtedy, gdy w każdej z H pierwszych kolumn macierzy stanu wystąpi dokładnie jedna wartość niedodatnia. Wymaganie, aby każdy realizator zjechał do bazy, jest ujęte w formie nierówności (2.122). Oznacza ona, że w $H+1$ kolumnie macierzy stanu wszystkie elementy muszą być niedodatnie, a dokładniej – całkowite niedodatnie. Ograniczenie (2.123) oznacza, że baza jest tym miejscem, w którym powinny znajdować się realizatory po zakończeniu pracy.

Definicja 2.12

Zbiór $\boldsymbol{D}_z(N) = \{\boldsymbol{z}(N) \in \boldsymbol{Z} : \text{zachodzi (2.120)} \wedge \text{(2.122)} \wedge \text{(2.123)}\}$ nazywamy zbiorem stanów końcowych.

■

Przejdziemy teraz do sformalizowania decyzji, które mają być podejmowane w każdym takcie n. Tworzą one macierz decyzji $\boldsymbol{v}(n)$, określoną jako

$$\boldsymbol{v}(n) = [v_{r,h}(n)]_{\substack{r=1,2,...,R \\ h=1,2,...,H+1}}. \tag{2.124}$$

Elementy tej macierzy, w zależności od swojej wartości, rozstrzygają o zadaniach wykonywanych przez realizatory w takcie n, a więc wartość $v_{r,h}(n)$, $r = 1, 2, ..., R$, $h = 1, 2, ..., H+1$ daje informację o wykonywaniu zadania h przez realizator r. Macierz $\boldsymbol{v}(n)$ jest macierzą binarną, to znaczy

$$v_{r,h}(n) \in \{0,1\} = \begin{cases} 1, & \text{jeśli w takcie } n \text{ realizator } r \text{ wykonuje zadanie } h, \\ 0, & \text{w przeciwnym razie.} \end{cases} \tag{2.125}$$

Na decyzje $\boldsymbol{v}(n)$ nakładamy ograniczenia

$$\sum_{r=1}^{R} v_{r,h}(n) \leq 1, \; h = 1, 2, ..., H+1, \tag{2.126}$$

$$\sum_{h=1}^{H+1} v_{r,h}(n) \le 1, \quad r = 1,2,...,R, \tag{2.127}$$

$$0 < z_{r,h}(n) < \bar{\tau}_{r,h} + \hat{\tau}_{r,z_{r,H+2}(n),h} \rightarrow v_{r,h}(n) = 1,$$
$$r = 1, 2, ..., R, \quad h = 1, 2, ..., H+1. \tag{2.128}$$

Zależności (2.126) i (2.127) zapewniają, że w każdym takcie odpowiednio każde zadanie może być wykonywane przez co najwyżej jeden realizator oraz każdy realizator wykonuje najwyżej jedno zadanie. Implikacja (2.128) jest warunkiem niepodzielności zadań. Podane ograniczenia pozwalają sformułować definicję.

Definicja 2.13

Zbiór $\boldsymbol{D}_v(\boldsymbol{z}(n)) = \{\boldsymbol{v}(n) \in \{0,1\}: \text{zachodzi (2.126)} \wedge \text{(2.127)} \wedge \text{(2.128)}\}$ nazywamy zbiorem decyzji dopuszczalnych w takcie n.

■

Należy zauważyć, że zbiór decyzji określony w definicji 2.13 zależy od aktualnego stanu $\boldsymbol{z}(n)$. Możemy teraz określić równania stanu. Wpierw podamy ogólne równanie, odpowiadające elementom z pierwszych $H + 1$ kolumn macierzy stanu. Ma ono postać równania różnicowego

$$\begin{aligned} & z_{r,h}(n+1) = [1 - \boldsymbol{\delta}(z_{r,h}(n) - \tilde{\tau}_{r,h})] \\ & \cdot [z_{r,h}(n) - v_{r,h}(n)\Delta(n) - \boldsymbol{1}(v_{r,h}(n)\Delta(n) - z_{r,h}(n))] \\ & + [1 - \boldsymbol{\delta}(h - z_{r,H+2}(n))]\boldsymbol{\delta}(z_{r,h}(n) - \tilde{\tau}_{r,h}) \\ & \cdot [[1 - v_{r,h}(n)]\tilde{\tau}_{r,h} + v_{r,h}(n)[\bar{\tau}_{r,h} + \hat{\tau}_{r,z_{r,H+2}(n),h} - \Delta(n)]], \\ & n = 0,1,...,N-1, \end{aligned} \tag{2.129}$$

gdzie $\boldsymbol{\delta}(\cdot)$ jest funkcją przyjmującą wartości 1 lub 0, a mianowicie

$$\boldsymbol{\delta}(x) \stackrel{\Delta}{=} \begin{cases} 1, \text{ dla } x = 0 \\ 0, \text{ dla } x \neq 0 \end{cases} \tag{2.130}$$

oraz $\Delta(n)$ jest długością taktu n, czyli czasem, który upłynął od poprzedniego zdarzenia, zdefiniowaną jako

$$\Delta(n) \stackrel{\Delta}{=} \min_{\substack{s=1,2,...,R \\ p=1,2,...,H+1}} \{z_{s,p}(n) : v_{s,p}(n) = 1\}. \tag{2.131}$$

Drugie równanie stanu, również w formie równania różnicowego, odpowiada elementom z ostatniej kolumny macierzy stanu, to znaczy

$$z_{r,H+2}(n+1) = z_{r,H+2}(n)\boldsymbol{\delta}\left(\sum_{h=1}^{H+1} h\, v_{r,h}(n)\boldsymbol{\delta}(z_{r,h}(n)-\Delta(n))\right)$$

$$+\sum_{h=1}^{H+1} h\, v_{r,h}(n)\boldsymbol{\delta}(z_{r,h}(n)-\Delta(n)),\ \ n=0,1,2,...,N-1. \tag{2.132}$$

Równanie (2.129) zapewnia prawidłowe obliczanie nowych wartości zmiennych stanu w kolejnych taktach. Składa się ono z dwóch głównych składników. Istotną rolę pełnią w nim: różnica $z_{r,h}(n)-\tilde{\tau}_{r,h}$ oraz $v_{r,h}(n)$. Ważne jest, czy różnica ta jest równa zeru, czy też nie, oraz czy element $v_{r,h}(n)$ macierzy decyzji jest równy 1, czy 0. Są możliwe cztery przypadki, w których $z_{r,h}(n+1)$ będzie przyjmowała różne wartości. W dwóch pierwszych przypadkach $z_{r,h}(n)-\tilde{\tau}_{r,h}\neq 0$, co oznacza, że do taktu *n* włącznie zostało rozpoczęte wykonywanie zadania *h* przez realizator *r*. Wtedy wartość odpowiedniej funkcji $\boldsymbol{\delta}(\cdot)$ jest równa zeru i drugi główny składnik (2.129) jest również równy zeru, a $z_{r,h}(n+1)$ otrzymuje nową wartość zgodnie z postacią pierwszego składnika.

Przypadek 1. dotyczy sytuacji, gdy $v_{r,h}(n)=1$, to znaczy, gdy zadanie *h* jest w trakcie wykonywania przez realizator *r*. W sytuacji tej rozpatrywana zmienna stanu w następnym takcie będzie miała wartość mniejszą o długość taktu $\Delta(n)$. Taki charakter zmniejszania się $z_{r,h}(n)$ trwa dopóty, dopóki jest ona dodatnia (tzn. funkcja $\boldsymbol{1}(\cdot)$ ma wartość zero).

Jeśli osiąga ona wartość zero, czyli zadanie *h* jest zakończone, to $v_{r,h}(n)=0$ i zachodzi przypadek 2. Wtedy $z_{r,h}(n)$ zmniejsza swoją wartość, aż do zakończenia całego procesu decyzyjnego w ten sposób, że w kolejnych taktach są odejmowane wartości jeden. Zmienna $z_{r,h}(n+1)$ jest ujemna i zawiera informację wykorzystywaną do określania aktualnej trasy przejazdu realizatora *r*. Niedodatnie elementy macierzy stanu, leżące w *r*-tym wierszu, oznaczają zadania wykonane przez realizator *r*, natomiast mniejsza wartość takiego elementu oznacza wcześniejsze wykonanie zadania. Dwa następne przypadki zachodzą wtedy, gdy $z_{r,h}(n)-\tilde{\tau}_{r,h}=0$, co oznacza, że w takcie *n* może się dopiero rozpocząć wykonywanie zadania *h* przez realizator *r*. Wtedy pierwszy główny składnik (3.17) jest zerowy i istotny jest tylko drugi.

Z przypadkiem 3 mamy do czynienia wtedy, gdy $v_{r,h}(n)=1$. W tej sytuacji w następnym takcie zmienna stanu $z_{r,h}(n+1)$ będzie miała wartość mniejszą o $\Delta(n)$

od wartości początkowej równej sumie czasu wykonywania zadania na stanowisku i czasu dojazdu.

Przypadek 4 zachodzi, gdy $v_{r,h}(n) = 0$, co oznacza brak przyporządkowania realizatora *r* do wykonywania zadania *h*. Wtedy wartość $z_{r,h}(n+1)$ w następnym takcie pozostaje bez zmian, tzn. jest równa $z_{r,h}(n)$. Drugi główny składnik (2.129) zawiera ponadto zabezpieczenie przed nadawaniem nowych wartości tym zmiennym stanu, które odnoszą się do stanowiska, z którego aktualnie wyruszył realizator. Równanie (2.132) umożliwia bieżące uaktualnianie informacji o stanowisku, w którym ostatnio przebywał realizator, czyli o ostatnim elemencie jego trasy. Informacja taka jest potrzebna, aby odpowiednio ustalać czasy dojazdów realizatorów. Jest ona wykorzystywana w tym celu w równaniu stanu (2.129). Zmienna stanu $z_{r,H+2}(n+1)$ przyjmuje nową wartość *h* tylko wtedy, gdy w bieżącym takcie realizator *r* kończy wykonywanie zadania *h*. W przeciwnym przypadku $z_{r,H+2}(n+1) = z_{r,H+2}(n)$.

Analiza równań stanu (2.129) i (2.132) wskazuje, że nie są one niezależne i należy je traktować łącznie. Dlatego proponujemy łączny zapis macierzowy. Przedstawimy wpierw (2.129) i (2.132) w postaci ogólnej

$$\begin{aligned} &z_{r,h}(n+1) \stackrel{\Delta}{=} \varphi_{r,h}(\boldsymbol{z}(n), \boldsymbol{v}(n)),\ r = 1,2,...,R,\ h = 1,2,...,H+2, \\ &n = 0,1,...,N-1, \end{aligned} \tag{2.133}$$

gdzie funkcje $\varphi_{r,h}$ są zdefiniowane przez (2.129) lub (2.132). Niech $\boldsymbol{\varphi}$ oznacza macierz funkcji

$$\boldsymbol{\varphi} \stackrel{\Delta}{=} \begin{bmatrix} \varphi_{1,1},\ldots,\varphi_{1,H+2} \\ \vdots \\ \varphi_{R,1},\ldots,\varphi_{R,H+2} \end{bmatrix}. \tag{2.134}$$

Wtedy

$$\boldsymbol{z}(n+1) = \boldsymbol{\varphi}(\boldsymbol{z}(n), \boldsymbol{v}(n)), \quad n = 0,1,...,N-1 \tag{2.135}$$

dla danego stanu początkowego $\boldsymbol{z}(0)$ oraz $\boldsymbol{v}(n) \in \boldsymbol{D}_v(\boldsymbol{z}(n))$ jest modelem rozpatrywanego kompleksu operacji, przedstawionym w postaci macierzy równań różnicowych.

Sformułowanie problemu

Naszym celem jest takie kształtowanie tras poruszania się realizatorów między stanowiskami, na których są wykonywane zadania, aby łączny czas wykonania wszystkich zadań oraz zjazdów realizatorów do bazy był jak najkrótszy. Cel ten może być osiągnięty, jeśli będziemy minimalizować wartość kryterium

$$\hat{Q}_N(\bar{\boldsymbol{v}}(N-1)) = \sum_{n=0}^{N(n)-1} \varphi_0(\boldsymbol{z}(n), \boldsymbol{v}(n)), \tag{2.136}$$

gdzie $\varphi_0(\boldsymbol{z}(n), \boldsymbol{v}(n)) \stackrel{\Delta}{=} \Delta(n)$,

$$\begin{aligned} &N(n) = N(n-1) + 1 - \sum_{r=1}^{R} \sum_{h=1}^{H+1} [\boldsymbol{1}(z_{r,h}(n)) - \boldsymbol{1}(z_{r,h}(n+1))], \\ &n = 0,1,...,N-1, \;\; N(-1) = H + R, \end{aligned} \tag{2.137}$$

$\bar{\boldsymbol{v}}(N-1) = (\boldsymbol{v}(0), \boldsymbol{v}(1),...,\boldsymbol{v}(N-1))$ – ciąg macierzy decyzyjnych. (2.138)

Kryterium jakości (2.136) jest sumą długości taktów $\Delta(n)$. Ponieważ w jednym takcie może się zakończyć wykonywanie więcej niż jednego zadania, więc horyzont podejmowania decyzji jest zmienny, tzn. liczba taktów niekoniecznie musi wynosić $H + R$. Aby pozbyć się tej niedogodności i używać kryterium jakości ze stałym horyzontem, wprowadzamy dodatkowe ograniczenia dotyczące trajektorii stanu. Dotychczas sformułowaliśmy ograniczenia dotyczące stanu początkowego i stanu końcowego, a stany pośrednie nie były ograniczane. Teraz wprowadzenie takich ograniczeń zapobiegnie ujmowaniu w obrębie jednego taktu zakończenia wykonywania więcej niż jednego zadania. Jeśli wystąpi taka sytuacja, że faktycznie w tym samym momencie kończy się wykonywanie więcej niż jednego zadania, to będą tworzone następne takty o długości $\Delta(n) = 0$, a więc liczba wszystkich taktów N będzie zawsze równa $H + R$. Ograniczenia, o których mowa, sformułujemy w postaci zbiorów stanów dopuszczalnych $\boldsymbol{D}_z(n + 1)$, $n = 0, 1, ..., N - 2$.

Definicja 2.14

Zbiór

$$\boldsymbol{D}_z(n+1) = \left\{ \boldsymbol{z}(n+1) = \varphi(\boldsymbol{z}(n), \boldsymbol{v}(n)) \in \boldsymbol{Z} : \sum_{r=1}^{R} \sum_{h=1}^{H+1} \left[\boldsymbol{1}(z_{r,h}(n)) - \boldsymbol{1}(z_{r,h}(n+1)) \right] = 1 \right\},$$

nazywamy zbiorem stanów dopuszczalnych w takcie n.

■

W dalszym ciągu będziemy stosować kryterium jakości w postaci

$$Q_N(\bar{\boldsymbol{v}}(N-1)) = \sum_{n=0}^{N-1} \varphi_0(\boldsymbol{z}(n), \boldsymbol{v}(n)). \tag{2.139}$$

Możemy teraz sformułować problem szeregowania, z uwzględnieniem ruchu realizatorów po dekompozycji czasowej.

Dane są:

1. Zbiory $\overline{\boldsymbol{H}}$ i $\boldsymbol{R}$ oraz macierze czasów $\bar{\boldsymbol{\tau}}$ i $\hat{\boldsymbol{\tau}}$ tak, jak to określono w danych problemu PC w punkcie 2.2.

2. Model $\boldsymbol{z}(n+1) = \boldsymbol{\varphi}(\boldsymbol{z}(n), \boldsymbol{v}(n)),\ n = 0,1,..., N-1$ oraz stan początkowy $\boldsymbol{z}(0)$.

3. Ograniczenia na trajektorię stanu $\boldsymbol{D}_z(n)$, n = 1, 2, ..., N.

Należy wyznaczyć ciąg dopuszczalnych macierzy decyzyjnych $\bar{\boldsymbol{v}}(N-1)$, gdzie $\boldsymbol{v}(n) \in D_v(z(n))$ tak, aby minimalizować $Q_N(\bar{\boldsymbol{v}}(N-1))$.

Po analizie problemu wyjściowego PC oraz określonego problemu po dekompozycji czasowej możemy sformułować twierdzenie zawierające porównanie obu problemów.

Twierdzenie 2.5

Sformułowany w niniejszym punkcie problem szeregowania z ruchomymi realizatorami po dekompozycji czasowej jest równoważny w sensie kryteriów jakości szeregowania i ograniczeń problemowi PC.

Dowód

Przede wszystkim równoważne są kryteria jakości (2.9) i (2.139). Wynika to wprost z ich interpretacji, ponieważ wartości obu z nich są czasami wykonania zbioru zadań oraz zjazdów realizatorów do bazy. Formalnie, łatwo jest przekształcić kryterium (2.9) do postaci (2.137). Znajomość macierzy $\boldsymbol{\gamma}$ umożliwia obliczenie momentów zakończenia wykonywania zadań. Biorąc, dla danego r, wszystkie elementy $\gamma_{r,g,h} = 1$, potrafimy utworzyć trasę dla realizatora r (jej dopuszczalność jest zagwarantowana przez ograniczenia) i w konsekwencji obliczyć C_h^r, czyli momenty zakończenia wykonywania zadań h przez realizatory r. Niech $\boldsymbol{H}_r$ będzie ciągiem zadań o początku i końcu w bazie, o liczności H_r wykonywanych przez realizator r i tworzących trasę dla tego realizatora, to znaczy

$$\boldsymbol{H}_r = (h_r(1) = H+1, h_r(2), ..., h_r(H_r) = H+1) = (h \in \overline{\boldsymbol{H}} : \gamma_{r,g,h} = 1).$$

Wtedy

$$C^r_{h_r(i)} = C^r_{h_r(i-1)} + \hat{\tau}_{r,h_r(i-1),h_r(i)} + \bar{\tau}_{r,h_r(i)},\ i = 2,3, ..., H_r$$

oraz $C_0^r = 0$. Po uporządkowaniu w kolejności niemalejących wartości momenty C_h^r dla wszystkich r tworzą ciąg $\hat{C}_n$, $n = 1,2, ..., N = H + R$. Wówczas $\hat{C}_n - \hat{C}_{n-1} = \Delta(n)$, n = 1, 2, ..., N, gdzie $\hat{C}_0 = 0$. Stąd wprost otrzymujemy (2.139). W podobny sposób

można wykazać równoważność ograniczeń. Ze względu na oczywistość porównywanych własności, nie będziemy niepotrzebnie komplikować zapisu i dalsze porównanie przedstawimy w sposób opisowy. W obu problemach należy wyznaczyć trasy poruszania się realizatorów o początkach i końcach w bazie. W problemie PC sprowadza się to do wyznaczenia wartości elementów trójwymiarowej macierzy $\boldsymbol{\gamma}$, a spełnienie wymienionego wymagania, nałożonego na trasy, jest zapewnione przez ograniczenia (2.7) i (2.6). Dla problemu po dekompozycji czasowej określane w kolejnych taktach macierze decyzyjne w sposób jednoznaczny wyznaczają takie trasy, a ich przebieg jest zapamiętywany w stanie $\boldsymbol{z}(n)$, a dokładniej – w tych jego elementach, które przyjmują wartości niedodatnie. Wymaganie o początku i końcu trasy każdego realizatora w bazie jest spełnione przez odpowiednią definicję stanu początkowego i stanu końcowego – dokładnie przez wyrażenia (2.118) i (2.122).

Przeanalizujemy teraz pozostałe ograniczenia, obowiązujące w problemie PC, i zauważymy, że są one spełnione również w drugim rozważanym problemie optymalizacyjnym. Równania (2.5) wymagają, aby każde zadanie było wykonane. Po dekompozycji czasowej jest to zapewnione przez wyrażenie (2.120). Z kolei zależność (2.6) zawiera wymaganie, aby wyznaczone trasy realizatorów były ciągłe. Po dekompozycji czasowej ciągłość tras wynika przede wszystkim z bieżącego ich tworzenia, ale również znajduje odzwierciedlenie w postaci równań stanu, w których zapamiętuje się stanowisko, gdzie realizator ostatnio wykonywał zadanie (zmienne stanu $z_{r,H+2}$). Informacja taka jest wykorzystywana w równaniu (2.129) do określania czasów przejazdów realizatora, rozpoczynającego wykonywanie następnego zadania. Wymagania, aby każdy realizator dokładnie jeden raz wyruszył z bazy oraz aby wyznaczone trasy nie tworzyły podcykli, które w problemie PC są wyrażone w postaci (2.7) i (2.8), nie występują wprost w sformułowaniu problemu po dekompozycji czasowej. Ich spełnienie wynika jednak z istoty proponowanej metody. Jak już wspomnieliśmy, jest wymagane, aby trasy realizatorów rozpoczynały się i kończyły się w bazie. Gdyby założyć na chwilę, że pewien realizator w trakcie swojej pracy przybywa do bazy, by ją po pewnym czasie opuścić i powrócić na stałe dopiero po zakończeniu wykonywania pozostałych zadań, wówczas z tego wynikałoby, że taka trasa nie jest optymalna, ponieważ zjazd do bazy nie wiąże się z wykonaniem jakiegokolwiek zadania i powoduje tylko zwiększenie czasu pracy tego realizatora. Sytuacja z wielokrotnymi zjazdami do bazy mogłaby wystąpić tylko wtedy, gdyby macierze czasów przejazdów nie były pełne [98], ale z taką sytuacją nie mamy do czynienia. Jest to jednocześnie uzasadnienie na brak podcykli dla tras w drugim problemie optymalizacyjnym.

■

Opis algorytmu

Do rozwiązania problemu postawionego w poprzednim punkcie zastosujemy sposób oparty na zasadzie podziału i ograniczeń. Będziemy się posługiwać geometryczną interpretacją tej metody, w której proces dochodzenia do rozwiązania optymalnego jest przedstawiany w formie poruszania się w grafie, będącym drzewem. Wierzchołki w takim drzewie reprezentują podzbiory rozwiązań możliwe do uzyskania w poddrzewie, w którym dany wierzchołek jest wierzchołkiem początkowym. Każdy wierzchołek końcowy jest skojarzony dokładnie z jednym rozwiązaniem. Poszukiwanym przez nas rozwiązaniem są trasy poruszania się realizatorów wykonujących zadania. Poruszanie się w drzewie od wierzchołka początkowego oznacza tworzenie tras dla realizatorów. Im bliżej wierzchołka końcowego się znajdujemy, tym mniej liczne są zbiory możliwych tras dla realizatorów. Wierzchołki w drzewie są ułożone w warstwy i możemy się poruszać tylko między wierzchołkami należącymi do sąsiednich warstw. Numer warstwy jest jednocześnie numerem taktu. Wierzchołek początkowy leży w warstwie $n = 0$, wierzchołki końcowe natomiast – w warstwie $n = N$. Przejście do następnego wierzchołka w drzewie oznacza zmianę stanu kompleksu operacji (obiektu). Stanem tego obiektu jest macierz $\boldsymbol{z}(n)$. Jeśli posługujemy się interpretacją geometryczną i mówimy, że obiekt znajduje się w określonym wierzchołku warstwy n, $n = 0, 1, ..., N$, to oznacza, że jest dla niego określony stan $\boldsymbol{z}(n)$ i odwrotnie – tzn. stanowi $\boldsymbol{z}(n)$ odpowiada określony wierzchołek w drzewie rozwiązań. Przejściom między wierzchołkami odpowiada zmiana stanu obiektu. Na przykład przejście od pewnego wierzchołka, leżącego w warstwie n, któremu odpowiada stan $\boldsymbol{z}(n)$, do innego dopuszczalnego wierzchołka z warstwy następnej wiąże się z koniecznością obliczenia stanu $\boldsymbol{z}(n+1)$ dla tego wierzchołka. Posługujemy się przy tym równaniami stanu (2.135). Dla opisywanego przejścia wybór, spośród wielu możliwości konkretnego wierzchołka w warstwie $n+1$, oznacza nadanie wartości macierzy decyzyjnej $\boldsymbol{v}(n)$ i odwrotnie – tzn. ustalona postać tej macierzy determinuje wierzchołek z warstwy następnej, do której następuje przejście. W każdym wierzchołku jest liczona funkcja dolnych ograniczeń, będąca oszacowaniem czasu realizacji tras, które mogą być utworzone po opuszczeniu wierzchołka.

Wprowadźmy oznaczenia. Niech $\overline{\boldsymbol{H}}'(\boldsymbol{z}(n))$ oznacza zbiór zadań nie rozpoczętych do taktu n włącznie, o liczności $\overline{H}'(\boldsymbol{z}(n))$, a $\boldsymbol{R}'(\boldsymbol{z}(n))$ – zbiór realizatorów, którym w takcie n można przydzielić następne zadanie do wykonania o liczności $R'(\boldsymbol{z}(n))$. Do zbioru $\boldsymbol{R}'(\boldsymbol{z}(n))$ należą wszystkie realizatory, które aktualnie nie wykonują zadań. Bieżąca, czyli n-ta warstwa drzewa rozwiązań, jest scharakteryzowana zbiorem $\boldsymbol{L}(n)$ dopuszczalnych podzbiorów rozwiązań, czyli wierzchołków, o licz-

ności $L(n)$. Element tego zbioru, czyli $l(n)$, oznacza konkretny wierzchołek lub inaczej podzbiór rozwiązań. Dla warstwy następnej, czyli $n + 1$, podobnie określamy zbiór $\boldsymbol{M}(l(n))$ następników wierzchołka (podzbioru rozwiązań) $l(n) \in \boldsymbol{L}(n)$, o liczności $M(l(n))$. Każdy element tak wprowadzonego zbioru jest oznaczony przez $m(l(n))$. Jest oczywiste, że każdy zbiór $\boldsymbol{M}(l(n))$ jest podzbiorem $\boldsymbol{L}(n+1)$ dla $n = 0,1, ..., N-1$. Przez $t(l(n))$ oznaczamy czas osiągnięcia podzbioru rozwiązań $l(n)$, czyli czas dojścia do wierzchołka $l(n)$.

Najważniejszymi elementami omawianej metody rozwiązania są procedury generowania następników i obliczania dolnych ograniczeń wartości kryterium.

Generowanie następników

Załóżmy, że są dane: stan $\boldsymbol{z}(n)$ oraz zbiór decyzji dopuszczalnych $\boldsymbol{D}_v(n)$, gdzie $n = 0, 1, ..., N - 1$. Znajomość stanu $\boldsymbol{z}(n)$ oznacza, że znajdujemy się w wierzchołku $l(n)$ drzewa rozwiązań. Liczbę $L(n+1)$ wierzchołków w warstwie $n+1$ określamy w sposób rekurencyjny na podstawie znajomości liczby $M(l(n))$ następników dla węzła $l(n)$. Wartość $M(l(n))$ jest funkcją liczby $\overline{H}'(\boldsymbol{z}(n))$ wolnych zadań i liczby $R'(\boldsymbol{z}(n))$ wolnych realizatorów, to znaczy

$$M(l(n)) = \prod_{i=0}^{R'(\boldsymbol{z}(n))-1} [\overline{H}'(\boldsymbol{z}(n)) - i]. \tag{2.140}$$

Wtedy

$$L(n+1) = \sum_{l(n)=1}^{L(n)} M(l(n)), \; n = 1, 2, ..., N-1, \tag{2.141}$$

natomiast

$$L(1) = \prod_{i=0}^{R-1} (\overline{H} - i) = M(l(0)). \tag{2.142}$$

Zbiory $\boldsymbol{R}'(\boldsymbol{z}(n))$ i $\overline{\boldsymbol{H}}'(\boldsymbol{z}(n))$ oraz ich liczności określamy na podstawie znajomości stanu $\boldsymbol{z}(n)$ i zbioru $\boldsymbol{D}_v(n)$, a mianowicie

$$\overline{\boldsymbol{H}}'(\boldsymbol{z}(n)) = \left\{ h \in \overline{\boldsymbol{H}} : z_{r,h}(n) = \tilde{\tau}_{r,h}, \; r = 1, 2, ..., R \right\}, \tag{2.143}$$

$$\boldsymbol{R}'(\boldsymbol{z}(n)) = \left\{ r \in \boldsymbol{R} : \left(\forall \, h \in \overline{\boldsymbol{H}} \right) v_{r,h}(n) = 0 \text{ dla } \boldsymbol{v}(n) \in \boldsymbol{D}_v(\boldsymbol{z}(n)) \right\}. \tag{2.144}$$

Korzystając z podanych oznaczeń i zależności, procedurę generowania następników wierzchołka $l(n)$, gdzie $l(n) = 1, 2, ..., L(n)$ i n = 1, 2, ..., N – 1, możemy przedstawić następująco:

1. Ustalamy postaci oraz liczności zbiorów $\boldsymbol{R}'(\boldsymbol{z}(n))$ i $\overline{\boldsymbol{H}}'(\boldsymbol{z}(n))$ – na podstawie (2.143) i (2.144)).

2. Obliczamy $M(l(n))$, czyli liczbę następników wierzchołka $l(n)$ – na podstawie (2.140)).

3. Dla $m(l(n)) = 1, 2, ..., M(l(n))$:

a) ustalamy macierz $\boldsymbol{v}(n) \in \boldsymbol{D}_v(\boldsymbol{z}(n))$, której elementy $v_{r(m(l(n))),h(m(l(n)))}(n)$ są równe 1 dla $r(m(l(n))) \in \boldsymbol{R}'(\boldsymbol{z}(n))$ i $h(m(l(n))) \in \overline{\boldsymbol{H}}'(\boldsymbol{z}(n))$,

b) na podstawie (2.135) obliczamy $\boldsymbol{z}(n+1)$ dla $\boldsymbol{v}(n)$ ustalonego w p. 3.a,

c) obliczamy czas dojścia do wierzchołka, będącego następnikiem $l(n)$, z wyrażenia

$$t(m(l(n))) = t(l(n+1)) = t(l(n)) + \Delta(n), \tag{2.145}$$

gdzie $\Delta(n)$ jest określone przez (2.131)).

Kolejne elementy $m(l(n))$ zbioru następników, określane w p. 3. procedury, odpowiadają kolejnym przyporządkowaniom realizatorów ze zbioru $\boldsymbol{R}'(\boldsymbol{z}(n))$ do zadań ze zbioru $\overline{\boldsymbol{H}}'(\boldsymbol{z}(n))$.

Wyznaczanie dolnych ograniczeń

Dla każdego podzbioru rozwiązań $l(n)$ z warstw od pierwszej do $N-1$ obliczamy dolne ograniczenie $Q_{N,LB}(l(n))$ wartości kryterium (2.139). Stosujemy przy tym relaksację, polegającą na przyjęciu założenia, że zadania dotąd nie zrealizowane będą wykonane w możliwie najkrótszym czasie, zarówno ze względu na czasy dojazdów $\hat{\tau}_{r,g,h}$, jak i czasy wykonywania czynności na stanowiskach $\bar{\tau}_{r,h}$. Wiąże się to z pominięciem założenia o ciągłości tras realizatorów. Stąd

$$Q_{N,LB}(l(n)) = t(l(n)) + \frac{1}{R} \sum_{h \in \overline{\boldsymbol{H}}'(z(n))} \min_{r=1,2,...,R} \left\{ \bar{\tau}_{r,h} + \min_{\substack{g \in \overline{\boldsymbol{H}}'(z(n)) \\ g \neq h}} \{\hat{\tau}_{r,g,h}\} \right\}. \tag{2.146}$$

Korzystanie z dolnych ograniczeń w poszczególnych wierzchołkach drzewa rozwiązań wymaga znajomości wartości kryterium jakości dla pierwszego dopuszczalnego rozwiązania. Rozwiązanie takie w proponowanej metodzie znajdujemy w ten sposób, że w każdej warstwie – spośród wszystkich następników podzbioru rozwiązań $l(n)$ – wybieramy tylko jeden następnik, zapewniający minimalną wartość czasu $t(m(l(n)))$, to znaczy

$$\min_{m(l(n)) \in \boldsymbol{M}(l(n))} \{t(m(l(n)))\} = \min_{m(l(n)) \in \boldsymbol{M}(l(n))} \{t(l(n)) + \Delta(n)\}$$
$$= t(l(n)) + \min_{m(l(n)) \in \boldsymbol{M}(l(n))} \{\Delta(n)\}. \tag{2.147}$$

W rezultacie otrzymujemy jeden podzbiór rozwiązań w warstwie następnej, czyli $l(n+1)$, a dla $n = N-1$ – konkretne rozwiązanie, oraz wartość kryterium jakości $t(l(N))$, będącą jednocześnie wartością dotychczas najlepszego rozwiązania, którą dalej będziemy oznaczać jako $\tilde{Q}_N$. W tak ograniczonym procesie generacji następników wierzchołków $l(n)$ punkt 3, poprzednio określonej procedury, składa się tylko z jednej iteracji dla określonego w (2.147) następnika $m(l(n))$, który umożliwia jednoznaczne wyznaczenie macierzy $\boldsymbol{v}(n)$, stanu $\boldsymbol{z}(n+1)$ oraz czasu $t(m(l(n)))$.

Jak wiadomo, znajdowanie rozwiązania optymalnego na podstawie metody podziału i ograniczeń polega na kolejnym sprawdzaniu wszystkich podzbiorów rozwiązań, co prowadzi albo do polepszenia dotychczas najlepszego rozwiązania, albo do pominięcia rozpatrywanego podzbioru rozwiązań, ponieważ nie zawiera on rozwiązania lepszego od dotychczas znalezionego. Procedura kończy się, gdy wszystkie podzbiory rozwiązań (wierzchołki drzewa rozwiązań) są sprawdzone. W celu minimalizacji wielkości pamięci, niezbędnej do uzyskania rozwiązania, zastosowano procedurę w głąb sprawdzania wierzchołków drzewa rozwiązań. Przedstawia się ona następująco:

1. Uzyskujemy pierwsze rozwiązanie dopuszczalne i oznaczamy drogę do niego jako $l(n)=1$, $n=1,2,...,N$, obliczamy $M(l(n))$, $n=0,1,...,N-1$ oraz zapamiętujemy, odpowiadające otrzymanemu rozwiązaniu: wartość kryterium jakości jako $\tilde{Q}_N$, stan końcowy jako $\tilde{\boldsymbol{z}}(N)$ oraz wszystkie stany $\boldsymbol{z}(n)$, $n=0,1,...,N-1$. Następnie podstawiamy $n = N - 1$.

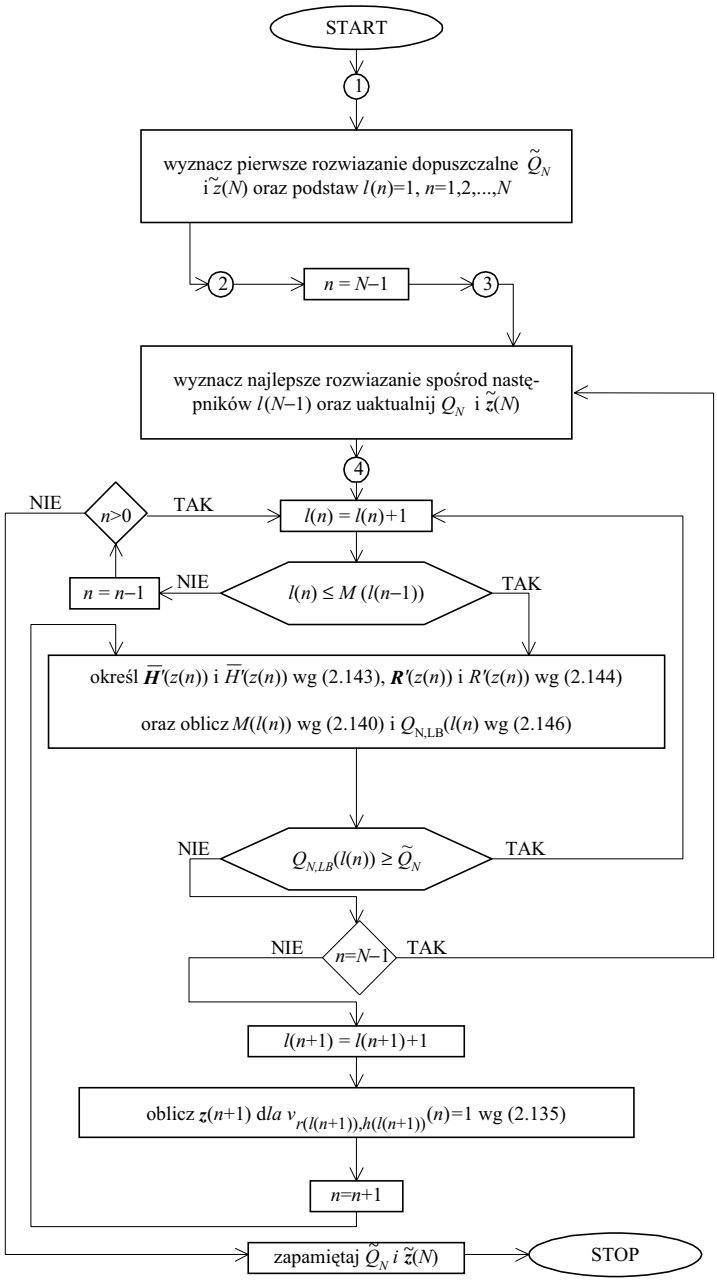
2. Sprawdzamy wszystkie rozwiązania, będące następnikami $l(N-1)$ i w konsekwencji wybieramy rozwiązanie najlepsze, stosując porównanie wszystkich wartości kryterium jakości, oraz aktualizujemy $\tilde{Q}_N$ i $\tilde{\boldsymbol{z}}(N)$.

3. Podstawiamy $l(n) = l(n)+1$. Jeśli $l(n) \le M(l(n-1))$, to przechodzimy do kroku 4, w przeciwnym przypadku – do kroku 6.

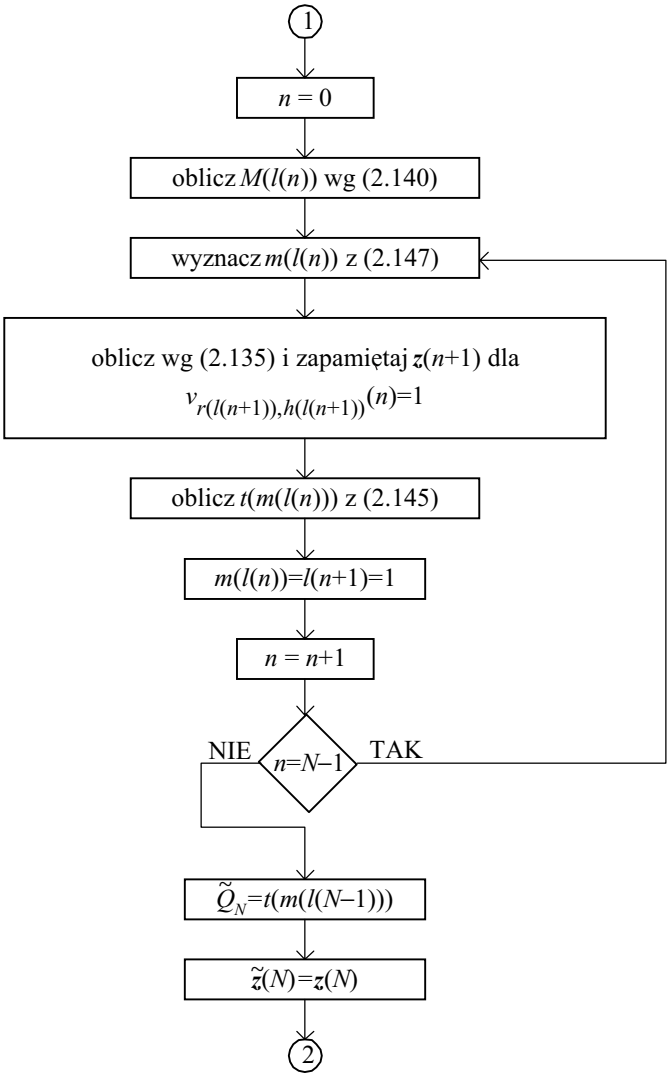
4. Określamy postaci i liczności zbiorów $\overline{\boldsymbol{H}}'(z(n)), \boldsymbol{R}'(z(n))$, oraz obliczamy liczbę $M(l(n))$ następników wierzchołka $l(n)$ i dolne ograniczenie $Q_{N,LB}(l(n))$. Jeśli $Q_{N,LB}(l(n)) \ge \tilde{Q}_N$, to przechodzimy do kroku 3, w przeciwnym przypadku – do kroku 5.

5. Jeśli $n = N-1$, to przechodzimy do kroku 2, w przeciwnym przypadku podstawiamy $l(n+1) = l(n+1)+1$ i obliczamy $\boldsymbol{z}(n+1)$ dla $v_{r(l(n+1)),h(l(n+1))}(n)=1$ oraz podstawiamy $n = n + 1$, a następnie przechodzimy do kroku 4.

6. Podstawiamy $n = n-1$. Jeśli $n > 0$, to przechodzimy do kroku 3, w przeciwnym przypadku kończymy procedurę z optymalną wartością kryterium jakości równą $\tilde{Q}_N$ oraz optymalnymi trasami poruszania się realizatorów, zawartymi w stanie $\tilde{\boldsymbol{z}}(N)$.

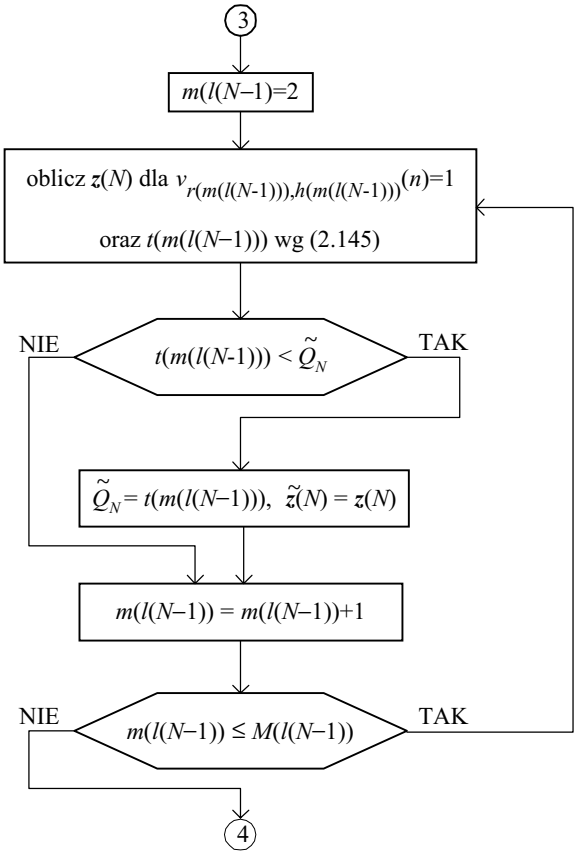


Rys. 2.8. Schemat blokowy zasady podziału i ograniczeń, stosowanej do rozwiązania problemu szeregowania, z uwzględnieniem ruchu realizatorów po dekompozycji czasowej



Rys. 2.9. Fragment schematu blokowego z rys. 2.8, ilustrujący sposób wyznaczania pierwszego rozwiązania dopuszczalnego

Procedurę tę przedstawiamy również na rys. 2.8, 2.9 i 2.10 w postaci schematów blokowych.



Rys. 2.10. Fragment schematu blokowego z rys. 2.8, ilustrujący sposób wyznaczania najlepszego rozwiązania spośród następników wierzchołka $l(N-1)$

Przykład 2.3

Przyjmujemy dane takie jak w przykładzie 2.1. Stan początkowy obliczony zgodnie z (2.118) ma postać

$$\boldsymbol{z}(0) = \begin{bmatrix} 126 & 127 & 122 & 118 & 117 & 65 & 6 \\ 126 & 111 & 118 & 119 & 122 & 63 & 6 \end{bmatrix}.$$

W wyniku zastosowania omówionego algorytmu rozwiązania, zrealizowanego w postaci odpowiedniego programu komputerowego, otrzymano macierz $\tilde{\boldsymbol{z}}(7)$ w postaci

$$\tilde{\boldsymbol{z}}(7)=\begin{bmatrix}-6 & 127 & 122 & -4 & 117 & -3 & 6\\ 126 & -2 & -5 & 119 & -6 & -1 & 6\end{bmatrix}$$

oraz $\tilde{Q}_N = 379$. Z macierzy $\tilde{\boldsymbol{z}}(7)$ odczytujemy trasy poruszania się realizatorów:

$$\text{realizator } r = 1 - \text{trasa}: 6 \to 1 \to 4 \to 6,$$

$$\text{realizator } r = 2 - \text{trasa}: 6 \to 5 \to 3 \to 2 \to 6.$$

Kolejność wykonywania zadań przedstawiamy również na rys. 2.11 w postaci wykresu Gantta (ostatnie pola oznaczają zjazdy do bazy, a w nawiasach podano czasy wykonywania zadań).

Rys. 2.11. Kolejność wykonywania zadań, uzyskana w przykładzie 2.3

Zwróćmy jeszcze uwagę na dwa zagadnienia.

1. Optymalna wartość kryterium $\tilde{Q}_N = Q^* = 379$ jest taka sama jak w przypadku 2 z przykładu 2.1. Uzyskana trasa różni się jednak od analogicznej trasy z przykładu 2.1 (por. np. rys. 2.11 i rys. 2.4).

2. Ponieważ z macierzy $\tilde{\boldsymbol{z}}(7)$ możemy odczytać informacje o momentach zakończenia wykonywania zadań, zauważamy, że w momencie równym 111 (w takcie n = 1) oba realizatory kończą wykonywanie zadań; realizator $r = 1$ kończy wykonywanie zadania h = 1, a realizator r = 2 – zadania h = 5. Informują o tym takie same wartości – 6, występujące w pierwszej i piątej kolumnie macierzy $\tilde{\boldsymbol{z}}(7)$. Tak więc, w takcie n = 1, kiedy należy podjąć decyzję o $\boldsymbol{v}(1)$, $R'(\boldsymbol{z}(1)) = 2$ ($\overline{H}'(\boldsymbol{z}(1)) = 5$, ponieważ należy wykonać jeszcze 3 zadania i 2 zjazdy do bazy). W pozostałych taktach $R'(\boldsymbol{z}(n)) = 1$, a wartości $\overline{H}'(\boldsymbol{z}(n))$ zmniejszają się o jeden w kolejnych taktach.

Zgodnie z (2.142), już w pierwszej warstwie $n = 1$ znajduje się $L(1) = 6 \cdot 5 = 30$ wierzchołków. Biorąc pod uwagę wartości $\overline{H}'(\boldsymbol{z}(n))$ i $R'(\boldsymbol{z}(n))$ dla $n = 0,1, ..., 7$ określamy liczbę wszystkich rozwiązań, czyli liczbę rozwiązań w warstwie ostatniej i otrzymujemy wartość $l(7)$ = 14 400. W przypadku 1556 wierzchołków obliczenie dolnego ograniczenia (2.146) okazało się skuteczne i rozwiązania wynikają-

ce z tych węzłów nie musiały być dalej sprawdzane, natomiast w 666 przypadkach algorytm sprawdzał rozwiązania w warstwie najniższej.

■

2.4.2. Badania symulacyjne i porównanie algorytmów rozwiązania

Ocena jakości (efektywności) algorytmu rozwiązania każdego problemu decyzyjnego jest sprawą kluczową. Jakość algorytmów przybliżonych lub heurystycznych wiąże się z ich dokładnością, czyli w różny sposób definiowaną odległością od rozwiązania optymalnego. W przypadku algorytmów dokładnych o jakości świadczy krótki czas działania. Wyznaczanie ocen jakości w sposób analityczny nie jest zwykle proste. Nawet jeśli oceny takie istnieją, nie umożliwiają wszechstronnej oceny algorytmów rozwiązania. Dlatego dużą role pełnią badania symulacyjne, realizowane przy pomocy specjalnie opracowanych w tym celu programów komputerowych. Istotne jest przy tym określenie tych parametrów obiektu, wobec którego są podejmowane decyzje i (lub) parametrów algorytmu decyzyjnego, które mogą mieć znaczenie dla jakości algorytmów. Niemniej ważne jest zdefiniowanie miary jakości algorytmów. Badania symulacyjne polegają wtedy na określeniu empirycznych zależności między wartościami parametru a wprowadzoną miarą oceniającą jakość algorytmu decyzyjnego, czyli na obliczaniu wartości tej miary dla różnych wartości parametru.

Koncentrując się na rozważanym w tym rozdziale problemie szeregowania zadań z uwzględnieniem ruchu realizatorów, zauważmy, że analitycznej oceny algorytmów rozwiązania dokonano tylko dla algorytmów przybliżonych (twierdzenia 2.2 i 2.4). Oceny te nie umożliwiają jednak wszechstronnej oceny algorytmów. W przypadku tego problemu istotne jest zbadanie wpływu parametrów obiektu, czyli liczby zadań, liczby realizatorów i czasów wykonywania zadań na jakość rezultatu generowanego przez wyznaczony algorytm. Najczęściej stosowaną miarą jakości algorytmów (przybliżonych lub heurystycznych) jest wskaźnik będący względną różnicą wartości kryteriów jakości dla rozwiązania uzyskanego przez oceniany algorytm oraz dla rozwiązania optymalnego. Dla rozważanego problemu szeregowania sformułujemy ten wskaźnik w postaci

$$\delta_1 = \frac{\tilde{Q} - Q^*}{Q^*}, \tag{2.148}$$

gdzie: $\tilde{Q}$ – wartość kryterium jakości uzyskana w wyniku działania dokładnego algorytmu rozwiązania problemu po dekompozycji funkcjonalnej,

Q^* – optymalna wartość kryterium jakości.

Wskaźnik jakości (2.148) ma charakter uniwersalny i jest prawdziwy zarówno dla kryterium w postaci długości uszeregowania, jak i maksymalnego opóźnienia. Dlatego, w zależności od stosowanego kryterium, $\tilde{Q}$ może oznaczać $\tilde{Q}^{*}_{\mathrm{C}}(\tilde{\gamma}_0)$ lub $\tilde{Q}^{*}_{\mathrm{L}}(\tilde{\alpha}_0)$. Podobnie Q^{*} może oznaczać Q^{*}_{C} lub Q^{*}_{L}.

Przed dalszą prezentacją podstaw do badań symulacyjnych wyjaśnienia wymaga kwestia użytego sformułowania, że wartość kryterium została uzyskana w wyniku działania algorytmu dokładnego. Należy podkreślić, że wskaźnik (2.148) ocenia pogorszenie jakości rozwiązania w stosunku do rozwiązania optymalnego, powstałe tylko w wyniku zastosowania algorytmu rozwiązania, składającego się z trzech kroków i przedstawionego w podpunkcie 2.3. Algorytm ten oparty jest na dekompozycji funkcjonalnej, która sama – zgodnie z twierdzeniem 2.1 lub 2.3 w przypadku maksymalnego opóźnienia – nie powoduje pogorszenia jakości rozwiązania. Inną sprawą – uwzględnioną we wskaźniku (2.148) – jest dokładność rozwiązania problemów szeregowania i komiwojażera z punktów 2 i 3 ocenianych algorytmów. Wyjaśniane sformułowanie oznacza, że problemy te są rozwiązywane w sposób dokładny, to znaczy przy użyciu algorytmów dokładnych o wykładniczej złożoności obliczeniowej. W przypadku zastosowania algorytmów przybliżonych wskaźnik (2.148) zapisujemy jako

$$\delta'_1 = \frac{\tilde{Q}' - Q^{*}}{Q^{*}}, \tag{2.149}$$

gdzie: $\tilde{Q}'$ – wartość kryterium jakości (któregokolwiek z rozpatrywanych) uzyskana w wyniku działania przybliżonego algorytmu rozwiązania problemu po dekompozycji funkcjonalnej.

Algorytm jest traktowany jako przybliżony, jeśli co najmniej jeden z problemów, to znaczy problem szeregowania z drugiego kroku lub problem komiwojażera z trzeciego kroku jest rozwiązywany przez algorytm przybliżony. Można podać związek między wartościami $\tilde{Q}$ oraz $\tilde{Q}'$. Określimy go dla kryterium w postaci długości uszeregowania w formie następującego twierdzenia.

Twierdzenie 2.6

Niech $\tilde{Q}'_{\mathrm{C}}(\tilde{\gamma}_0)$ oznacza wartość długości uszeregowania dla problemu po dekompozycji funkcjonalnej, uzyskaną za pomocą przybliżonych algorytmów szeregowania i komiwojażera. Wtedy

$$\frac{\tilde{Q}'_{\mathrm{C}}(\tilde{\gamma}_0)}{\tilde{Q}^{*}_{\mathrm{C}}(\tilde{\gamma}_0)} \le \kappa_{\mathrm{I}}\,\kappa_{\mathrm{II}}, \tag{2.150}$$

gdzie κ_{I} i κ_{II} to oszacowania najgorszego przypadku (1.61) dla przybliżonych algorytmów rozwiązania odpowiednio problemu szeregowania i problemu komiwojażera.

Dowód

Wykorzystujemy własności współczynników κ_{I} i κ_{II} oraz przekształcamy zależność (2.55), tj.

$$Q_{\mathrm{C}}^{*} = \min_{\substack{\boldsymbol{c}\in\boldsymbol{C}\\ \tilde{\boldsymbol{\gamma}}\in\tilde{\boldsymbol{\Gamma}}}}[\tilde{Q}_{\mathrm{C}}(\boldsymbol{c},\tilde{\boldsymbol{\gamma}})] \le \min_{\tilde{\boldsymbol{\gamma}}\in\boldsymbol{\Gamma}} \min_{\substack{\boldsymbol{c}\in\boldsymbol{C}\\ \tilde{\boldsymbol{\gamma}}=\tilde{\boldsymbol{\gamma}}_0\in\tilde{\boldsymbol{\Gamma}}}} \tilde{Q}_{\mathrm{C}}(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}) \le \min_{\tilde{\boldsymbol{\gamma}}\in\boldsymbol{\Gamma}(\boldsymbol{c}^{*}(\tilde{\boldsymbol{\gamma}}_0))} \kappa_{\mathrm{I}}\overline{Q}(\tilde{\boldsymbol{\gamma}};\tilde{\boldsymbol{\gamma}}_0)$$

$$= \kappa_{\mathrm{I}} \max_{r=1,2,\ldots,R} \left\{ \min_{\tilde{\boldsymbol{\gamma}}^{r}\in\boldsymbol{\Gamma}^{r}(\boldsymbol{c}^{*}(\tilde{\boldsymbol{\gamma}}_0))} \overline{Q}_{\mathrm{C}}^{r}(\tilde{\boldsymbol{\gamma}}^{r};\tilde{\boldsymbol{\gamma}}_0) \right\} \le \kappa_{\mathrm{I}} \max_{r=1,2,\ldots,R} \left\{ \kappa_{\mathrm{II}} \tilde{Q}_{\mathrm{C}}^{r,*}(\tilde{\boldsymbol{\gamma}}_0) \right\}$$

$$= \kappa_{\mathrm{I}}\kappa_{\mathrm{II}} \max_{r=1,2,\ldots,R} \left\{ \tilde{Q}_{\mathrm{C}}^{r,*}(\tilde{\boldsymbol{\gamma}}_0) \right\} = \kappa_{\mathrm{I}}\kappa_{\mathrm{II}} \tilde{Q}^{*}(\tilde{\boldsymbol{\gamma}}_0) \stackrel{\Delta}{=} \tilde{Q}'_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}}_0). \qquad (2.151)$$

Z porównania (2.55) i (2.151) wyciągamy wniosek, że w najbardziej niekorzystnym przypadku zastosowanie algorytmów przybliżonych κ_{I} κ_{II}-krotnie zwiększa wartość kryterium jakości $\tilde{Q}_{\mathrm{C}}^{*}(\tilde{\boldsymbol{\gamma}}_0)$. Stąd bezpośrednio wynika (2.150).

■

Konsekwencją twierdzeń 2.2 i 2.6 jest wniosek dotyczący porównania $\tilde{Q}'_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}}_0)$ i Q_{C}^{*}.

Wniosek

Całkowite pogorszenie jakości rozwiązania problemu PC w rezultacie zastosowania algorytmu rozwiązania wykorzystującego dekompozycję funkcjonalną, w którym użyto przybliżonych algorytmów rozwiązania problemów szeregowania i komiwojażera, można wyrazić w postaci nierówności

$$\tilde{Q}'_{\mathrm{C}}(\tilde{\boldsymbol{\gamma}}_0) - Q_{\mathrm{C}}^{*} \le \kappa_{\mathrm{I}}\kappa_{\mathrm{II}}(\varepsilon_{\gamma} + \varepsilon_{c}). \qquad (2.152)$$

■

Analogiczny wniosek można wyciągnąć dla kryterium w postaci maksymalnego opóźnienia. Wtedy

$$\tilde{Q}'_{\mathrm{L}}(\tilde{\boldsymbol{\alpha}}_0) - Q_{\mathrm{L}}^{*} \le \kappa_{\mathrm{I}}\ \kappa_{\mathrm{II}}\ (\varepsilon_{\mathrm{I}} + \varepsilon_{\mathrm{II}}). \qquad (2.153)$$

Wracając do głównego nurtu rozważań należy zauważyć, że często są stosowane inne postaci wskaźników (2.148) i (2.149), które umożliwiają procentową ocenę pogorszenia jakości otrzymywanego rozwiązania, a mianowicie

$$\hat{\delta} = \frac{\tilde{Q} - Q^*}{\tilde{Q}}, \tag{2.154}$$

$$\hat{\delta}' = \frac{\tilde{Q}' - Q^*}{\tilde{Q}'}. \tag{2.155}$$

Wszystkie one są użyteczne tylko wtedy, gdy są znane wartości rozwiązań optymalnych. Te z kolei, wobec wykładniczej złożoności dokładnych algorytmów rozwiązania dla rozpatrywanego problemu, można efektywnie obliczyć tylko dla niewielkich rozmiarów problemu, czyli dla niewielkiej liczby zadań oraz realizatorów. Wobec tego istnieje potrzeba definiowania innych wskaźników, nie wykorzystujących wartości kryterium dla rozwiązań optymalnych.

Badania symulacyjne

Sformułujemy teraz trzy zastępcze wskaźniki, umożliwiające oszacowanie jakości rozpatrywanych algorytmów rozwiązania również dla problemów o dużych rozmiarach. Będą one podawane tylko w wersji dla przybliżonego algorytmu rozwiązania problemu po dekompozycji funkcjonalnej. Sformułowania dla algorytmu dokładnego są analogiczne. W pewnych sytuacjach użyteczna jest wprost wartość kryterium jakości. Wtedy wskaźnik jakości ma postać

$$\delta_2' = \tilde{Q}'. \tag{2.156}$$

Do konstrukcji kolejnego wskaźnika wykorzystamy zależność między $\tilde{Q}'$ i Q^* przedstawioną w (2.152) dla długości uszeregowania lub w (2.153) dla maksymalnego opóźnienia. Jeśli prawą stronę w (2.152) lub w (2.153) w skrócie oznaczymy przez E, to możemy otrzymać nierówność $Q^* \geq \tilde{Q}' - E$ (tak jak poprzednio, brak dolnego indeksu przy kryterium oznacza, że odpowiednie zmienne dotyczą zarówno długości uszeregowania, jak i maksymalnego opóźnienia – w zależności od rozpatrywanego przypadku). Nierówność tę wykorzystujemy do oszacowania wskaźnika (2.149), a mianowicie

$$\delta_1' = \frac{\tilde{Q}'}{Q^*} - 1 \leq \frac{\tilde{Q}'}{\tilde{Q}' - E} - 1, \tag{2.157}$$

skąd otrzymujemy wskaźnik δ_3' w postaci

$$\delta_3' = \frac{E}{\tilde{Q}' - E} \geq \delta_1', \text{ dla } E \in [0, \tilde{Q}']. \tag{2.158}$$

Wskaźnik δ_3' jest górnym oszacowaniem δ_1'. Łączne wykorzystanie wskaźników δ_2' i δ_3' umożliwia ocenę i zastąpienie trudnego do wyznaczenia wskaźnika δ_1', ponieważ δ_2' informuje o bezwzględnych zmianach wartości kryterium jakości ocenianego algorytmu, a δ_3' jest górnym oszacowaniem δ_1'. W trzeciej propozycji zastąpienia δ_1' innym wskaźnikiem proponujemy zastosować zamiast Q^* wartość jego dolnego ograniczenia Q_{LB}. Wówczas

$$\delta_1' = \frac{\tilde{Q}'}{Q^*} - 1 \le \frac{\tilde{Q}'}{Q_{\mathrm{LB}}} - 1 . \tag{2.159}$$

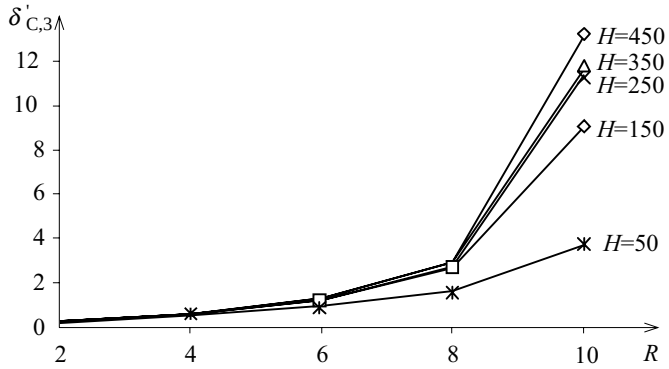
Stąd

$$\delta_4' = \frac{\tilde{Q}' - Q_{\mathrm{LB}}}{Q_{\mathrm{LB}}} \ge \delta_1' , \tag{2.160}$$

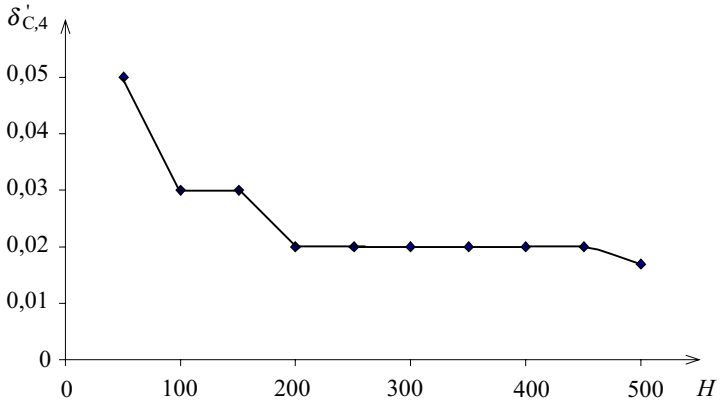
czyli δ_4' jest również górnym oszacowaniem δ_1'. Dla kryterium w postaci długości uszeregowania do wyznaczania funkcji dolnego ograniczenia można na przykład wykorzystać wyrażenie (2.146).

Wskaźnik δ_3' najpełniej wykorzystuje specyfikę rozważanego problemu i dotychczas uzyskane wyniki analityczne. Wskaźniki δ_2' i δ_4' mają ogólny charakter i mogą być wykorzystywane do oceny jakości innych algorytmów decyzyjnych niż te, które są przedmiotem niniejszych rozważań. Jakość wskaźnika δ_4' zależy od trafności doboru dolnych ograniczeń optymalnych wartości kryterium jakości.

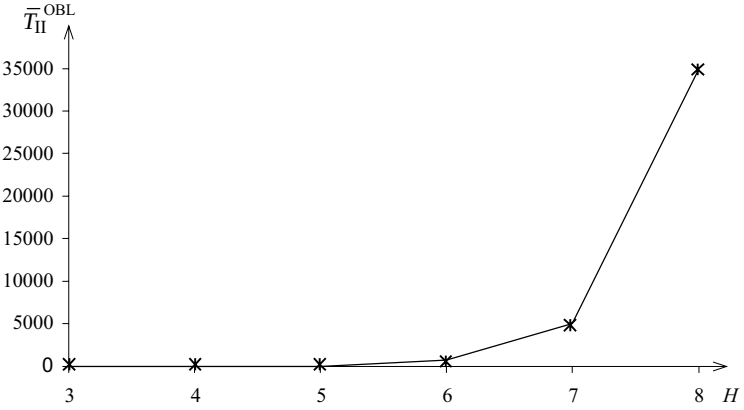
W przypadku oceny algorytmów dokładnych jedyną istotną oceną jego jakości jest czas obliczeń, czyli czas uzyskiwania przez ten algorytm rozwiązania optymalnego. Czas ten będziemy oznaczać jako $T_{\mathrm{II}}^{\mathrm{OBL}}$. Jak już wspomniano, dla rozpatrywanego problemu szeregowania istotne jest udzielenie odpowiedzi na pytanie: Czy dokładność rozwiązania uzyskiwanego przez algorytmy przybliżone oraz czas uzyskiwania rozwiązania przez algorytm dokładny zależą od danych problemu szeregowania? Dokładność rozwiązania może być wyrażana analitycznie w postaci wprowadzonych wcześniej wskaźników. Jeśli przy zmianie danych problemu uzyskujemy różną dokładność rozwiązania, to ważny jest przebieg tej zależności. Odpowiedź na to pytanie jest ważna z poznawczego punktu widzenia, ale również jest istotna dla projektantów odpowiednich systemów produkcyjnych, ponieważ umożliwia ustalenie takiej struktury systemu, dla której otrzymywane rozwiązania problemu szeregowania są bliższe rozwiązaniom optymalnym. Wobec uzyskanych skromnych rezultatów analitycznych, odpowiedź na to pytanie można uzyskać jedynie przez komputerowe badania symulacyjne. Badania takie zostały przeprowadzone głównie dla kryterium w postaci długości uszeregowania. Sprawdzono w nich wpływ licz-



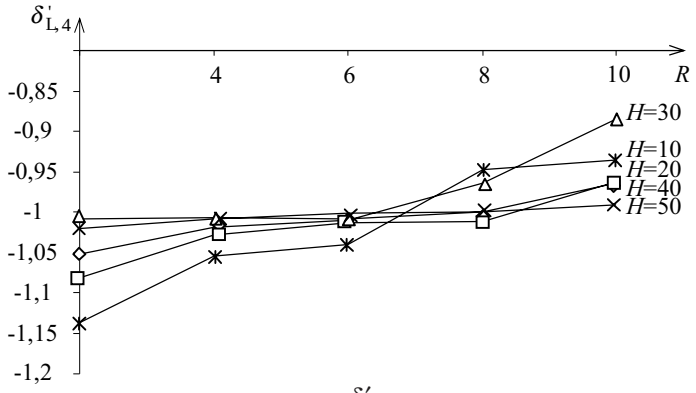
Rys. 2.12. Zależność $\delta'_{C,3}$ od R dla różnych H



Rys. 2.13. Zależność $\delta'_{C,4}$ od H dla $R = 2$



Rys. 2.14. Zależność $\bar{T}_{II}^{OBL}$ od H dla $R = 2$



Rys. 2.15. Zależność $\delta'_{\mathrm{L},4}$ od R dla różnych H

by zadań H, liczby realizatorów R oraz czasów wykonywania zadań na wprowadzone wskaźniki jakości. Dokładny plan badań symulacyjnych oraz wyczerpujące wyniki wraz z komentarzami przedstawiono w pracach [58, 63]. Na rysunkach 2.12–2.15 zaprezentowano przykładowe wyniki. Na rys. 2.14 przedstawiono wykres dla unormowanych wartości czasu obliczeń, oznaczonych jako $\overline{T}_{\mathrm{II}}^{\mathrm{OBL}}$. Wartość unormowana jest ilorazem $T_{\mathrm{II}}^{\mathrm{OBL}}$ uzyskanego dla określonej wartości zmiennej niezależnej (w tym przypadku H), oraz wartości $T_{\mathrm{II}}^{\mathrm{OBL}}$ dla pierwszej, w danym badaniu, wartości tej zmiennej. Dlatego na rys. 2.14 wartość $\overline{T}_{\mathrm{II}}^{\mathrm{OBL}}$ dla $H = 3$ wynosi 1.

Analizując odpowiednie wykresy można stwierdzić, że:

1. Ze wzrostem wartości R zwiększa się wartość $\delta'_{\mathrm{C},3}$. Dynamika tej zależności jest tym większa, im większe jest R (rys. 2.12). Uwzględniając oszacowanie (2.158), można się spodziewać, że wzrost R powoduje wzrost błędu $\delta'_{\mathrm{C},1}$. Jest to zrozumiałe, ponieważ podobną własność ma zastosowany przybliżony algorytm rozwiązania klasycznego problemu szeregowania w drugim kroku algorytmu rozwiązania. Zastosowano algorytm opisany w pracy [47], który przedstawiono również w punkcie 1.3. Oszacowanie najgorszego przypadku dla tego algorytmu zależy od liczby realizatorów R.

2. Zwiększanie wartości H powoduje zmniejszanie się wartości wskaźnika $\delta'_{\mathrm{C},4}$ (rys. 2.13). Można to uzasadnić większą dynamiką wzrostu $Q_{\mathrm{C,LB}}$ niż $\widetilde{Q}'_{\mathrm{C}}$.

3. Czas obliczeń $T_{\mathrm{II}}^{\mathrm{OBL}}$ gwałtownie (wykładniczo) zwiększa się wraz ze wzrostem liczby zadań H (rys. 2.14).

4. Dla stałej liczby zadań wzrost liczby realizatorów R powoduje uzyskiwanie lepszych wartości kryterium jakości. Wzrost wartości wskaźnika $\delta'_{\mathrm{L},4}$ wynika z faktu, że wartość dolnego ograniczenia jest wtedy ujemna. Charakter tej zależności jest

przedstawiony na rys. 2.15. Dynamika zmian jest nieznacznie większa dla mniejszej liczby zadań.

Porównanie algorytmów rozwiązania

Algorytmy przybliżone i dokładne mają swoje wady i zalety. W sytuacjach kiedy można stosować każdy z nich, warto rozstrzygnąć, którego z algorytmów użyć: czy przybliżonego, który umożliwia szybsze uzyskiwanie rozwiązania, czy też dokładnego, który zapewnia otrzymanie rozwiązania optymalnego? Mówiąc w skrócie, można wtedy wybierać miedzy dokładnością rozwiązania a czasem jego otrzymywania. Przedstawimy teraz formalny sposób porównania obu algorytmów. Jest on prawdziwy dla obu rozpatrywanych wcześniej kryteriów jakości szeregowania, dlatego zastosujemy – tak jak poprzednio – oznaczenia ogólne, prawdziwe dla obu przypadków. Przyjmujemy, że problemy szeregowania i komiwojażera w krokach 2. i 3. są rozwiązywane w sposób dokładny, a otrzymywana niedokładność wynika wyłącznie z zastosowanego algorytmu, bazującego na dekompozycji funkcjonalnej. Przyjęcie takiego założenia oznacza, że dla obu przypadków algorytmy rozwiązania odznaczają się wykładniczą złożonością obliczeniową. Pomimo to czasy działania algorytmów dla tych przypadków mogą być zasadniczo różne dla tych samych danych problemów i dlatego istotne jest przeprowadzenie wspomnianego porównania z uwzględnieniem zarówno dokładności, jak i czasu. Wprowadźmy wskaźnik porównania algorytmów. Jest on funkcją wskaźników oceniających poszczególne algorytmy. Wskaźnik oceniający algorytm przybliżony zależy od straty związanej ze zmniejszeniem dokładności rozwiązania w wyniku zastosowania algorytmu oraz od przewidywanego zysku związanego ze zmniejszeniem czasu uzyskania rozwiązania – w stosunku do algorytmu dokładnego. Z kolei wskaźnik oceniający algorytm dokładny jest zależny od straty spowodowanej wydłużonym, w stosunku do drugiego z porównywanych algorytmów, czasem uzyskiwania rozwiązania oraz od zysku na dokładności rozwiązania. Wspomniane wskaźniki przedstawiamy w formie analitycznej. Niech

$$J_{\mathrm{I}} = J_{\mathrm{I},1} + J_{\mathrm{I},2}\,, \tag{2.161}$$

gdzie: J_{I} – wskaźnik oceniający algorytm przybliżony,
$J_{\mathrm{I},1}$ – strata związana ze zmniejszeniem dokładności rozwiązania,
$J_{\mathrm{I},2}$ – zysk związany ze zmniejszeniem czasu obliczania rozwiązania

oraz

$$J_{\mathrm{I},1} = \lambda \frac{\tilde{Q} - Q^*}{Q^*}, \tag{2.162}$$

$$J_{\mathrm{I},2} = (1-\lambda)\frac{T_{\mathrm{I}}^{\mathrm{OBL}} - T_{\mathrm{II}}^{\mathrm{OBL}}}{T_{\mathrm{II}}^{\mathrm{OBL}}}, \tag{2.163}$$

gdzie: $\lambda \in [0, 1]$ – współczynnik wagowy,

$T_{\mathrm{I}}^{\mathrm{OBL}}$ – czas uzyskania rozwiązania przez algorytm przybliżony.

Analogicznie określamy wskaźnik oceniający drugi algorytm, a mianowicie

$$J_{\mathrm{II}} = J_{\mathrm{II},1} + J_{\mathrm{II},2}, \tag{2.164}$$

gdzie: J_{II} – wskaźnik oceniający algorytm dokładny,

$J_{\mathrm{II},1}$ – strata związana ze zwiększeniem czasu wyznaczania rozwiązania,

$J_{\mathrm{II},2}$ – zysk związany z uzyskaniem rozwiązania optymalnego

oraz

$$J_{\mathrm{II},1} = (1-\lambda)\frac{T_{\mathrm{II}}^{\mathrm{OBL}} - T_{\mathrm{I}}^{\mathrm{OBL}}}{T_{\mathrm{I}}^{\mathrm{OBL}}}, \tag{2.165}$$

$$J_{\mathrm{II},2} = \lambda\frac{Q^* - \tilde{Q}}{\tilde{Q}}. \tag{2.166}$$

Współczynniki λ i $(1-\lambda)$, występujące w równaniach (2.162) i (2.165), wyrażają wagę strat związanych odpowiednio ze zmniejszeniem dokładności i zwiększeniem czasu obliczeń. Współczynniki $(1 - \lambda)$ i λ w równaniach (2.163) i (2.166) określają wagę zysków związanych odpowiednio ze zmniejszeniem czasu obliczeń i uzyskaniem rozwiązania optymalnego. Wskaźniki $J_{\mathrm{I},1}$ i $J_{\mathrm{II},1}$ są nieujemne, wskaźniki $J_{\mathrm{I},2}$ i $J_{\mathrm{II},2}$ przyjmują natomiast wartości niedodatnie. Wspomniany wskaźnik porównania obu algorytmów oznaczamy jako J i określamy jako różnicę J_{I} i J_{II}, tj.

$$J \stackrel{\Delta}{=} J_{\mathrm{I}} - J_{\mathrm{II}}. \tag{2.167}$$

Umożliwia on określenie warunku oceny algorytmów. W warunku tym jest uwzględniana wartość wskaźnika. Jeśli $J < 0$, to uważamy, że lepszy jest algorytm przybliżony, natomiast gdy $J > 0$, wtedy lepszy jest algorytm dokładny. Gdy $J = 0$, przyjmujemy, że oba algorytmy są równoważne. Po uwzględnieniu szczegółowych zależności, określających J_{I} i J_{II}, wskaźnik J możemy zapisać w postaci

$$J = \lambda\left(\frac{\tilde{Q}}{Q^*} - \frac{Q^*}{\tilde{Q}}\right) + (1-\lambda)\left(\frac{T_{\mathrm{I}}^{\mathrm{OBL}}}{T_{\mathrm{II}}^{\mathrm{OBL}}} - \frac{T_{\mathrm{II}}^{\mathrm{OBL}}}{T_{\mathrm{I}}^{\mathrm{OBL}}}\right). \tag{2.168}$$

Zwróćmy uwagę na szczególne przypadki. Ustalenie $\lambda = 0$ oznacza, że w ocenie algorytmów w ogóle nie uwzględniamy dokładności rozwiązania. Wtedy, zgodnie z (2.168), $J < 0$, ponieważ $T_{\mathrm{I}}^{\mathrm{OBL}} < T_{\mathrm{II}}^{\mathrm{OBL}}$ i lepszy jest algorytm przybliżony. Założenie $\lambda = 1$ oznacza z kolei, że w ocenie algorytmów jest uwzględniana wyłącznie dokładność rozwiązania. W takiej sytuacji drugi składnik w (2.168) jest równy zeru, a więc $J > 0$ i lepszy jest algorytm dokładny. Warunek oceny algorytmów możemy przedstawić również w innych formach, w których szacujemy stosunek wartości kryteriów jakości $\tilde{Q}$ i Q^* oraz czasu obliczeń $T_{\mathrm{I}}^{\mathrm{OBL}}$ i $T_{\mathrm{II}}^{\mathrm{OBL}}$. Niech dla $\lambda > 0$

$$A_1 \stackrel{\Delta}{=} \frac{\sqrt{\left[\left(\frac{T_{\mathrm{I}}^{\mathrm{OBL}}}{T_{\mathrm{II}}^{\mathrm{OBL}}}\right)^2 - 1\right]^2 (1-\lambda)^2 + 4\lambda^2 \left(\frac{T_{\mathrm{I}}^{\mathrm{OBL}}}{T_{\mathrm{II}}^{\mathrm{OBL}}}\right)^2} - (1-\lambda)\left[\left(\frac{T_{\mathrm{I}}^{\mathrm{OBL}}}{T_{\mathrm{II}}^{\mathrm{OBL}}}\right)^2 - 1\right]}{2\lambda \frac{T_{\mathrm{I}}^{\mathrm{OBL}}}{T_{\mathrm{II}}^{\mathrm{OBL}}}}. \quad (2.169)$$

Wtedy:

dla $\tilde{Q}/Q^* < A_1$ – lepszy jest algorytm przybliżony,
dla $\tilde{Q}/Q^* = A_1$ – oba algorytmy są równoważne,
dla $\tilde{Q}/Q^* > A_1$ – lepszy jest algorytm dokładny.

Jeżeli natomiast dla $\lambda < 1$

$$A_2 \stackrel{\Delta}{=} \frac{\sqrt{\left[\left(\frac{\tilde{Q}}{Q^*}\right)^2 - 1\right]^2 \lambda^2 + 4(1-\lambda)^2 \left(\frac{\tilde{Q}}{Q^*}\right)^2} - \lambda\left[\left(\frac{\tilde{Q}}{Q^*}\right)^2 - 1\right]}{2(1-\lambda)\frac{\tilde{Q}}{Q^*}}, \quad (2.170)$$

to

dla $T_{\mathrm{I}}^{\mathrm{OBL}}/T_{\mathrm{II}}^{\mathrm{OBL}} < A_2$ – lepszy jest algorytm przybliżony,
dla $T_{\mathrm{I}}^{\mathrm{OBL}}/T_{\mathrm{II}}^{\mathrm{OBL}} = A_2$ – oba algorytmy są równoważne,
dla $T_{\mathrm{I}}^{\mathrm{OBL}}/T_{\mathrm{II}}^{\mathrm{OBL}} > A_2$ – lepszy jest algorytm dokładny.

Oczywiście wynik porównania w zasadniczy sposób zależy od wartości λ, która wyraża sposób preferencji użytkownika, dotyczącej dokładności bądź czasu obliczeń.

2.4.3. Przypadek probabilistyczny

Zagadnienie to rozpatrzymy w nawiązaniu do punktu 1.4.1. Podobnie jak tam rozważymy tylko przypadek, gdy niezdeterminowane są czasy wykonywania zadań oraz, gdy są one realizacjami zmiennych losowych [73]. Wyniki przedstawimy dla kryterium w postaci długości uszeregowania i dla tej wersji problemu wyjściowego PC, która została otrzymana w wyniku zastosowania dekompozycji funkcjonalnej. Dla szeregowania z ruchomymi realizatorami czas wykonania zadania przez realizator jest sumą czasu dojazdu realizatora do stanowiska i czasu wykonania czynności na stanowisku. Zakładamy, że oba składniki sumy są realizacjami zmiennych losowych. Przyjmujemy również, że wszystkie zmienne losowe są wzajemnie niezależne. Wtedy kryterium jakości (2.30) jest wartością oczekiwaną czasu wykonania wszystkich zadań ze względu na macierze zmiennych losowych $\underline{\bar{\tau}}$ i $\underline{\hat{\tau}}$, a mianowicie

$$\tilde{Q}_{\mathrm{C}}^{\mathrm{E}}(\boldsymbol{c},\boldsymbol{\gamma}) = \underset{\underline{\hat{\tau}},\,\underline{\bar{\tau}}}{\mathrm{E}}\left(\max_{r=1,2,\ldots,R} \{ \sum_{h=1}^{H+1} c_{r,h} (\underline{\bar{\tau}}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{g,h}^{r} \cdot \underline{\hat{\tau}}_{r,g,h}) \} \right). \tag{2.171}$$

W stosowanym zapisie, w celu jego uproszczenia, zmienne losowe $\underline{\hat{\tau}}_{r,h,h}$ nie są pomijane, mimo że nie są brane pod uwagę.

Algorytm rozwiązania, przedstawiony dla przypadku deterministycznego w p. 2.3.1 w formie trzech kroków, ma teraz formę analogiczną:

1. Ustalamy $\tilde{\boldsymbol{\gamma}} = \tilde{\boldsymbol{\gamma}}_0 \in \hat{\boldsymbol{\Gamma}}$.
2. Minimalizujemy $\tilde{Q}_{\mathrm{C}}^{\mathrm{E}}$ względem $\boldsymbol{c} \in \tilde{\boldsymbol{C}}$, to znaczy

$$\min_{\boldsymbol{c}} \tilde{Q}_{\mathrm{C}}^{\mathrm{E}}(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}) = \min_{\boldsymbol{c}} \underset{\underline{\hat{\tau}},\,\underline{\bar{\tau}}}{\mathrm{E}} \left(\max_{r=1,2,\ldots,R} \{ \sum_{h=1}^{H+1} c_{r,h} \underline{\tilde{T}}_{r,h}(\tilde{\gamma}_0^r) \} \right) \stackrel{\Delta}{=} \bar{Q}_{\mathrm{C}}^{\mathrm{E}}(\tilde{\boldsymbol{\gamma}};\tilde{\boldsymbol{\gamma}}_0), \tag{2.172}$$

gdzie

$$\underline{\tilde{T}}_{r,h}(\tilde{\gamma}_0^r) = \underline{\bar{\tau}}_{r,h} + \sum_{g=1}^{H+1} \tilde{\gamma}_{0,g,h}^{r} \underline{\hat{\tau}}_{r,g,h}.$$

Tak jak w przypadku deterministycznym zjazdy realizatorów do bazy nie podlegają szeregowaniu, dlatego problem optymalizacyjny (2.172) powinien być zastąpiony następującym zadaniem optymalizacji

$$\min_{\boldsymbol{c}} \mathop{\mathrm{E}}_{\tilde{\underline{\boldsymbol{T}}}} \Big(\max_{r=1,2,...,R} \Big\{ \sum_{h=1}^{H} c_{r,h} \tilde{\underline{T}}_{r,h}(\tilde{\gamma}_0^r) \Big\} \Big), \tag{2.173}$$

gdzie $\tilde{\underline{\boldsymbol{T}}} = [\tilde{\underline{T}}_{r,h}]_{\substack{r=1,2,...,R \\ h=1,2,...,H}}$. Na podstawie znajomości funkcji gęstości rozkładu prawdopodobieństwa $\hat{f}_{r,g,h}$ oraz $\bar{f}_{r,h}$ odpowiednio dla zmiennych losowych $\hat{\underline{\tau}}_{r,g,h}$ oraz $\bar{\underline{\tau}}_{r,h}$ możemy wyznaczyć funkcje gęstości $f_{r,h}^{\tilde{\underline{T}}}$ zmiennych losowych $\tilde{\underline{T}}_{r,h}$ dla $r = 1,2,...,R$ oraz $h = 1,2,...,H+1$, np. [41]. Przed podaniem zależności analitycznej na $f_{r,h}^{\tilde{\underline{T}}}$ oznaczmy przez $\boldsymbol{G}_{r,h}$ zbiór tych elementów macierzy $\tilde{\gamma}_0^r$, które przyjmują wartości różne od zera, to znaczy $\boldsymbol{G}_{r,h} = \{g \in \overline{\boldsymbol{H}} : \tilde{\gamma}_{0,g,h}^r \neq 0\}$. Dodatkowo niech $g_i \in \boldsymbol{G}_{r,h}$, $i = 1,2,...,G_{r,h}$ będzie indeksem bieżącego elementu zbioru $\boldsymbol{G}_{r,h}$, a $G_{r,h}$ – jego licznością. Wówczas

$$f_{r,h}^{\tilde{\underline{T}}}(t) = \frac{1}{\prod_{i=1}^{G_{r,h}} \left| \tilde{\gamma}_{0,g,h}^r \right|} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \bar{f}_{r,h}(w_0) \prod_{i=1}^{G_{r,h}-1} \hat{f}_{r,g_i,h}\left(\frac{w_i}{\tilde{\gamma}_{0,g_i,h}^r} \right)$$

$$\hat{f}_{r,G_{r,h},h}\left(\frac{t - \sum_{g_i=1}^{G_{r,h}-1} \hat{\tau}_{r,g,h} - \bar{\tau}_{r,h}}{\tilde{\gamma}_{0,G_{r,h},h}^r} \right) dw_{G_{r,h}-1} \cdots dw_1. \tag{2.174}$$

Bardziej zwartą postać (2.174) możemy otrzymać dla szczególnego przypadku, gdy $G_{r,h} = 1$, to znaczy dla ustalonych r oraz h tylko jeden element macierzy $\tilde{\gamma}_0^r$ jest równy 1. Odpowiada to na przykład postaciom (2.67) lub (2.68) macierzy $\tilde{\gamma}_0^r$. Wtedy

$$\tilde{\underline{T}}_{r,h}\left(\tilde{\gamma}_0^r\right) = \bar{\underline{\tau}}_{r,h} + \hat{\underline{\tau}}_{r,g_0,h}$$

oraz

$$f_{r,h}^{\tilde{\underline{T}}}(t) = \int_{-\infty}^{+\infty} \bar{f}_{r,h}(w)\, \hat{f}_{r,g_0,h}(t-w)\, dw, \tag{2.174a}$$

gdzie g_0 jest numerem tego stanowiska, dla którego $\tilde{\gamma}_{0,g,h}^r = 1$. Po wstawieniu (2.174) do (2.173) otrzymujemy klasyczny problem szeregowania w warunkach pro-

babilistycznych. Wynikami jego rozwiązania są: macierz $\boldsymbol{c}^*(\tilde{\gamma}_0)$ lub równoważnie zbiory $\overline{\boldsymbol{H}}_c^{r,*}$ zdefiniowane w (2.50) oraz nowa postać kryterium jakości szeregowania

$$\overline{Q}_{\mathrm{C}}^{\mathrm{E}}(\tilde{\gamma},\tilde{\gamma}_0) = \underset{\hat{\underline{\tau}},\,\underline{\tau}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H}\boldsymbol{c}_{r,h}^{*}(\tilde{\gamma}_0) \cdot\Big(\underline{\tau}_{r,h}+\sum_{g=1}^{H+1}\tilde{\gamma}_{0,g,h}^{r}\hat{\underline{\tau}}_{r,g,h}\Big)+\sum_{g=1}^{H+1}\tilde{\gamma}_{0,g,H+1}\hat{\underline{\tau}}_{r,g,H+1}\Big\}\Big). \tag{2.175}$$

3. Minimalizujemy $\overline{Q}_{\mathrm{C}}^{\mathrm{E}}(\tilde{\gamma};\tilde{\gamma}_0)$ względem $\tilde{\gamma}\in\tilde{\boldsymbol{\Gamma}}$, czyli optymalizujemy wskaźnik jakości

$$\overline{Q}_{\mathrm{C}}^{\mathrm{E}}(\tilde{\gamma};\tilde{\gamma}_0) = \underset{\hat{\underline{\tau}},\,\underline{\tau}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{\overline{H}_c^{r,*}}(\underline{\tau}_{r,h}+\sum_{g=1}^{\overline{H}_c^{r,*}}\tilde{\gamma}_{g,h}^{r}\hat{\underline{\tau}}_{r,g,h})\Big\}\Big), \tag{2.176}$$

gdzie $\overline{H}_c^{r,*}$ jest licznością zbioru $\overline{\boldsymbol{H}}_c^{r,*}$. Zarówno argumenty funkcji maksimum w zależności (2.176) jak i ograniczenia definiujące zbiory $\tilde{\boldsymbol{\Gamma}}^r(\boldsymbol{c})$ tworzące $\tilde{\boldsymbol{\Gamma}}$ są niezależne dla różnych *r*. Z tego powodu podwektory $\tilde{\gamma}^r(\tilde{\gamma}_0)$ (w skrócie $\tilde{\gamma}^r$) wektora $\tilde{\gamma}(\tilde{\gamma}_0)$ dla różnych *r* mogą być wyznaczane niezależnie. Ich optymalne wartości są wynikami minimalizacji kryteriów cząstkowych

$$\overline{Q}_{\mathrm{C}}^{\mathrm{E,r}}(\tilde{\gamma}^r) = \sum_{h=1}^{\overline{H}_C^{r,*}}\underset{\underline{\tau}_{r,h}}{\mathrm{E}}(\underline{\tau}_{r,h}) + \underset{\substack{\hat{\underline{\tau}}_{r,g,h}\\ g,h\in\overline{\boldsymbol{H}}_c^{r,*}\\ g\neq h}}{\mathrm{E}}\left(\sum_{h=1}^{\overline{H}_C^{r,*}}\sum_{g=1}^{\overline{H}_C^{r,*}}\tilde{\gamma}_{g,h}^{r}\cdot\hat{\underline{\tau}}_{r,g,h}\right). \tag{2.177}$$

Ze względu na stałość pierwszego składnika sumy w zależności (2.177) wystarczy minimalizować drugi składnik. Takie zadanie optymalizacji, oczywiście z uwzględnieniem ograniczeń (2.38)–(2.41), jest formalnym zapisem stochastycznego problemu komiwojażera. Opisany sposób dekompozycji w trzecim kroku algorytmu rozwiązania możemy zapisać jako

$$\begin{aligned}\min_{\tilde{\gamma}^r,\,r=1,2,\ldots,R}[\overline{Q}_{\mathrm{C}}^{\mathrm{E}}(\tilde{\gamma};\tilde{\gamma}_0] &= \min_{\tilde{\gamma}^r,\,r=1,2,\ldots,R}\left\{\max_{r=1,2,\ldots,R}\{\overline{Q}_{\mathrm{C}}^{\mathrm{E},r}(\tilde{\gamma};\tilde{\gamma}_0)\}\right\}\\ &= \max_{r=1,2,\ldots,R}\{\min_{\tilde{\gamma}^r}[\overline{Q}_{\mathrm{C}}^{\mathrm{E},r}(\tilde{\gamma};\tilde{\gamma}_0)]\} = \max_{r=1,2,\ldots,R}\{\overline{Q}_{\mathrm{C}}^{\mathrm{E},r,*}(\tilde{\gamma}_0)\} = \tilde{Q}_{\mathrm{C}}^{\mathrm{E},*}(\tilde{\gamma}_0).\end{aligned} \tag{2.178}$$

Do rozwiązania problemów optymalizacyjnych z drugiego i trzeciego kroku przedstawionego algorytmu rozwiązania można stosować różne algorytmy rozwią-

zujące konkretne probabilistyczne zagadnienia szeregowania i komiwojażera. Można w tym celu stosować też procedurę determinizacji, o której wspomnieliśmy w p. 1.4.1.

Ocena algorytmu rozwiązania

Podobnie jak w przypadku deterministycznym podajemy oszacowanie jakości zaprezentowanego algorytmu rozwiązania w stosunku do rozwiązania optymalnego.

Twierdzenie 2.7

Niech $Q_{\mathrm{C}}^{\mathrm{E},*}$ będzie optymalną wartością kryterium jakości dla problemu PC w wersji probabilistycznej. Wtedy

$$\begin{aligned} \widetilde{Q}_{\mathrm{C}}^{\mathrm{E}}(\widetilde{\gamma}_0) - Q_{\mathrm{C}}^{\mathrm{E},*} \le & \mathop{\mathrm{E}}_{\underline{\hat{\tau}}^{\max},\underline{\hat{\tau}}^{\min}} \Big(\max_{r=1,2,\ldots,R} \Big\{ \sum_{h=1}^{H+1} (\underline{\hat{\tau}}_{r,\cdot,h}^{\max} - \underline{\hat{\tau}}_{r,\cdot,h}^{\min}) \Big\} \Big) \\ & + \mathop{\mathrm{E}}_{\substack{\underline{\hat{\tau}}_{r,g^*,H+1} \\ r=1,2,\ldots,R \\ g=1,2,\ldots,H}} \Big(\max_{r=1,2,\ldots,R} \{\underline{\hat{\tau}}_{r,g^*,H+1}\} - \min_{r=1,2,\ldots,R} \{\underline{\hat{\tau}}_{r,g^*,H+1}\} \Big), \end{aligned} \tag{2.179}$$

gdzie

$$\hat{\tau}_{r,\cdot,h}^{\max} = \max_{\substack{g=1,2,\ldots,H+1 \\ g \neq h}} \{\hat{\tau}_{r,g,h}\}, \qquad \hat{\tau}_{r,\cdot,h}^{\min} = \min_{\substack{g=1,2,\ldots,H+1 \\ g \neq h}} \{\hat{\tau}_{r,g,h}\},$$

$$\hat{\boldsymbol{\tau}}^{\max} = [\hat{\tau}_{r,\cdot,h}^{\max}]_{\substack{r=1,2,\ldots,R \\ h=1,2,\ldots,H+1}}, \qquad \hat{\boldsymbol{\tau}}^{\min} = [\hat{\tau}_{r,\cdot,h}^{\min}]_{\substack{r=1,2,\ldots,R \\ h=1,2,\ldots,H+1}}$$

oraz g^* jest stanowiskiem, dla którego $\widetilde{\gamma}_{0,g^*,H+1}^{r} = 1$.

Dowód

Postać oszacowania (2.179) jest związana z dwiema niezależnymi przyczynami pogorszenia jakości rozwiązania, otrzymywanego przez proponowany algorytm rozwiązania. Są one podobne jak w przypadku deterministycznym (twierdzenie 2.2) i są spowodowane ustaleniem w arbitralny sposób macierzy $\widetilde{\gamma}_0$ w pierwszym kroku algorytmu oraz rozwiązywaniem uproszczonego problemu szeregowania w drugim kroku algorytmu. Rozpocznijmy od rozpatrzenia pierwszej przyczyny. Ustalenie macierzy $\widetilde{\gamma}_0$, może sprawić, że przyjęte czasy dojazdów realizatorów do stanowisk, będące danymi dla problemu szeregowania w drugim kroku algorytmu, będą się różnić od odpowiednich czasów optymalnych. Różnica miedzy tymi czasami nie jest większa niż $\hat{\tau}_{r,g,h}^{\max} - \hat{\tau}_{r,g,h}^{\min}$. Wtedy różnica między wartościami kryteriów dla rozpa-

trywanych przypadków wyraża oceniane pogorszenie jakości i może być oszacowana od góry następująco

$$\underset{\underline{\hat{\mathbf{T}}}^{\max}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H+1} c_{r,h}\,\underline{\hat{\tau}}^{\max}_{r,\cdot,h}\Big\}\Big) - \underset{\underline{\hat{\mathbf{T}}}^{\min}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H+1} c_{r,h}\,\underline{\hat{\tau}}^{\min}_{r,\cdot,h}\Big\}\Big)$$

$$= \underset{\underline{\hat{\mathbf{T}}}^{\max},\underline{\hat{\mathbf{T}}}^{\min}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H+1} c_{r,h}\,\underline{\hat{\tau}}^{\max}_{r,\cdot,h}\Big\} - \max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H+1} c_{r,h}\,\underline{\hat{\tau}}^{\min}_{r,\cdot,h}\Big\}\Big) \le \underset{\underline{\hat{\mathbf{T}}}^{\max},\underline{\hat{\mathbf{T}}}^{\min}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}$$

$$\Big\{\sum_{h=1}^{H+1} c_{r,h}\,\underline{\hat{\tau}}^{\max}_{r,\cdot,h} - \sum_{h=1}^{H+1} c_{r,h}\,\underline{\hat{\tau}}^{\min}_{r,\cdot,h}\Big\}\Big) = \underset{\underline{\hat{\mathbf{T}}}^{\max},\underline{\hat{\mathbf{T}}}^{\min}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H+1}(\underline{\hat{\tau}}^{\max}_{r,\cdot,h} - \underline{\hat{\tau}}^{\min}_{r,\cdot,h})\Big\}\Big). \tag{2.180}$$

Przechodząc do drugiej przyczyny pogorszenia jakości otrzymywanego rozwiązania, przypomnijmy, że wspomniane uproszczenie polega na pominięciu w procedurze szeregowania zjazdów realizatorów do bazy. Wyjściowy problem szeregowania jako problem optymalizacyjny (2.172) można oszacować w taki sposób

$$\begin{aligned}
&\min_{\boldsymbol{c}} \tilde{Q}^{\mathrm{E}}_{\mathrm{C}}(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}) = \min_{\boldsymbol{c}} \underset{\underline{\hat{\tau}},\underline{\bar{\tau}}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H+1} c_{r,h}\,\underline{\tilde{T}}_{r,h}(\tilde{\gamma}_0^r)\Big\}\Big)\\
&= \min_{\boldsymbol{c}}\Big[\underset{\underline{\hat{\tau}},\underline{\bar{\tau}}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H} c_{r,h}\,\underline{\tilde{T}}_{r,h}(\tilde{\gamma}_0^r) + c_{r,H+1}\,\underline{\tilde{T}}_{r,H+1}(\tilde{\gamma}_0^r)\Big\}\Big)\\
&\le \min_{c}\Big[\underset{\underline{\bar{\tau}},\underline{\hat{\tau}}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H} c_{r,h}\,\underline{\tilde{T}}_{r,h}(\tilde{\gamma}_0^r)\Big\}\Big) + \underset{\substack{\underline{\hat{\tau}}_{r,g,H+1}\\ r=1,2,\ldots,R\\ g=1,2,\ldots,H}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\{\underline{\tilde{T}}_{r,H+1}(\tilde{\gamma}_0^r)\}\Big)\Big],
\end{aligned} \tag{2.181}$$

gdzie

$$\underline{\tilde{T}}_{r,H+1} = \sum_{g=1}^{H} \tilde{\gamma}^r_{0,g,H+1}\,\underline{\hat{\tau}}_{r,g,H+1}. \tag{2.182}$$

Wartość oczekiwana $\underset{\substack{\underline{\hat{\tau}}_{r,g,H+1}\\ r=1,2,\ldots,R\\ g=1,2,\ldots,H}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\{\underline{\tilde{T}}_{r,H+1}(\tilde{\gamma}_0^r)\}\Big) \stackrel{\Delta}{=} D$ jako niezależna od $\boldsymbol{c}$ nie ma wpływu na minimalizację. Stąd

$$\min_{\boldsymbol{c}} \tilde{Q}^{\mathrm{E}}_{\mathrm{C}}(\boldsymbol{c},\tilde{\boldsymbol{\gamma}}) \le \min_{\boldsymbol{c}} \underset{\underline{\bar{\tau}},\underline{\hat{\tau}}}{\mathrm{E}}\Big(\max_{r=1,2,\ldots,R}\Big\{\sum_{h=1}^{H+1} c_{r,h}\,\underline{\tilde{T}}_{r,h}(\tilde{\gamma}_0^r)\Big\}\Big) + D.$$

Tak więc różnica miedzy wartościami kryterium jakości przed i po uproszczeniu nie jest większa niż D, czyli od wartości oczekiwanej najdłuższego zjazdu do bazy. Uzyskane oszacowanie może być jeszcze zmniejszone. Zauważmy bowiem, że czas powrotu realizatora do bazy jest nie mniejszy niż

$$\mathop{\mathrm{E}}_{\substack{\hat{\underline{\tau}}_{r,g,H+1} \\ r=1,2,...,R \\ g=1,2,...,H}} (\min_{r=1,2,...,R} \{\tilde{\underline{T}}_{r,H+1}(\tilde{\gamma}_0^r)\}).$$

Ostatecznie pogorszenie jakości rozwiązania można od góry oszacować jako

$$\mathop{\mathrm{E}}_{\substack{\hat{\underline{\tau}}_{r,g,H+1} \\ r=1,2,...,R \\ g=1,2,...,H}} (\max_{r=1,2,...,R} \{\sum_{g=1}^{H} \tilde{\gamma}_{0,g,H+1}^r \hat{\underline{\tau}}_{r,g,H+1}\} - \min_{r=1,2,...,R} \{\sum_{g=1}^{H} \tilde{\gamma}_{0,g,H+1}^r \hat{\underline{\tau}}_{r,g,H+1}\}). \quad (2.183)$$

W przypadku gdy macierze $\tilde{\gamma}_0^r$ są binarne, oszacowanie (2.183) przyjmuje prostszą postać, to znaczy

$$\mathop{\mathrm{E}}_{\substack{\hat{\underline{\tau}}_{r,g^*,H+1} \\ r=1,2,...,R \\ g=1,2,...,H}} (\max_{r=1,2,...,R} \{\hat{\underline{\tau}}_{r,g^*,H+1}\} - \min_{r=1,2,...,R} \{\hat{\underline{\tau}}_{r,g^*,H+1}\}), \quad (2.184)$$

gdzie g^* ma znaczenie takie jak w sformułowaniu twierdzenia. Uzasadnienie niezależności obu przyczyn pogorszenia jakości jest takie samo jak w przypadku deterministycznym. Dlatego sumując oszacowanie z (2.180) oraz wyrażenie (2.184) otrzymujemy oszacowanie (2.179).

■

Zakończmy rozważania dwiema uwagami.

1. Przyjęcie, że czasy wykonywania zadań są realizacjami zmiennych losowych, czyli rozpatrywanie przypadku probabilistycznego, to tylko jeden ze sposobów opisu niedeterminizmu problemu szeregowania, polegającego na braku dokładnej informacji o tych czasach. Spośród innych możliwych sposobów wskażemy tylko na dwa. W pierwszym z nich zakłada się, że czasy są liczbami rozmytymi, a informacje o nich określono w postaci tak zwanych funkcji przynależności. Funkcje te, podobnie jak funkcje gęstości dla przypadku probabilistycznego, muszą być dane (znane) a priori. Istnieje jednak zasadnicza różnica interpretacyjna między oboma sposobami opisu niedeterminizmu. Podejście to jest obecnie żywo rozwijane dla różnych konkretnych problemów szeregowania, a zastosowanie go do problemu szeregowania z ruchomymi realizatorami jest sprawą otwartą.

Drugi sposób opisu niedeterminizmu, o którym warto wspomnieć, polega na podaniu informacji o czasach wykonywania zadań w postaci przedziałów o ustalonych i znanych początkach i końcach. Kryterium jakości szeregowania ma wtedy postać minimaksową, czyli należy wybrać najlepsze uszeregowanie dla odpowiednio zdefiniowanego najgorszego przypadku ze względu na czas wykonywania zadań, np. [36]. Sposób ten jest szczególnie uzasadniony dla szeregowania z ruchomymi realizatorami w przypadku, gdy dotyczy czasu dojazdu realizatora do stanowiska. Określenie czasu w postaci funkcji przynależności, a zwłaszcza w postaci funkcji gęstości jest znacznie trudniejsze.

2. Niedeterminizm szeregowania z uwzględnieniem ruchu realizatorów (ale również innych problemów szeregowania) nie musi dotyczyć czasów wykonania zadań. Innym przykładem może być sytuacja, w której nie jest ustalona liczba stanowisk, na których są wykonywane zadania, to znaczy nie wszystkie zadania muszą być wykonane. Zastosowanie podejścia probabilistycznego, jak na przykład w pracy [48] dla problemu komiwojażera, polegałoby na przyjęciu opisu, według którego każde zadanie jest wykonywane z określonym prawdopodobieństwem. Zakres zastosowań wykracza w tym przypadku poza systemy produkcyjne i może obejmować na przykład systemy transportowe lub usługowe.

3. Podejmowanie decyzji w kompleksach operacji z uwzględnieniem ruchu oraz transportu

3.1. Wprowadzenie

Zagadnienia prezentowane w tym rozdziale są w pewnym sensie uogólnieniem tych problemów, które przedstawiono w rozdziale 1, a przede wszystkim w rozdziale 2. W poprzednim rozdziale rozważano rozszerzenie klasycznego problemu szeregowania, polegające na uwzględnieniu w procesie decyzyjnym ruchu realizatorów wykonujących zadania. Pozostając przy interpretacji z zakresu dyskretnych procesów produkcyjnych, dodajmy, że chodziło w tym przypadku o zadania produkcyjne lub inaczej technologiczne, czyli takie, których wykonanie w sposób bezpośredni wpływa na postęp w produkcji wytwarzanego obiektu. Uwzględnienie ruchu realizatorów faktycznie polegało na rozpatrywaniu jeszcze innego rodzaju zadań, które w rozdziale 2. nazwaliśmy dojazdami realizatorów do stanowisk. Ogólnie, chodziłoby o takie zadania, które są związane z ruchem. Będziemy je dalej nazywać zadaniami transportowymi. Zadania takie mogą dotyczyć różnych elementów systemu produkcyjnego, a mianowicie samych realizatorów, wytwarzanych obiektów, ale również innych elementów, jak narzędzia, materiały, detale, palety, zamocowania, które łącznie nazywamy zasobami dodatkowymi. W przypadku realizatorów i obiektów mówimy o ich ruchu, a nie transporcie. Dla zasobów dodatkowych jest uzasadnione używanie określenia transport. Dla każdego zadania musi być określony podmiot, który je wykonuje. Ruch realizatorów ma swoją specyfikę polegającą na tym, że realizator jest tym podmiotem, który wykonuje zarówno zadania technologiczne, jak i zadania transportowe – związane z ruchem realizatora. Przykładem może być ruchomy robot, którego część manipulacyjna może służyć do wykonywania zadań technologicznych, układ jezdny natomiast do poruszania się. Dobrym przykładem jest tu również człowiek uczestniczący w wykonywaniu procesu produkcyjnego – w przypadku braku jego automatyzacji. Podczas ruchu innych elementów systemu produkcyjnego podmioty wykonujące zadania technologiczne i zadania transportowe są już różne. Dla zadań technologicznych są nimi realizato-

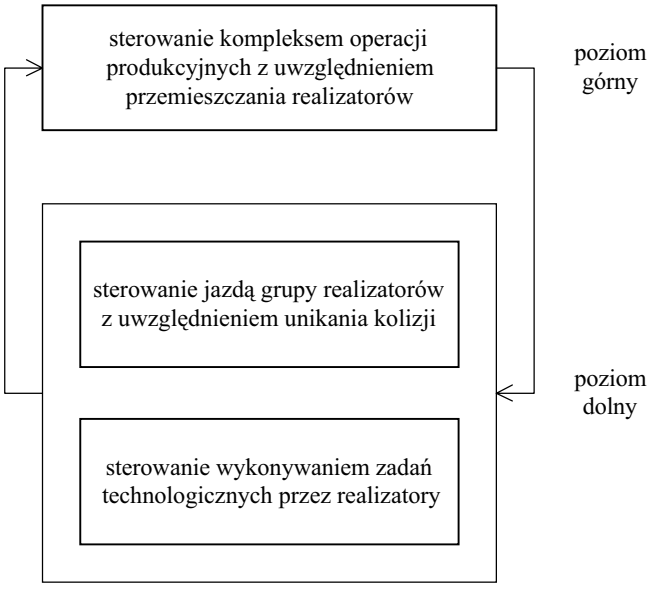
ry, a dla zadań transportowych podmioty (zwykle urządzenia) nazywane ogólnie środkami transportu. Wtedy mówimy o transporcie. Wyjątkiem jest ruch obiektów. Obiekty są tymi przedmiotami ruchu, na których są wykonywane zarówno zadania technologiczne, jak i zadania transportowe. Tak więc o ruchu mówimy wtedy, gdy jego podmioty lub przedmioty są takie same zarówno dla zadań technologicznych, jak i transportowych. W przeciwnym razie można wyróżnić wykonywanie zadań technologicznych przez realizatory oraz transport zasobów dodatkowych. Wówczas zadania transportowe są wykonywane przez środki transportu.

W klasycznych problemach decyzyjnych dla kompleksów operacji decyzje dotyczące wykonywania zadań technologicznych i zadań transportowych były rozwiązywane niezależnie lub problemy były ograniczane do rozpatrywania zadań technologicznych. Problematykę uwzględnienia ruchu realizatorów w wybranych zagadnieniach szeregowania zadań obszernie przedstawiono w rozdziale 2. Zastosowano tam prostszą i bardziej jednoznaczną nomenklaturę. Zadanie transportowe nazwano dojazdem realizatora do stanowiska, a zadanie technologiczne – czynnością. Oba z nich tworzą zadanie. Problematyka szeregowania zadań technologicznych i transportowych w przypadku ruchu obiektów jest poruszana w wielu pracach, np. [11, 83, 107, 103, 104, 118]. Ze względu na złożoność problemów, zwykle stosuje się dekompozycję prowadzącą do niezależnego szeregowania obu typów zadań. Oryginalny sposób rozwiązania tego zagadnienia przedstawiono w pracy [96]. Powiązanie ruchu obiektów i (lub) realizatorów z innymi problemami decyzyjnymi dla kompleksów operacji pozostaje sprawą otwartą. Interesujące i ważne z praktycznego punktu widzenia jest także powiązanie klasycznych zagadnień sterowania kompleksami operacji z transportem. W dalszej części przedstawiono zagadnienie alokacji zadań produkcyjnych w powiązaniu z transportem surowców do produkcji oraz z transportem produktów.

Precyzując pojęcie ruchu, zwróćmy uwagę na dwa jego możliwe znaczenia, które odnoszą się do ruchu realizatorów i do ruchu środków transportu. Pierwsze polega na p r z e m i e s z c z a n i u realizatorów lub środków transportu w celu wykonania kolejnych zadań. Przemieszczanie podczas ruchu środków transportu polega na określeniu, między którymi elementami systemu produkcyjnego (najczęściej realizatorami) ma być przetransportowany obiekt lub zasób dodatkowy, który ze środków transportu ma tego dokonać i kiedy ma się ono rozpocząć. W przypadku ruchu realizatorów przemieszczanie wiąże się z wyznaczaniem tras przejazdów realizatorów między stanowiskami, w których są wykonywane kolejne zadania technologiczne. Ruch w drugim znaczeniu polega na j e ź d z i e realizatora lub środka transportu między zadanymi położeniami: początkowym i końcowym, w określonej

przestrzeni roboczej, w której jest wykonywany proces produkcyjny. Proces decyzyjny polega w tym przypadku na sterowaniu mechanizmem jazdy realizatora lub środka transportu i ma zupełnie inny charakter niż w przypadku ruchu w pierwszym wymienionym znaczeniu. Proces taki jest nazywany sterowaniem jazdą realizatora lub środka transportu (ang. *movement control* lub *motion control*). Z tym zakresem znaczeniowym pojęcia ruch wiąże się jeszcze jeden problem sterowania, polegający na koordynacji jazdy poruszających się realizatorów lub środków transportu (ang. *traffic control*), zapewniającej ruch bezkolizyjny.

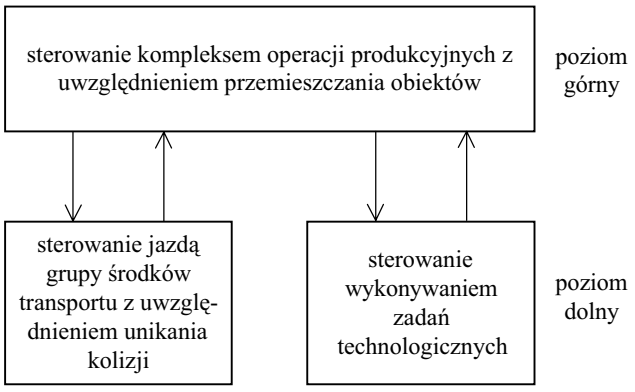
W elastycznych systemach produkcyjnych, traktowanych jako kompleksy operacji produkcyjnych, problemy decyzyjne, związane z oboma znaczeniami ruchu, a także z wykonywaniem zadań technologicznych, są wzajemnie zależne i należy je rozpatrywać łącznie. Dla ruchu realizatorów tworzą one system dwupoziomowy z wybranym zagadnieniem sterowania kompleksem operacji, uwzględniającym przemieszczanie się realizatorów – na poziomie górnym, oraz ze sterowaniem jazdą realizatorów wraz z jej koordynacją i sterowaniem wykonywaniem zadań technologicznych – na poziomie dolnym. Ogólny schemat blokowy takiego dwupoziomowego systemu przedstawiono na rys. 3.1. Wielkościami, wiążącymi problemy sterowania



Rys. 3.1. Schemat blokowy dwupoziomowego systemu sterowania kompleksem operacji produkcyjnych z uwzględnieniem ruchu realizatorów

na obu poziomach, są trasy poruszania się (przemieszczania) realizatorów oraz czasy wykonywania zadań. Pierwsza z wymienionych wielkości jest wynikiem decyzji na poziomie górnym i określa dla każdego realizatora ciąg położeń, między którymi ma się on przemieszczać w celu wykonywania zadań technologicznych. Sąsiednie elementy ciągu tworzą pary. Każda taka para, czyli położenie początkowe i końcowe, jest daną dla problemu sterowania jazdą. Ustalanie ciągu położeń dla wszystkich realizatorów ma wpływ na wynik decyzji podejmowanych na poziomie dolnym, przede wszystkim z tego powodu, że dla różnych ciągów położeń są różne potencjalne możliwości kolizji realizatorów. Konieczność ich unikania wpływa na uzyskiwane czasy jazdy. Rezultaty decyzji z poziomu dolnego, czyli czasy wykonywania zadań transportowych oraz czasy wykonywania zadań technologicznych, będących rezultatami sterowania wykonywaniem zadań technologicznych, są z kolei danymi dla problemu rozwiązywanego na poziomie górnym i określają w ten sposób postaci uzyskiwanych tras przemieszczania się realizatorów.

Różne procesy decyzyjne związane z ruchem można również przedstawić w formie systemu dwupoziomowego z wybranym zagadnieniem sterowania kompleksem operacji, uwzględniającym przemieszczanie się obiektów – na poziomie górnym, oraz ze sterowaniem jazdą środków transportu wraz z ich koordynacją i sterowaniem wykonywaniem zadań technologicznych przez realizatory niezależne od środków transportu – na poziomie dolnym (rys. 3.2). Na poziomie górnym są podejmowane decyzje dotyczące wykonywania zadań technologicznych i transportowych, polegające w szczególności na określaniu momentów czasu, w których ma się rozpocząć wykonywanie zadań oraz dla zadań technologicznych na wyborze realizatorów, które



Rys. 3.2. Schemat blokowy dwupoziomowego systemu sterowania kompleksem operacji produkcyjnych z uwzględnieniem ruchu obiektów

wykonują dane zadania, a dla zadań transportowych na wyborze środków transportu i miejsc, do których należy dojechać. Dane te są przekazywane na poziom dolny. Tam z kolei odpowiednie problemy decyzyjne są podobne jak dla ruchu realizatorów, dotyczą jednak różnych podmiotów wykonujących zadania i mogą być rozpatrywane niezależnie. Czasy wykonywania zadań są danymi niezbędnymi do podejmowania decyzji na poziomie górnym.

Uwzględnienie sterowania jazdą w przypadku ruchu środków transportu przemieszczających zasoby dodatkowe nie prowadzi do nowych problemów, jeśli zostanie zastosowana interpretacja odwołująca się do ruchu obiektów lub ruchu realizatorów, w zależności od własności środków transportu. Wystarczy przyjąć, że odpowiednikiem zadania technologicznego jest z a d a n i e u s ł u g o w e, polegające na odbiorze lub dostarczeniu przez realizator zasobu z/lub na środek transportu, a odpowiednikiem obiektu jest transportowany zasób. Rozpatrywany problem staje się wówczas równoważny scharakteryzowanemu wcześniej złożonemu problemowi decyzyjnemu z uwzględnieniem ruchu obiektów. W przypadku, gdy środek transportu jest zaopatrzony w urządzenie do załadunku (rozładunku) transportowanego zasobu, wówczas można go traktować jako realizator wykonujący zadania usługowe i w konsekwencji rozwiązywać odpowiednie zagadnienie z uwzględnieniem ruchu realizatorów.

Reasumując te wstępne rozważania, dotyczące uwzględniania ruchu w problemach decyzyjnych dla kompleksów operacji, wymieńmy te pojęcia, które są istotne dla formułowania nowych problemów decyzyjnych.

1. Rozpatrujemy dwa typy zadań: technologiczne i transportowe.

2. Określamy podmioty i przedmioty ruchu. Gdy dla zadań technologicznych i transportowych zarówno przedmioty, jak i podmioty ruchu są różne, wówczas używamy terminu „transport", a w przeciwnym razie mówimy o ruchu. Wyszczególniamy więc następujące rodzaje ruchu: ruch realizatorów, ruch obiektów i transport zasobów dodatkowych.

3. Rozważamy dwa znaczenia ruchu: przemieszczanie i jazdę.

Połączenie różnych rodzajów ruchu z różnymi jego znaczeniami pozwala na wyróżnienie kilku istotnych problemów badawczych dla kompleksów operacji produkcyjnych.

1. Sterowanie kompleksem operacji z uwzględnieniem ruchu realizatorów:

a) dla ruchu rozumianego w sensie przemieszczania realizatorów,

b) dla ruchu rozumianego w sensie przemieszczania i jazdy realizatorów.

2. Sterowanie kompleksem operacji z uwzględnieniem ruchu obiektów:

a) dla ruchu rozumianego w sensie przemieszczania obiektów,

b) dla ruchu rozumianego w sensie przemieszczania obiektów i jazdy środków transportu.

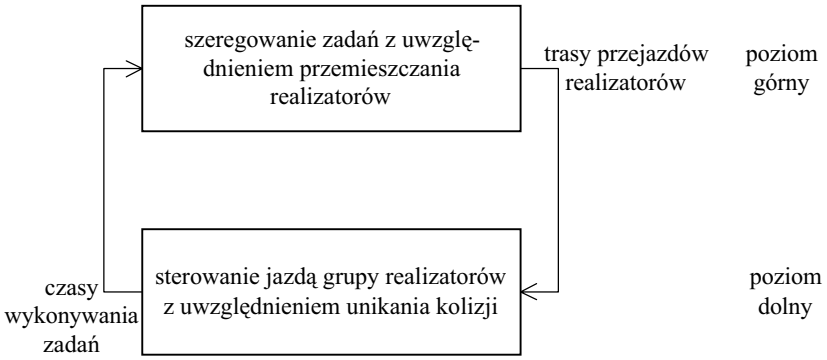
3. Sterowanie kompleksem operacji z uwzględnieniem transportu zasobów dodatkowych dla ruchu środków transportu, rozumianego w sensie przemieszczania.

Wymienione w punktach 1, 2 i 3 pojęcie „sterowanie kompleksem operacji" może oznaczać różne konkretne problemy decyzyjne, co może prowadzić do zwiększenia liczby szczegółowych złożonych problemów decyzyjnych, dla których należy wyznaczać algorytmy rozwiązania.

W kolejnych punktach rozdziału omówiono problemy 1b, 2a oraz 3. Tak więc treść rozdziału jest zarówno rozszerzeniem i uogólnieniem (punkt 3.2), jak i uzupełnieniem (punkty 3.3 i 3.4) treści zawartych w rozdziałach 1 i 2.

3.2. Dwupoziomowy system sterowania kompleksem operacji z uwzględnieniem przemieszczania i jazdy realizatorów

Niniejszy punkt nawiązuje do rozdziału 2, ponieważ jako problem sterowania kompleksem operacji w systemie dwupoziomowym przyjęto ten sam problem szeregowania, który był tam rozważany. Przypomnijmy, że jest to zagadnienie szeregowania zadań niezależnych i niepodzielnych o równych momentach gotowości na realizatorach dowolnych w celu minimalizacji długości uszeregowania – dla założenia, że realizatory są ruchome, czyli w celu wykonania czynności (zadania technologicznego) muszą dojechać do odpowiedniego stanowiska (czyli wykonać zadanie transportowe). Rozszerzenie i uogólnienie tego zagadnienia, będące przedmiotem rozważań, polega na rozpatrywaniu dodatkowo zagadnienia sterowania bezkolizyjną jazdą grupy realizatorów. Oba zagadnienia – zgodnie z uzasadnieniem podanym w poprzednim punkcie – są zależne i należy je rozpatrywać łącznie. Tworzą one system dwupoziomowy. Gdy założymy, że nie będziemy rozpatrywać zagadnień sterowania wykonywaniem zadań technologicznych, wówczas dwupoziomowy system sterowania kompleksem operacji, którego ogólna postać została wprowadzona w punkcie 3.1 (rys. 3.1), ma teraz postać przedstawioną na rys. 3.3 i w tej wersji będzie przedmiotem rozważań. Zagadnienie decyzyjne z poziomu górnego w skrócie będziemy nazywać podproblemem szeregowania, a zagadnienie z poziomu dolnego – podproblemem sterowania jazdą grupy realizatorów lub, jeśli nie będzie to powodowało niejasności, podproblemem sterowania. Pojęcia: stanowisko, realizator, zadanie, czynność, dojazd realizatora do stanowiska, z uwzględnieniem uwag podanych w punkcie 3.1, związanych z ogólną koncepcją ruchu w kompleksie operacji – mają takie same znaczenie jak w rozdziale 2. Rozpatrywany złożony problem decy-



Rys. 3.3. Schemat blokowy dwupoziomowego systemu sterowania kompleksem operacji produkcyjnych dla szeregowania zadań i z uwzględnieniem przemieszczania i jazdy realizatorów

zyjny dla kompleksu operacji, czyli problem szeregowania z ruchomymi realizatorami może dotyczyć przypadków w wybranych dyskretnych systemach produkcyjnych. Chodzi o takie przypadki, w których po pierwsze, ruch wytwarzanego obiektu jest niewskazany lub wręcz niemożliwy ze względu na jego rozmiary czy konieczność wykorzystywania specjalistycznych zasobów niezbędnych do jego wytworzenia, a zlokalizowanych na stanowisku, na którym obiekt się znajduje. Po drugie zaś, jednocześnie jest wytwarzana pewna liczba takich obiektów, a zbiór realizatorów jest mniej liczny niż zbiór wytwarzanych obiektów – może to być spowodowane dużymi kosztami realizatora. Realizatory muszą „obsłużyć" większą liczbę obiektów. Rozważamy takie sytuacje, w których na każdym stanowisku jest wykonywane tylko jedno zadanie produkcyjne, a więc nie rozpatrujemy całego cyklu wytwarzania obiektu, ale jego fragment, polegający na wykonaniu jednego określonego zadania produkcyjnego. Przykładem może być tu z kolei specjalistyczne gniazdo produkcyjne, obsługiwane przez jednofunkcyjne realizatory. Realizatory zadań produkcyjnych mogą mieć różną naturę. Mogą nimi być inteligentne urządzenia techniczne, takie jak np. roboty manipulacyjne lub roboty ruchome, samodzielnie wykonujące zadania produkcyjne. Scharakteryzowany system produkcyjny jest wtedy elastycznym systemem produkcyjnym. Bardziej konkretnym przykładem może być wtedy komórka produkcyjna do zgrzewania punktowego obudów pralek automatycznych [50, 58], traktowanych jako obiekty umieszczone w specjalistycznych stanowiskach wyposażonych w odpowiednie zamocowania, umożliwiające ich pozycjonowanie. Realizatorami są pistolety do zgrzewania punktowego, przemieszczane ręcznie lub

automatycznie między stanowiskami, z których każdy może wykonać przewidziane wymogami technologicznymi czynności zgrzewania na każdym elemencie (obiekcie), znajdującym się w komórce produkcyjnej. Ogólnie, tak jak w przytoczonym przykładzie, czynność wykonywana przez realizator nie musi być jednorodna, ale może się składać z pewnej liczby czynności elementarnych. Nie rozpatrujemy ich jednak oddzielnie, ale traktujemy je jako jedną czynność. Realizatorami zadań produkcyjnych mogą być również ludzie. Mamy wówczas do czynienia z bardziej tradycyjnym dyskretnym systemem produkcyjnym. Przykładem zastosowania rozpatrywanego zagadnienia w takich systemach może być oddział produkcyjny z tradycyjnymi obrabiarkami sterowanymi numerycznie, obsługiwany przez wykwalifikowanych robotników, których zadaniem jest przezbrojenie (przygotowanie) obrabiarek do produkcji nowego typu obiektu. Minimalizacja czasu przezbrojenia wszystkich obrabiarek wymaga uwzględnienia czasu przemieszczania się robotników między nimi. Zagadnienie takie wiąże się również z problematyką zarządzania w systemach produkcyjnych.

Omawiając dyskretne systemy produkcyjne i charakterystyki możliwych zastosowań rozpatrywanego w pracy problemu szeregowania, zwróćmy uwagę, że zastosowania te nie muszą dotyczyć zadań produkcyjnych, ale mogą się również odnosić do zadań usługowych (pomocniczych), np. do transportu detali do stanowisk montażu w systemie montażowym. Realizatorami mogą być wtedy pojazdy autonomiczne (ang. AGV) lub ruchome roboty wyposażone w urządzenia manipulacyjne. Osiągnięcie jak najkrótszego czasu dostarczenia detali do montażu do określonej liczby stanowisk montażowych wymaga uwzględnienia nie tylko czasu rozładunku detali na stanowiskach, traktowanego jako zadanie produkcyjne, ale również czasu dojazdu do tych stanowisk. Zastosowania w różnego rodzaju systemach produkcyjnych wydają się najbardziej oczywiste i intuicyjne, ale nie są jedyne. Rozpatrywane w zagadnieniu szeregowania zbiory zadań i realizatorów mogą mieć również inny charakter. Specyfika ruchu realizatorów sprawia, że mogą być np. brane pod uwagę wszystkie zastosowania typowe dla problemu komiwojażera (lub wielu komiwojażerów), np. [98]. Ponieważ są one powszechnie znane, nie będziemy ich tu szerzej omawiać. Zwrócimy tylko uwagę na oczywistą różnicę między problemem szeregowania z ruchomymi realizatorami a problemem wielu komiwojażerów, wyrażającą się w innych kryteriach jakości. W pierwszym przypadku dążymy do minimalizacji czasu realizacji zbioru zadań, a w drugim do minimalizacji kosztu przejazdów komiwojażerów, który w szczególnym przypadku może być proporcjonalny do sumy czasów przejazdów.

3.2.1. Sterowanie jazdą grupy realizatorów

Obiektem sterowania są mechanizmy jazdy poszczególnych realizatorów. Jest to złożony obiekt wejściowo-wyjściowy. Cel sterowania nie jest jednorodny i składają się na niego dwa cele cząstkowe: osiągnięcie przez poszczególne realizatory zadanych stanów końcowych dla założenia znajomości stanów początkowych tak, aby minimalizować określone kryterium jakości sterowania (jazdy) oraz uzyskanie przejazdów bezkolizyjnych. Omówieniu tego typu problemów sterowania dla konkretnych poruszających się obiektów, np. robotów mobilnych lub pojazdów autonomicznych jest poświęconych wiele specjalistycznych opracowań ([84]), w których w różny sposób dokonuje się formalizacji oraz stosuje się różnorodne sposoby wyznaczania algorytmów rozwiązania. Ze względu na charakter podproblemu sterowania jazdą, odbiegający od głównego nurtu rozważań, zostanie zaprezentowana oryginalna metoda rozwiązania, wcześniej przedstawiona w pracach [51–56, 76], w zakresie niezbędnym do precyzyjnego przedstawienia problemu dla systemu dwupoziomowego i bez porównania z innymi stosowanymi formalizmami, metodami i algorytmami rozwiązania.

Model mechanizmu jazdy

W celu konkretyzacji rozważań przyjmujemy założenie, że mechanizmy jazdy wszystkich realizatorów są takie same. Układ jezdny jest trójkołowy. Przednie koło może się obracać wokół osi, powodując ruch postępowy układu oraz może się skręcać w celu zmiany kierunku jazdy. Dwa tylne koła są zamocowane na wspólnej osi i mogą wykonywać obrót wokół niej. Do wyznaczania algorytmu sterowania jazdą jest stosowany uproszczony model mechanizmu jazdy (obiektu sterowania), uwzględniający własności dynamiczne, dla którego przyjęto następujące założenia upraszczające: obiekt jest bryłą sztywną, nie są uwzględniane siły tarcia oraz masa całkowita obiektu jest skupiona punktowo nad osią koła przedniego. Jako wielkości sterujące dla bieżącego, czyli r-tego realizatora przyjęto moment siły napędzający $u_r^{(1)}(t)$ powodujący przesunięcie liniowe obiektu oraz moment siły skręcający $u_r^{(2)}(t)$, zapewniający skręt koła przedniego i w konsekwencji zmianę kierunku jazdy. Obie wielkości tworzą wektor wielkości sterujących $\boldsymbol{u}_r(t) = [u_r^{(1)}(t),\ u_r^{(2)}(t)]^{\mathrm{T}}$. Pomijając skręt koła przedniego, położenie obiektu może być jednoznacznie określone przez współrzędne dwóch punktów w układzie współrzędnych kartezjańskich, na przykład punktu, w którym znajduje się oś koła przedniego oraz punktu będącego środkiem odcinka łączącego koła tylne. Maksymalna liczba stopni swobody dla dwóch punktów na płaszczyźnie wynosząca cztery musi być pomniejszona o trzy, ponieważ koła

znajdujące się w rozważanych punktach wnoszą dwa więzy do układu oraz ze względu na przyjęte założenie, odległość między punktami (kołami) jest stała. Tak więc ostatecznie, układ ma jeden stopień swobody bez uwzględnienia kąta skrętu koła przedniego. Biorąc pod uwagę ten kąt, otrzymujemy dwa stopnie swobody.

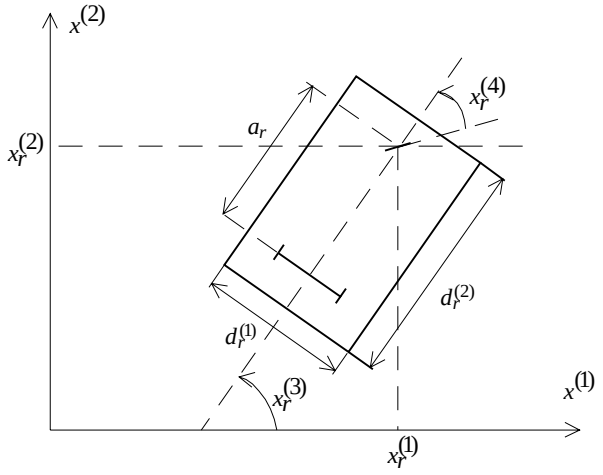
Inną wygodniejszą formą opisu położenia układu jest zestaw: punkt, w którym znajduje się oś koła przedniego oraz kąt nachylenia osi realizatora, przechodzącej przez oba poprzednio wprowadzone punkty, w stosunku do osi odciętych układu współrzędnych. Kąt nachylenia jest kątem skierowanym, wyznaczonym przy założeniu, że oś realizatora jest skierowana od kół tylnych do koła przedniego. Wtedy można podać następujący zestaw współrzędnych, jednoznacznie opisujących położenie *r*-tego realizatora, dla $r = 1, 2, ..., R$, gdzie R jest liczbą wszystkich poruszających się realizatorów – nazywanych też współrzędnymi pierwotnymi (rys. 3.4)

$x_r^{(1)}$, $x_r^{(2)}$ – współrzędne położenia osi koła przedniego w kartezjańskim układzie współrzędnych,

$x_r^{(3)}$ – kąt między osią realizatora a osią odciętych układu kartezjańskiego,

$x_r^{(4)}$ – kąt skrętu, czyli kąt między osią realizatora a płaszczyzną koła przedniego.

Inną ważną wielkością charakteryzującą ruch obiektu jest prędkość. Dla rozważanego obiektu, ze względu na wykonywane przez niego dwa ruchy, możemy dla bieżącego realizatora *r* wyróżnić dwie prędkości: prędkość ruchu postępowego (prędkość liniową) $x_r^{(5)}$ i prędkość skrętu $x_r^{(6)}$.



Rys. 3.4. Rzut prostokątny mechanizmu jazdy realizatora z określonymi współrzędnymi pierwotnymi

Wprowadzone wielkości położeń i prędkości umożliwiają jednoznaczne określenie stanu obiektu, dlatego są traktowane jako zmienne stanu. Tworzą one wektor stanu dla r-tego pojazdu $\boldsymbol{x}_r(t) = [x_r^{(1)}, x_r^{(2)}, x_r^{(3)}, x_r^{(4)}, x_r^{(5)}, x_r^{(6)}]^{\mathrm{T}}$. W celu wyznaczenia czterech pierwszych równań stanu należy określić model kinematyki, wiążący położenia z prędkościami obiektu. Odpowiednie zależności analityczne można w stosunkowo łatwy sposób uzyskać, wykorzystując proste zależności geometryczne obiektu.

Dwa ostatnie równania stanu opisują własności dynamiczne obiektu. Ich wyznaczenie, dla założenia o masie skupionej punktowo, jest również proste. Wystarczy w tym celu zróżniczkować po czasie odpowiednie energie kinetyczne obiektu. Dla bardziej złożonych układów mechanicznych lub bez wprowadzania założeń upraszczających należy zastosować jedną z kilku metod modelowania dynamiki takich układów np. [33, 80, 92], opracowanych na gruncie mechaniki analitycznej [105]. Jedną z nich jest metoda Lagrange'a-Eulera. Zastosowanie tej metody wymaga określenia tak zwanych współrzędnych (położeń) uogólnionych w liczbie równej liczbie stopni swobody układu. W rozważanym przypadku jako takie współrzędne należałoby przyjąć zdefiniowane już przesuniecie kątowe $x_r^{(4)}$ oraz przesuniecie liniowe koła przedniego, które można uzależnić od współrzędnych pierwotnych $x_r^{(1)}$, $x_r^{(2)}$, $x_r^{(3)}$. W konsekwencji, można również otrzymać dwa ostatnie równania stanu. Pomijając nieskomplikowane przekształcenia analityczne, można podać ostateczną postać równań stanu:

$$\dot{x}_r^{(1)}(t) = x_r^{(5)}(t)\cos\left(x_r^{(3)}(t) - x_r^{(4)}(t)\right), \tag{3.1a}$$

$$\dot{x}_r^{(2)}(t) = x_r^{(5)}(t)\sin\left(x_r^{(3)}(t) - x_r^{(4)}(t)\right), \tag{3.1b}$$

$$\dot{x}_r^{(3)}(t) = -\frac{1}{a_r} x_r^{(5)}(t)\sin x_r^{(4)}(t), \tag{3.1c}$$

$$\dot{x}_r^{(4)}(t) = x_r^{(6)}(t), \tag{3.1d}$$

$$\dot{x}_r^{(5)}(t) = \frac{1}{m_r\ \rho_r} u_r^{(1)}(t), \tag{3.1e}$$

$$\dot{x}_r^{(6)}(t) = \frac{1}{J_r} u_r^{(2)}(t), \tag{3.1f}$$

gdzie: a_r – odległość między osiami realizatora *r* (rys. 3.4),
m_r – masa całkowita realizatora *r*,
ρ_r – promień koła realizatora *r*,
J_r – moment bezwładności koła przedniego realizatora *r*.

Układ równań stanu (3.1) wraz ze stanem początkowym

$$\mathrm{x}_{r,0} = [x_{r,0}^{(1)}, x_{r,0}^{(2)}, x_{r,0}^{(3)}, x_{r,0}^{(4)}, x_{r,0}^{(5)}, x_{r,0}^{(6)}]^{\mathrm{T}}, \tag{3.2}$$

gdzie $x_{r,0}^{(i)} \stackrel{\Delta}{=} x_r^{(i)}(\underline{t})$, $i = 1, 2, ..., 6$, $\underline{t}$ – moment rozpoczęcia jazdy przez realizator *r*, jest modelem obiektu sterowania, czyli mechanizmu jazdy *r*-tego realizatora.

Sterowanie jazdą pojedynczego realizatora

Do wyznaczania algorytmów sterowania będą stosowane równania stanu (3.1) w wersji dyskretnej, uzyskane w wyniku zastosowania dyskretyzacji przez rozwiniecie funkcji $x_r^{(i)}(t)$ wokół punktu $t = \underline{t} + \nu h$ w szereg Taylora i uwzględnienie składników rozwinięcia do drugiego rzędu włącznie, gdzie *h* – stała długość taktu dyskretyzacji oraz ν, $\nu = 0, 1, 2, ...$ – numer taktu sterowania. Niech $x_{r,\nu}^{(i)} \stackrel{\Delta}{=} x_r^{(i)}(t = \nu h)$, $i = 1, 2, ..., 6$, $u_{r,\nu}^{(i)} \stackrel{\Delta}{=} u_r^{(i)}(t = \nu h)$, $i = 1, 2$. Wtedy równania (3.1) po dyskretyzacji można przedstawić w ogólnej formie jako układ równań różnicowych

$$x_{r,\nu+1}^{(i)} = f_r^{(i)}(\boldsymbol{x}_{r,\nu}, \boldsymbol{u}_{r,\nu}),\ i = 1, 2, ..., 6,\ \nu = 0, 1, 2, ... \tag{3.3}$$

ze stanami początkowymi $x_{r,0}^{(i)} \stackrel{\Delta}{=} x_r^{(i)}(\underline{t})$. Żmudne obliczenia analityczne, a także obszerne postaci analityczne funkcji $f_r^{(i)}$ nie będą tu przytaczane, ponieważ nie jest to konieczne do dalszych rozważań.

Bardzo istotne natomiast są postaci ograniczeń na wielkości występujące w modelu obiektu. Odnosi się to zwłaszcza do dwóch pierwszych współrzędnych stanu. Rozpocznijmy od wielkości sterujących, na które obowiązują typowe ograniczenia przedziałowe, to znaczy

$$u_{r,\nu}^{(i)} \in [\underline{u}_r^{(i)}, \overline{u}_r^{(i)}],\ i = 1, 2, \tag{3.4}$$

gdzie granice przedziałów są minimalnymi i maksymalnymi wartościami odpowiednich momentów sił, które można podać na obiekt. Na współrzędną stanu $x_r^{(3)}$ nie obowiązują żadne ograniczenia, ponieważ kąt między osią realizatora a osią odciętych układu współrzędnych może przyjmować wartości dowolne od zera, gdy oś re-

alizatora pokrywa się z osią odciętych, do 2π. Ograniczenie na kąt $x_r^{(4)}$ jest również przedziałowe, to znaczy

$$x_{r,\nu}^{(4)} \in [\underline{x}_r^{(4)}, \overline{x}_r^{(4)}], \tag{3.5}$$

gdzie $\underline{x}_r^{(4)}$ i $\overline{x}_r^{(4)}$ to odpowiednio minimalna i maksymalna wartość kąta skrętu. Przyjmujemy, że są to wartości $-\pi/2$, gdy koło jest skręcone maksymalnie w prawo patrząc w kierunku jazdy do przodu, oraz $\pi/2$, gdy koło jest skręcone maksymalnie w lewo. Podobne ograniczenia nakładamy na prędkości, czyli

$$x_{r,\nu}^{(i)} \in [\underline{x}_r^{(i)}, \overline{x}_r^{(i)}], \ i = 5,6, \tag{3.6}$$

gdzie $\underline{x}_r^{(i)}$ oraz $\overline{x}_r^{(i)}$ to odpowiednio najmniejsze oraz największe wartości prędkości dopuszczalnych.

Ograniczeń typu przedziałowego nie można nałożyć na zmienne stanu $x_r^{(1)}$ i $x_r^{(2)}$. Przyjecie takich ograniczeń oznaczałoby, że powierzchnia, po której może poruszać się realizator, ma kształt prostokąta. Byłoby to możliwe tylko wtedy, gdyby rozważać ruch jednego realizatora bez przeszkód stałych. W przeciwnym razie ograniczenia te powinny umożliwić wyznaczenie obszaru będącego zbiorem punktów na płaszczyźnie, w których może się znaleźć realizator, nie powodując przy tym kolizji z przeszkodami stałymi i ruchomymi. Obszar taki może się zmieniać w trakcie sterowania. Nazywamy go obszarem dopuszczalnym i oznaczamy jako $\boldsymbol{Y}_{r,\nu}$. Jest on różnicą mnogościową obszaru będącego przestrzenią roboczą oraz obszarów zajętych przez przeszkody ruchome i stałe, a mianowicie

$$\boldsymbol{Y}_{r,\nu} = \{(x_{r,\nu}^{(1)}, x_{r,\nu}^{(2)}) : (x_{r,\nu}^{(1)}, x_{r,\nu}^{(2)}) \in \boldsymbol{X}^{(1)} \times \boldsymbol{X}^{(2)} - (\bigcup_{\substack{s=1 \\ s \neq r}}^{R} \boldsymbol{D}_{s,\nu} \cup \bigcup_{g=1}^{G} \overline{\boldsymbol{D}}_g)\}, \tag{3.7}$$

gdzie: $\boldsymbol{X}^{(i)} \stackrel{\Delta}{=} [\underline{x}_r^{(i)}, \overline{x}_r^{(i)}]$, $i = 1,2$, a końce przedziałów wyznaczają przestrzeń roboczą dla realizatorów,

$\boldsymbol{D}_{s,\nu}$ – obszar zajęty przez realizator s w takcie ν,

$\overline{\boldsymbol{D}}_g$ – obszar zajęty przez przeszkodę stałą g,

G – liczba przeszkód stałych.

Zakładamy, że obszary $\overline{\boldsymbol{D}}_g$ są dane, natomiast zmienne w czasie obszary $\boldsymbol{D}_{s,\nu}$ należy wyznaczać w trakcie trwania procedury sterowania. Tak więc, określanie obszaru $\boldsymbol{Y}_{s,\nu}$ dla realizatora r faktycznie polega na wyznaczaniu obszarów $\boldsymbol{D}_{s,\nu}$ dla pozostałych realizatorów.

Przed podaniem sposobu wyznaczania tych obszarów zajmijmy się postacią kryterium jakości sterowania *point-to-point*. Strategia *point-to-point* polega na wyznaczaniu takich wartości wektora sterowań $\boldsymbol{u}_{r,v} = [u_{r,v}^{(1)}, u_{r,v}^{(2)}]^{\mathrm{T}}$, aby w danym takcie uzyskać maksymalne zbliżenie do zadanego położenia określonego przez wartości $x_r^{*(1)}$ i $x_r^{*(2)}$, osiągnąć odpowiednią orientację w położeniu końcowym wyrażoną przez wartość $x_r^{*(3)}$, a także uzyskać zadane (zazwyczaj zerowe) wartości prędkości końcowych $x_r^{*(5)}$ i $x_r^{*(6)}$. Wymagania te możemy zapisać w postaci wektora

$$\hat{\boldsymbol{x}}_r = [x_r^{*(1)}, x_r^{*(2)}, x_r^{*(3)}, x_r^{*(5)}, x_r^{*(6)}]^{\mathrm{T}} .$$

Jest to podwektor stanu końcowego

$$\boldsymbol{x}_r^{*} = [x_r^{*(1)}, x_r^{*(2)}, ..., x_r^{*(6)}]^{\mathrm{T}} .$$

Nie zawiera on wymagania na końcową wartość kąta skrętu. Pominiecie tego mało istotnego wymagania upraszcza analityczną postać kryterium $q_{r,v}$, które wyraża w formie ilościowej stosowaną strategię sterowania. Przyjmujemy je w następującej formie

$$q_{r,v}(\boldsymbol{u}_{r,v}) = \sum_{i\in\{1,2,5,6\}} \alpha_i [x_r^{*(i)} - f_r^{(i)}(\boldsymbol{x}_{r,v}, \boldsymbol{u}_{r,v})]^2 + \alpha_3 [\beta_{r,v} - (f_r^{(3)}(\boldsymbol{x}_{r,v}, \boldsymbol{u}_{r,v}) - f_r^{(4)}(\boldsymbol{x}_{r,v}, \boldsymbol{u}_{r,v}))]^2 , \tag{3.8}$$

gdzie: α_i, $i = 1, 2, 3, 5, 6$ – nieujemne wagi określające ważność osiągnięcia poszczególnych wymagań zawartych w wektorze $\hat{\boldsymbol{x}}_r$,

$\beta_{r,v}$ – kąt między prostą przechodzącą przez punkty $(x_{r,v}^{(1)}, x_{r,v}^{(2)})$ i $(x_r^{*(1)}, x_r^{*(2)})$ a osią odciętych układu współrzędnych.

Jeśli wymaganie na orientację realizatora w położeniu końcowym nie jest ważne, to drugi składnik w sumie, w kryterium (3.8) można pominąć. Postać kryterium wymaga podania warunku zatrzymania algorytmu. Przyjmujemy, że algorytm kończy działanie, jeśli odległość odpowiednich współrzędnych stanu od wektora $\hat{\boldsymbol{x}}_r$ jest nie większa niż zadana dokładność ε_r, czyli

$$\left(\sum_{i\in\{1,2,3,5,6\}} (x_r^{*(i)} - x_{r,v}^{(i)})^2 \right)^{\frac{1}{2}} \le \varepsilon_r . \tag{3.9}$$

Możemy teraz sformułować problem sterowania jazdą dla pojedynczego realizatora. Dla danych:

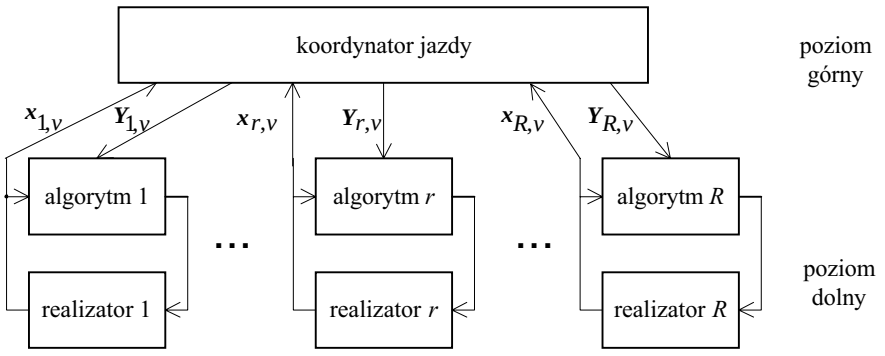
• modelu realizatora, czyli układu równań (3.3) i stanu początkowego $\boldsymbol{x}_{r,0}$,
• ograniczeń na zmienne stanu w postaci przedziałów (3.5) i (3.6),
• obszarów dopuszczalnych $\boldsymbol{Y}_{r,v}$, $v = 0,1,2,...$,
• wymagań na zakończenie jazdy, zawartych w wektorze $\hat{\boldsymbol{x}}_r$,
• wag α_i, $i = 1,2,3,5,6$ oraz dokładności ε_r, $r = 1, 2, ..., R$

należy wyznaczyć ciąg wielkości sterujących $(\boldsymbol{u}_{r,v})_{v=0,1,2,...}$ dopuszczalnych w sensie ograniczeń (3.4) tak, aby minimalizować kryterium (3.8), aż do spełnienia warunku stopu (3.9).

Obliczanie wartości $\boldsymbol{u}_{r,v}$ w kolejnych taktach jest problemem optymalizacyjnym, który ze względu na postać funkcji celu (kryterium $q_{r,v}$) oraz części ograniczeń, to znaczy obszarów $\boldsymbol{Y}_{r,v}$, należy rozwiązać stosując procedurę numeryczną.

Koordynacja jazdy

Powróćmy do pierwotnego zagadnienia, omawianego w tym podpunkcie, czyli do sterowania jazdą grupy R realizatorów. Mechanizmy jazdy wszystkich z nich tworzą obiekt sterowania. Z powodu konieczności unikania kolizji z przeszkodami stałymi i ruchomymi nie można niezależnie wyznaczać lokalnych algorytmów sterowania jazdą dla poszczególnych realizatorów. Wielkościami wiążącymi lokalne problemy sterowania są obszary dopuszczalne $\boldsymbol{Y}_{r,v}$. Są one swoistymi zmiennymi koordynacyjnymi w proponowanym systemie sterowania o strukturze hierarchicznej z koordynatorem jazdy na poziomie górnym (rys. 3.5). Inna metoda rozwiązania będzie przedstawiona w następnym rozdziale. W systemie sterowania o strukturze hierarchicznej na poziomie dolnym umieszczono lokalne algorytmy sterowania, będące wynikami rozwiązania sformułowanych wcześniej problemów sterowania

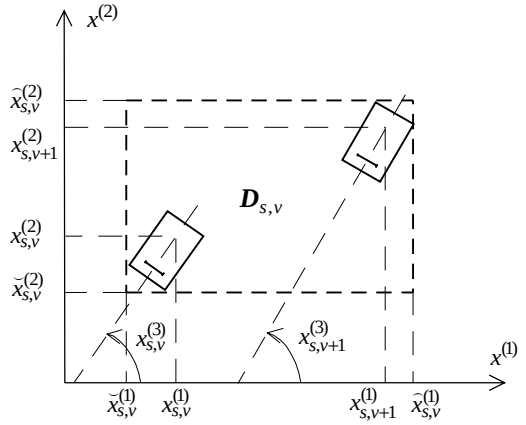


Rys. 3.5. System sterowania jazdą grupy realizatorów o strukturze hierarchicznej z koordynatorem jazdy na poziomie górnym

jazdą pojedynczych realizatorów. Umożliwiają one wyznaczenie ciągów wielkości sterujących $(\boldsymbol{u}_{r,\nu})_{\nu=0,1,2,...}$ – dla przekazywanych na bieżąco w poszczególnych taktach sterowania informacji o obszarach dopuszczalnych $\boldsymbol{Y}_{r,\nu}$. Koordynator jazdy, znajdujący się na poziomie górnym, wyznacza te obszary na podstawie informacji o bieżących stanach wszystkich realizatorów oraz informacji o wymiarach geometrycznych pojazdów. Ilustracja sposobu wyznaczania obszarów zajętości $\boldsymbol{D}_{s,\nu}$ jest przedstawiona na rys. 3.6. Polega ona na tym, że są brane pod uwagę wartości trzech pierwszych zmiennych stanu nie tylko dla bieżącego taktu ν, ale także jest uwzględniana prognoza wartości tych zmiennych w takcie następnym. Przykładem prostego algorytmu prognozy może być zależność

$$x_{r,\nu+1}^{(i)} = 2x_{r,\nu}^{(i)} - x_{r,\nu-1}^{(i)},\ \nu = 0,1,2,...,\ x_{r,-1}^{(i)} = 0,\ i = 1,2,3,, \tag{3.10}$$

uzyskana dla założenia, że dla wartości długości taktu dyskretyzacji *h* dążących do zera, różnice wartości odpowiednich zmiennych stanu dla taktów ν i $\nu+1$ oraz $\nu-1$ i ν są takie same. Na podstawie znajomości współrzędnych dwóch punktów $(x_{r,\nu}^{(1)}, x_{r,\nu}^{(2)})$ i $(x_{r,\nu+1}^{(1)}, x_{r,\nu+1}^{(2)})$, wartości kątów $x_{r,\nu}^{(3)}$ i $x_{r,\nu+1}^{(3)}$ oraz wymiarów geometrycznych pojazdu $d_r^{(1)}$ i $d_r^{(2)}$ (rys. 3.4) – w koordynatorze jazdy w prosty sposób mogą być wyznaczone współrzędne $\breve{x}_{r,\nu}^{(i)}$, $\hat{x}_{r,\nu}^{(i)}$, $i = 1,2$ (rys. 3.6) najmniejszego prostokąta o bokach równoległych do osi układu współrzędnych, zawierającego rzuty prostokątne realizatora w taktach ν i $\nu+1$. Wtedy obszary zajętości $\boldsymbol{D}_{s,\nu}$ możemy zapisać jako odpowiedni zbiór punktów przestrzeni roboczej, to znaczy



Rys. 3.6. Obszar zajętości $\boldsymbol{D}_{s,\nu}$

$$\boldsymbol{D}_{s,\nu} = \{ (x_{r,\nu}^{(1)}, x_{r,\nu}^{(2)}) \in \boldsymbol{X}^{(1)} \times \boldsymbol{X}^{(2)} : x_{r,\nu}^{(i)} \in [\breve{x}_{r,\nu}^{(i)}, \hat{x}_{r,\nu}^{(i)}],\ i = 1,2 \}. \qquad (3.11)$$

Tak więc, algorytm sterowania jazdą grupy realizatorów polega na równoległym i synchronicznym stosowaniu lokalnych algorytmów sterowania jazdą poszczególnych realizatorów, przy czym w każdym takcie sterowania z koordynatora jazdy są dostarczane postaci obszarów $\boldsymbol{Y}_{r,\nu}$. Należy zauważyć, że przy takiej koncepcji sterowania za unikanie kolizji są „odpowiedzialne" lokalne algorytmy sterowania.

3.2.2. Sformułowanie problemu i algorytmy rozwiązania w systemie dwupoziomowym

Rozważania ograniczymy do przypadku, gdy na poziomie górnym jest rozwiązywany problem szeregowania z kryterium w postaci długości uszeregowania w ujęciu deterministycznym (podpunkt 2.3.1). Podane tam sformułowanie uzupełnimy tylko wprowadzeniem oznaczeń dla tras, wzdłuż których poruszają się realizatory. Otrzymywane rozwiązanie było określone w postaci trójwymiarowej macierzy $\boldsymbol{\gamma}$ lub po dekompozycji w formie dwuwymiarowych macierzy $\tilde{\boldsymbol{\gamma}}$ i $\boldsymbol{c}$. Trasy są to ciągi stanowisk o początku i końcu w bazie. Oznaczamy je jako $\boldsymbol{M}_r = (m_r(j))_{j=1,2,...,M_r}$, gdzie $m_r(j)$ i M_r to odpowiednio j-ty element trasy i jej długość. Wartość bieżącego elementu trasy $\boldsymbol{M}_r$, tj. $m_r(j) = h$, $j = 1,2,...,M_r$ jest równa numerowi stanowiska (zadania), dla którego $\gamma_{r,\, m_r(j-1),\, m_r(j)} = 1$, gdzie $m_r(0) = m_r(M_r) = H+1$. Jeżeli realizator dojeżdża do stanowiska $m_r(j)$, to musi je również opuścić. Wówczas

$$\gamma_{r,\, m_r(j-1),\, m_r(j)} = 1 \rightarrow (\exists\, m_r(j+1) \in \overline{\boldsymbol{H}})\, (\gamma_{r,\, m_r(j),\, m_r(j+1)}) = 1. \qquad (3.12)$$

Dlatego, na podstawie macierzy $\boldsymbol{\gamma}$, trasy $\boldsymbol{M}_r$ mogą być wyznaczone w sposób rekurencyjny. Sterowanie na poziomie dolnym polega na przeprowadzaniu poszczególnych realizatorów między kolejnymi stanowiskami wzdłuż tras $\boldsymbol{M}_r$. W momencie czasu $\underline{t}_{m_r(j)}$ rozpoczyna się wykonywanie $m_r(j)$-tego elementu (odcinka) trasy $\boldsymbol{M}_r$ od stanowiska $m_r(j-1)$ do stanowiska $m_r(j)$ lub równoważnie – rozpoczyna się wykonywanie zadania $h = m_r(j)$. Oznaczając przez $\underline{t}_{m_r(j)}$ i $\bar{t}_{m_r(j)}$ odpowiednio momenty czasu, gdy zadanie $h = m_r(j)$ rozpoczyna się i kończy, można podać następujący związek między stanami początkowym i końcowym dla dwóch kolejnych elementów trasy $\boldsymbol{M}_r$

$$x_r^{*(i)} \ \stackrel{\Delta}{=}\ x_r^{(i)}(\bar{t}_{m_r(j-1)}) = x_r^{(i)}(\underline{t}_{m_r(j)}),\ \ j = 2,3,...,M_r. \qquad (3.13)$$

Ponadto jest prawdziwa równość $\underline{t}_{m_r(j+1)} = \bar{t}_{m_r(j)} + \bar{\tau}_{r,m_r(j)}$, co oznacza, że po dojeździe do bieżącego stanowiska $h = m_r(j)$ realizator może rozpocząć pokony-

wanie następnego odcinka trasy, dopiero po upływie czasu $\bar{\tau}_{r,h}$, czyli po wykonaniu czynności będącej częścią zadania $m_r(j)$.

Niech

$$\boldsymbol{a}_h = [a_h^{(1)}, a_h^{(2)}, a_h^{(3)}]^{\mathrm{T}}, \quad h = 1, 2, ..., H+1 \tag{3.14}$$

będzie wektorem zawierającym informacje o stanowisku *h*, a mianowicie niech $a_h^{(1)}$ i $a_h^{(2)}$ oraz $a_h^{(3)}$ oznaczają współrzędne środka oraz promień najmniejszego koła, będącego stanowiskiem zastępczym, zawierającego rzut prostokątny rzeczywistego stanowiska. Pojecie stanowiska zastępczego (przeszkody umownej) zostało wprowadzone po to, aby uniezależnić się od kształtu rzeczywistego stanowiska i uprościć problem wyznaczania obszarów $\boldsymbol{Y}_{r,v}$ podczas koordynacji jazdy realizatorów. Koło takie jest obszarem $\overline{\boldsymbol{D}}_h$ zajmowanym przez przeszkodę stałą, $h = 1, 2, ..., H+1$ występującą w (3.7), przy czym $G = H + 1$. Przy dojeździe do stanowiska *h* wykonujący odpowiednie zadanie realizator *r* nie powinien przekroczyć przeszkody umownej. Dlatego dla $m_r(j) = h$ wartość ε_r występująca po prawej stronie warunku stopu (3.9) powinna być zastąpiona wartością $a_h^{(3)}$. Ponadto pary punktów $(x_r^{(1)}(\underline{t}_{m_r(j)}), x_r^{(2)}(\underline{t}_{m_r(j)}))$ oraz $(x_r^{(1)}(\bar{t}_{m_r(j)}), x_r^{(2)}(\bar{t}_{m_r(j)}))$ dla $j = 1, 2, ..., M_r$, czyli dwie pierwsze składowe wektorów stanów początkowych i końcowych przyjmują wartości ze zbioru $\{(a_h^{(1)}, a_h^{(2)}), \ h = 1, 2, ..., H+1\}$, zależnie od decyzji podjętej na poziomie górnym.

Spośród różnych możliwości, za kryterium dla dwupoziomowego kompleksu operacji przyjęto długość uszeregowania, co oznacza, że jest rozwiązywany problem czasowo-optymalny w sensie długości uszeregowania. Dla poziomu dolnego nałożono warunek osiągania przez realizatory stanów końcowych z zadaną dokładnością, a nie sterowania czasowo-optymalnego. Czasy dojazdów $\hat{\tau}_{r,g,h}$ są jedynie pośrednimi wynikami sterowania. Dlatego macierz czasów $\hat{\boldsymbol{\tau}}$ zależy od $\boldsymbol{u} \stackrel{\Delta}{=} \{(\boldsymbol{u}_{r,v})_{v=0,1,2,...}, r = 1, 2, ..., R\}$. Bez precyzowania związków analitycznych, zależność tę symbolicznie możemy zapisać w formie odwzorowania $\hat{\boldsymbol{\tau}} = \hat{G}(\boldsymbol{u})$ i w konsekwencji w innej postaci możemy przedstawić kryterium jakości, to znaczy

$$Q_{\mathrm{M},u} \stackrel{\Delta}{=} Q(\boldsymbol{\gamma}; \hat{\boldsymbol{\tau}}) = Q(\boldsymbol{\gamma}; \hat{G}(\boldsymbol{u})) = Q(\boldsymbol{M}, \boldsymbol{u}), \tag{3.15}$$

gdzie $Q(\boldsymbol{\gamma}; \hat{\boldsymbol{\tau}})$ jest kryterium (2.9), a $\boldsymbol{M} \stackrel{\Delta}{=} \{\boldsymbol{M}_r, \ r = 1, 2, ..., R\}$ jest zbiorem wszystkich tras.

Ostatecznie, dla danych jak dla podproblemów z obu poziomów, uzupełnionych o informacje o stanowiskach w postaci wektorów $\boldsymbol{a}_h$ oraz dla $\varepsilon_r = a_h^{(3)}$, sterowanie dwupoziomowym kompleksem operacji polega na wyznaczeniu:

• zbioru tras przejazdów realizatorów $\boldsymbol{M}$ tak, aby minimalizować długość uszeregowania,

• zmiennych sterujących $\boldsymbol{u}$, minimalizujących dla poszczególnych realizatorów kryterium (3.8) aż do spełnienia warunku stopu (3.9), co w konsekwencji prowadzi do osiągania stanów końcowych $x_r^{(i)}(\bar{t}_{m_r(j)})$, $j = 1, 2, ..., M_r$, $r = 1, 2, ..., R$.

Podproblemy decyzyjne z obu poziomów są ze sobą powiązane. Oznacza to, że decyzje podejmowane w trakcie rozwiązywania jednego z nich mogą być traktowane jako dane dla drugiego. Tak więc trasy przejazdów realizatorów, uzyskiwane na poziomie górnym, są danymi dla podproblemu sterowania jazdą grupy realizatorów. Podobnie, czasy przejazdów z poziomu dolnego wchodzą w skład danych dla podproblemu szeregowania z poziomu górnego. Przedstawimy trzy heurystyczne algorytmy rozwiązania złożonego problemu decyzyjnego w systemie dwupoziomowym [72, 74, 77]. Pierwszy z nich, nazywany algorytmem iteracyjnym, polega na kolejnym rozwiązywaniu problemów z obu poziomów, dla ustalonego rozwiązania początkowego, aż do osiągnięcia określonego warunku stopu. Algorytm adaptacyjny, przedstawiany jako drugi, ma zupełnie inny charakter. Umożliwia on modyfikację algorytmu szeregowania na podstawie aktualnych informacji o uzyskiwanych czasach przejazdów. Czasy te mogą być różne w zależności od liczby sytuacji kolizyjnych, w których znalazły się realizatory. Znalezienie się w sytuacji kolizyjnej każdorazowo wydłuża czas przejazdu. Ostatni z algorytmów, określany jako algorytm dyspozytorski, nawiązuje do tak zwanego sterowania dyspozytorskiego i polega na przydzielaniu w każdym takcie sterowania nie wykonanych jeszcze zadań do aktualnie wolnych realizatorów.

Algorytm iteracyjny

Niech η, $\eta = 0, 1, 2, ...$ oznacza numer bieżącej iteracji. W trakcie każdej iteracji są rozwiązywane podproblemy z obu poziomów, to znaczy podproblem szeregowania w celu uzyskania zbioru tras przejazdów $\boldsymbol{M}$ oraz podproblem sterowania jazdą w celu obliczenia macierzy czasów dojazdów $\hat{\boldsymbol{\tau}}$. Obliczane są tylko te elementy macierzy $\hat{\boldsymbol{\tau}}$, które odpowiadają elementom tras ze zbioru $\boldsymbol{M}$. Pozostałe elementy macierzy pozostają bez zmian. Trasy i macierz dojazdów, uzyskane w iteracji η oznaczamy odpowiednio jako $\boldsymbol{M}(\eta)$ i $\hat{\boldsymbol{\tau}}(\eta)$. Wartości $\hat{\boldsymbol{\tau}}(\eta)$ są „mierzone na obiekcie", czyli systemie produkcyjnym lub są obliczane z modelu symulacyjnego po podaniu wielkości sterujących ze zbioru $\boldsymbol{u}(\eta)$. W dalszym ciągu będzie rozważany ten ostatni przypadek. W celu wyeliminowania powiązania podproblemów z obu poziomów, proponujemy przyjęcie punktu startowego algorytmu w postaci czasów przejazdów $\hat{\boldsymbol{\tau}}(0)$. Wartości elementów $\hat{\tau}_{r,g,h}(0)$ tej macierzy są obliczane w ten sposób, że jest

rozwiązywany podproblem sterowania jazdą realizatora r miedzy stanowiskami g i h, czyli między punktami o współrzędnych $(a_g^{(1)}, a_g^{(2)})$ i $(a_h^{(1)}, a_h^{(2)})$ bez obecności przeszkód stałych i ruchomych. Struktura systemu sterowania, w którym zastosowano algorytm iteracyjny jest przedstawiona na rys. 3.7. Warto zauważyć, że kolizje występujące w systemie sterowania, a dokładniej konieczność ich unikania, mogą być traktowane jako zakłócenia. W celu zatrzymania algorytmu rozwiązania wprowadzamy dwa warunki stopu

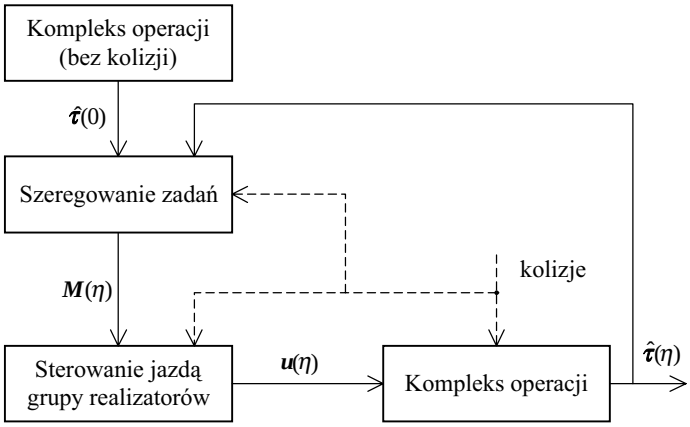
$$\sum_{k=\eta-\eta_0}^{\eta} \varsigma^{\eta-k} [Q_{\mathrm{M},u}(k-1) - Q_{\mathrm{M},u}(k)]^2 < \bar{\varepsilon}\,, \tag{3.16}$$

gdzie: $Q_{\mathrm{M},u}(k)$ – wartość kryterium jakości (3.15) w k-tej iteracji,
η_0 – liczba iteracji branych pod uwagę,
$\varsigma \in (0,1]$ – współczynnik pamięci,
$\bar{\varepsilon}$ – dokładność algorytmu

oraz

$$\hat{\boldsymbol{\tau}}(\eta) = \hat{\boldsymbol{\tau}}(\eta-1),\ \eta = 1,2,\ldots \tag{3.17}$$

Zatrzymanie algorytmu według warunku (3.16) następuje wtedy, gdy suma różnic wartości kryterium w sąsiednich iteracjach, której składniki są brane pod uwagę z coraz mniejszymi wagami w miarę oddalania się od bieżącej iteracji, jest mniejsza od zadanej dokładności $\bar{\varepsilon}$. W drugim warunku są porównywane macierze czasów dojazdu. Równość (3.17) jest rozumiana jako równość wszystkich elementów



Rys. 3.7. Dwupoziomowy system sterowania kompleksem operacji z algorytmem iteracyjnym

macierzy, czyli $\hat{\tau}_{r,g,h}(\eta) = \hat{\tau}_{r,g,h}(\eta - 1)$. Algorytm rozwiązania kończy działanie, gdy jest spełniona własność (3.17).

Bieżąca, czyli η-ta iteracja algorytmu iteracyjnego może być przedstawiona w formie następujących czterech kroków:

1. Rozwiąż problem sterowania jazdą grupy realizatorów i wyznacz $\boldsymbol{u}(\eta)$.
2. Oblicz z modelu symulacyjnego lub wyznacz z obiektu $\boldsymbol{\tau}(\eta)$.
3. Rozwiąż problem szeregowania i wyznacz $\boldsymbol{M}(\eta)$.
4. Sprawdź warunek stopu (3.16) i(lub) warunek stopu (3.17). Jeśli co najmniej jeden z nich nie jest spełniony, podstaw $\eta = \eta + 1$ i przejdź do kroku 1. W przeciwnym razie zakończ działanie algorytmu z rozwiązaniem w postaci zbioru tras $\boldsymbol{M}(\eta)$ i wartością kryterium $Q_{\mathrm{M},u}(\eta)$.

Algorytm adaptacyjny

Wprowadźmy pojęcie z d a r z e n i a. Polega ono na zakończeniu wykonania co najmniej jednego zadania. Zadanie się kończy, gdy zostanie wykonana czynność na stanowisku, czyli gdy upłynie czas $\bar{\tau}_{r,h}$ od chwili dojazdu realizatora do stanowiska h. W momencie czasu, w którym zachodzi zdarzenie, należy podjąć decyzję, czy wyznaczone poprzednio trasy nie będą zmieniane, czy też należy je zmodyfikować. Można również wówczas rozpocząć wykonywanie kolejnych zadań. Decyzje są więc podejmowane w dyskretnych momentach czasu. Inaczej można powiedzieć, że decyzje są podejmowane w taktach, a długości takich taktów, czyli przedziałów czasu, są zmienne. W dalszym ciągu n będzie oznaczać kolejne zdarzenie lub kolejny takt, w którym należy podjąć decyzję. Niech $\boldsymbol{M_r}(n)$, $r = 1, 2, ..., R$ oznacza trasę realizatora r, wyznaczoną w takcie n. W kolejnych taktach problem decyzyjny nie będzie polegał dokładnie na rozwiązywaniu sformułowanego wcześniej podproblemu szeregowania. Podczas tworzenia tras będą brane pod uwagę tylko te zadania, których wykonanie do momentu n nie zostało zakończone. Zbiór takich zadań oznaczamy przez $\boldsymbol{H}(n)$. Jest jasne, że $\boldsymbol{H}(0) = \hat{\boldsymbol{H}}$, gdzie $\hat{\boldsymbol{H}}$ jest sumą zbioru $\boldsymbol{H}$ oraz R zjazdów realizatorów do bazy. Ponadto określmy trzy inne zbiory: zbiór zadań $\boldsymbol{H}'(n)$, które w takcie n są właśnie wykonywane – zgodnie z ostatnio wyznaczonym algorytmem szeregowania, zbiór zadań $\boldsymbol{H}'''(n)$, które w takcie n się zakończyły oraz zbiór zadań $\boldsymbol{H}''(n)$, których wykonywanie do momentu n jeszcze się nie rozpoczęło. Z poziomu dolnego w każdym takcie są przekazywane informacje o zbiorach $\boldsymbol{H}'(n)$ i $\boldsymbol{H}'''(n)$ oraz czas trwania taktu $\Delta(n)$. Dla szeregowania ważny jest zbiór $\boldsymbol{H}''(n)$, który możemy wyznaczać w sposób rekurencyjny jako

$$\boldsymbol{H}''(n) = \boldsymbol{H}''(n-1) - [\boldsymbol{H}'(n) \cup \boldsymbol{H}'''(n)]. \tag{3.18}$$

Zależność (3.18) oznacza, że zbiór tych zadań, które do czasu n jeszcze się nie rozpoczęły pomniejszamy, w stosunku do analogicznego zbioru dla czasu $n-1$, o te zadania, które albo się rozpoczęły w czasie $n-1$ i nadal są wykonywane, albo się zakończyły w momencie czasu n. Zbiory zadań wykonywanych oraz nie rozpoczętych tworzą zbiór $\boldsymbol{H}(n)$, czyli

$$\boldsymbol{H}(n) = \boldsymbol{H}'(n) \cup \boldsymbol{H}''(n)$$

i są w różny sposób uwzględniane w procedurze szeregowania w takcie n. Dla zadań ze zbioru $\boldsymbol{H}'(n)$ należy obliczać nowe czasy wykonywania $\tau_{r,h}(n)$ według zależności $\tau_{r,h}(n) = \tau_{r,h}(n-1) - \Delta(n)$, czyli zmniejszać ich wartości o czas, który upłynął od poprzedniego zdarzenia. Czasy wykonywania zadań ze zbioru $\boldsymbol{H}''(n)$ nie ulegają zmianie, ponieważ wykonywanie tych zadań jeszcze się nie rozpoczęło.

Jak już wspomniano, w każdym momencie czasu (takcie) n może istnieć potrzeba wyznaczania nowych tras dla realizatorów. Warunek podjęcia procedury szeregowania będzie podany później. Aby wyznaczyć nowe trasy, należy rozwiązać problem szeregowania analogiczny do tego, który został przedstawiony w punkcie 2.2. Szeregowaniu podlegają tylko zadania ze zbioru $\boldsymbol{H}''(n)$. Początkiem nowo tworzonych tras są stanowiska, na których realizatory zakończyły wykonywanie ostatniego zadania lub do których zdążają. Oznaczmy te stanowiska przez $h'(n)$. Wtedy podproblem decyzyjny z poziomu górnego w bieżącym takcie n można sformułować następująco.

Dane są: $\boldsymbol{H}''(n)$, $\boldsymbol{R}$, $\bar{\boldsymbol{\tau}}$, $\hat{\boldsymbol{\tau}}(n)$. Należy wyznaczyć macierz $\boldsymbol{\gamma}(n)$, na którą nałożono ograniczenia

$$\gamma_{r,h,h}(n) = 0, \quad r = 1, 2, ..., R, \quad h = 1, 2, ..., H''(n), \tag{3.19}$$

$$\sum_{r=1}^{R} \sum_{g=1}^{H''(n)+1} \gamma_{r,g,h}(n) = 1, \quad h = 1, 2, ..., H''(n), \tag{3.20}$$

$$\sum_{g=1}^{H''(n)+1} \gamma_{r,g,p}(n) = \sum_{h=1}^{H''(n)+1} \gamma_{r,p,h}(n), \quad r = 1, 2, ..., R, \quad p = 1, 2, ..., H''(n)+1, \tag{3.21}$$

$$\sum_{h=1}^{H''(n)+1} \gamma_{r,h'(n),h}(n) = 1, \quad r = 1, 2, ..., R, \quad h'(n) = 1, 2, ..., H'(n), \tag{3.22}$$

$$\sum_{h=1}^{H''(n)} \gamma_{r,h,H+1}(n) = 1, \ r = 1, 2, ..., R, \tag{3.23}$$

$$\gamma \in \boldsymbol{S}, \tag{3.24}$$

gdzie:
$$\boldsymbol{S} = \left\{ \gamma(n) : \sum_{g \in \boldsymbol{H}_S} \sum_{h \in \boldsymbol{H}_S} \gamma_{r,g,h}(n) \le H_S - 1 \right\},$$
$$\boldsymbol{H}_S - \text{niepusty podzbiór } \boldsymbol{H}''(n), \ r = 1, 2, ..., R. \tag{3.25}$$

tak aby minimalizować kryterium

$$Q(\gamma(n)) = \max_{r=1,2,...,R} \left\{ \sum_{h=1}^{H''(n)+1} \sum_{h=1}^{H''(n)+1} \gamma_{r,g,h}(n) (\bar{\tau}_{r,h} + \hat{\tau}_{r,g,h}(n)) \right\}. \tag{3.26}$$

Problem ten jest analogiczny do problemu minimalizacji (2.9) z ograniczeniami (2.3)–(2.8). Najważniejszą różnicą jest ustalenie różnych początków tras oraz wykluczenie z procedury szeregowania tych zadań, które są w trakcie wykonywania. Różnice te nie są istotne, gdy do rozwiązania wykorzystamy algorytm przybliżony z pracy [47], wykorzystywany już w podpunkcie 2.4.2 dla wersji problemu po dekompozycji funkcjonalnej. Dekompozycję opisaną w podpunkcie 2.3.1 można zastosować w sposób bezpośredni.

Sformułowane zagadnienie polega faktycznie na szukaniu dróg Hamiltona dla poszczególnych realizatorów. Są one trasami przejazdów $\boldsymbol{M}_r(n)$ o początku na stanowiskach $h'(n)$ oraz końcu w bazie $h = H + 1$, czyli

$$\boldsymbol{M}_r(n) = (m_r(1,n), m_r(2,n), ..., m_r(M_r(n),n)), \tag{3.27}$$

gdzie $m_r(1,n) = h'(n)$ dla $h'(n) \in \boldsymbol{M}_r(n-1)$ oraz $m_r(M_r(n),n) = H + 1$.

Na podstawie wyznaczonych marszrut, można określić przewidywane momenty $t^h(n)$ zajścia zdarzeń polegających na zakończeniu wykonywania zadań lub zjazdach realizatorów do bazy. Liczba takich taktów $Z(n)$ jest nie większa niż $H + R$. Można je obliczyć w sposób rekurencyjny, to znaczy

$$t^h(n) = \bar{t}^{m_r(j,n)} = \underline{t}^{m_r(j-1,n)} + \tau_{r,h} = \underline{t}^{m_r(j-1,n)} + \bar{\tau}_{r,h} + \hat{\tau}_{r,m\,(j-1,n),m_r(j,n)}, \tag{3.28}$$

gdzie $m_r(j,n)$ jest elementem trasy $\boldsymbol{M}_r(n)$, który jest pokonywany przez realizator r w trakcie wykonywania zadania h. Ponadto, $m_r(0,n) = h'(n)$ oraz $m_r(0,0) = H + 1$.

Po uporządkowaniu, momenty czasu $t^h(n)$ tworzą ciąg $\boldsymbol{T}(n)$. W każdym takcie jest sprawdzany warunek

$$\boldsymbol{T}(n) = \boldsymbol{T}(n-1),\ n = 1, 2, ..., \tag{3.29}$$

to znaczy

$$t^z(n) = t^z(n-1),\ n = 1, 2, ...\ \text{dla}\ z = 1, 2, ..., Z(n). \tag{3.30}$$

Jeśli jest on spełniony, to w takcie n są realizowane trasy wyznaczone w poprzednim takcie. W przeciwnym razie dokonuje się ich adaptacji, a więc jest rozwiązywany problem szeregowania, polegający na minimalizacji (3.26) z ograniczeniami (3.19)–(3.25). Początkowa macierz czasów dojazdu $\hat{\boldsymbol{\tau}}(0)$ jest obliczana tak samo jak w algorytmie iteracyjnym.

Struktura systemu sterowania z algorytmem adaptacyjnym jest przedstawiona na rys. 3.8, a algorytm w zwartej formie może być zaprezentowany w postaci sześciu kroków.

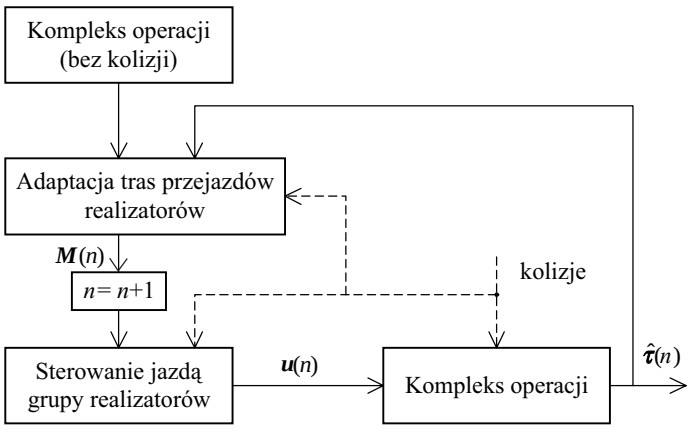
1. Ustal zbiór $\boldsymbol{H}(0) = \hat{\boldsymbol{H}}$ i $n = 0$ oraz oblicz $\hat{\boldsymbol{\tau}}(0)$.

2. Rozwiąż problem szeregowania i wyznacz $\boldsymbol{M}(n)$, a następnie utwórz ciąg $\boldsymbol{T}(n)$ i oblicz $Z(n)$.

3. Sprawdź, czy $H(n) > 0$, czyli czy zbiór $\boldsymbol{H}(n)$ jest niepusty. Jeśli tak, to przejdź do następnego kroku. W przeciwnym razie zakończ działanie algorytmu.

4. Rozwiąż problem sterowania jazdą grupy realizatorów.

5. Zaobserwuj (lub oblicz) momenty czasu $t^z(n)$, oblicz $\boldsymbol{\tau}(n)$ oraz wyznacz zbiór $\boldsymbol{H}(n)$.



Rys. 3.8. Dwupoziomowy system sterowania kompleksem operacji z algorytmem adaptacyjnym

6. Sprawdź, czy $(\forall\, h'(n) \in \boldsymbol{H}'''(n))\,(t^z(n) = t^{h'(n)}(n-1))$. Jeśli tak, to podstaw $n = n+1$ i przejdź do kroku 3. W przeciwnym razie przejdź do kroku 2.

W kroku 6. sprawdzamy, czy dla zadań, które się zakończyły w takcie *n*, planowane w poprzednim takcie momenty zakończenia $t^{h'(n)}(n-1)$ pokrywają się z faktycznymi momentami zakończenia, uzyskanymi w wyniku rozwiązania problemu sterowania w kroku 4.

Algorytm dyspozytorski

Algorytm ten jest w pewnym sensie szczególnym przypadkiem algorytmu adaptacyjnego. Polega on na przydziale nie rozpoczętych jeszcze zadań ze zbioru $\boldsymbol{H}''(n)$ do wolnych w danym takcie realizatorów ze zbioru oznaczonego jako $\boldsymbol{R}''(n)$, według arbitralnie przyjętego kryterium przydziału. Algorytm działa w obrębie jednego taktu, a więc realizuje strategię lokalnie optymalną. Oznaczmy dodatkowo przez $h_r(n)$ numer stanowiska, w którym w takcie *n* znajduje się wolny realizator ze zbioru $\boldsymbol{R}''(n)$. Algorytm można przedstawić w postaci następujących pięciu kroków.

1. Ustal $n = 0,\ \boldsymbol{H}''(0) = \boldsymbol{H},\ \boldsymbol{R}''(0) = \boldsymbol{R},\ h_r(0) = H+1,\ r = 1,2,...,R$.
2. Dla kolejnych realizatorów *r* począwszy od 1 aż do $R''(n)$:

   a) wybierz zadanie h^* o najmniejszym numerze, dla którego zachodzi nierówność

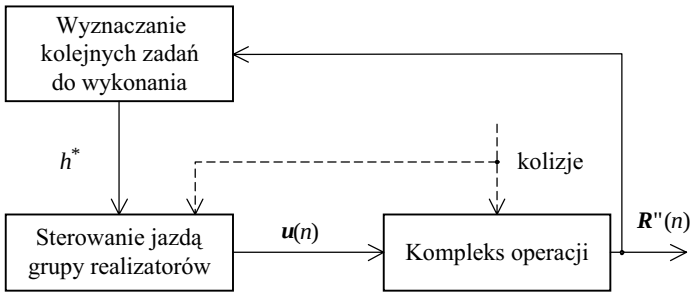
$$\bar{\tau}_{r,h^*} + \hat{\tau}_{r,h_r(n),h^*} \le \bar{\tau}_{r,h} + \hat{\tau}_{r,h_r(n),h},\ h \in \boldsymbol{H}''(n),\ h \neq h^*$$

   i przydziel go do wykonania na realizatorze *r*,

   b) podstaw $\boldsymbol{H}''(n) = \boldsymbol{H}''(n) - \{h^*\}$.

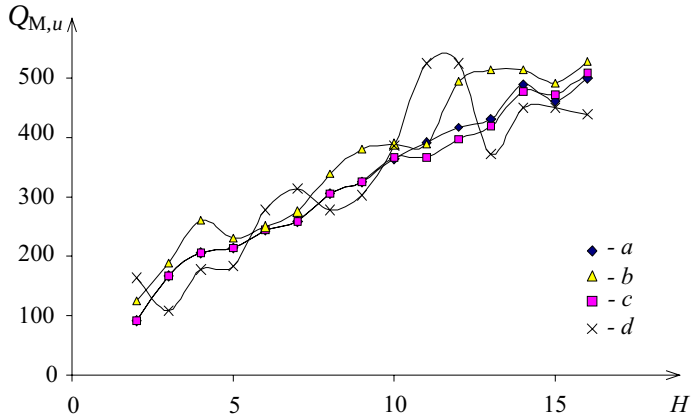
   Jeśli $n > 0$, to przejdź do kroku 4. W przeciwnym razie przejdź do kroku 3.
3. Uruchom algorytm sterowania jazdą grupy realizatorów.
4. Zaobserwuj zdarzenie *n*, wyznacz zbiór $\boldsymbol{R}''(n)$ i stanowiska $h_r(n)$ dla $r = 1,2,...,R''(n)$ oraz podstaw $n = n + 1$. Jeśli $H''(n) > 0$, czyli zbiór $\boldsymbol{H}''(n)$ jest niepusty, to przejdź do kroku 2. W przeciwnym razie przejdź do kroku 5.
5. Jeśli $R''(n) = R$, to zakończ działanie algorytmu sterowania jazdą. W przeciwnym razie wszystkim realizatorom ze zbioru $\boldsymbol{R}''(n)$ przyporządkuj zadania $h = H + 1$, czyli zjazd do bazy i przejdź do kroku 4.

Lokalna strategia wyboru realizatora z kroku 2. zapewnia przyporządkowanie kolejnym realizatorom nie wykonanych jeszcze zadań o najkrótszym czasie wykonania. Bez zmiany struktury całego algorytmu w kroku tym można stosować inne strategie, na przykład przydzielające realizatorom zadania o najdłuższym czasie wykonania. W tym przypadku struktura systemu sterowania jest taka jak na rys. 3.9.

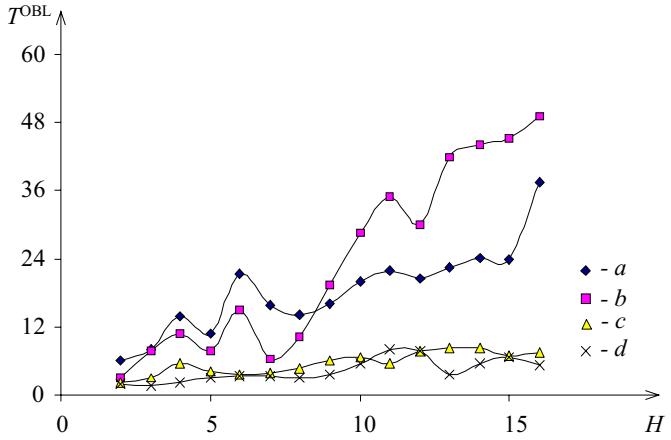


Rys. 3.9. Dwupoziomowy system sterowania kompleksem operacji z algorytmem dyspozytorskim

Na rysunkach 3.10 i 3.11 przedstawiono porównanie omówionych trzech algorytmów heurystycznych. Podstawą porównania dla różnej liczby zadań H były: kryterium jakości $Q_{\mathrm{M},u}$ oraz czas obliczeń T^{OBL}. Porównanie algorytmów ze względu na uzyskiwane wartości kryterium jakości wskazuje na brak istotnych różnic między nimi (rys. 3.10). Algorytm iteracyjny w obu wersjach jest nieco lepszy niż algorytm adaptacyjny. Wartości kryterium dla algorytmu dyspozytorskiego mają charakter bardziej nieregularny. Porównanie algorytmów ze względu na czas obliczeń pokazuje, że wyraźnie gorszy jest algorytm iteracyjny. Znacznie lepsze są algorytmy adaptacyjny i dyspozytorski. Biorąc pod uwagę oba wskaźniki porównania, wydaje się, że najlepsze własności ma algorytm adaptacyjny.



Rys. 3.10. Zależność kryterium jakości $Q_{\mathrm{M},u}$ od liczby zadań H dla stałej liczby realizatorów $R = 2$: a) algorytm iteracyjny z warunkiem stopu (3.16), b) algorytm iteracyjny z warunkiem stopu (3.17), c) algorytm adaptacyjny, d) algorytm dyspozytorski



Rys. 3.11. Zależność kryterium jakości T^{OBL} w sekundach od liczby zadań H dla stałej liczby realizatorów $R = 2$: a) algorytm iteracyjny z warunkiem stopu (3.16), b) algorytm iteracyjny z warunkiem stopu (3.17), c) algorytm adaptacyjny, d) algorytm dyspozytorski

3.3. Sterowanie kompleksami operacji z uwzględnieniem przemieszczania obiektów

Rozważmy teraz przypadek bardziej klasycznego dyskretnego systemu produkcyjnego, w którym poruszają się obiekty. Są one przemieszczane przez środki transportu miedzy nieruchomymi realizatorami, umieszczonymi na stanowiskach produkcyjnych. Rozpocznijmy od przedstawienia opisu hipotetycznego systemu produkcyjnego, tworzącego kompleks operacji, który będzie przedmiotem rozważań w niniejszym punkcie. Elementami systemu produkcyjnego są: stanowiska produkcyjne z umieszczonymi tam nieruchomymi realizatorami oraz magazynami lokalnymi; obiekty, na których realizatory wykonują zadania technologiczne; środki transportu realizujące zadania transportowe oraz centralne magazyny: wejściowy i wyjściowy dla obiektów oraz dla środków transportu. W rozważaniach nie są uwzględniane, często występujące elementy systemów produkcyjnych, takie jak narzędzia, materiały, detale, palety, zamocowania.

3.3.1. Opis systemu produkcyjnego. Model kompleksu operacji

W chwili początkowej wszystkie obiekty znajdują się w centralnym magazynie wejściowym systemu, oznaczanym jako $\underline{h}$, a wszystkie środki transportu w cen-

tralnym magazynie środków transportu $\hat{h}$. Określenie „obiekt" odnosi się do przedmiotu wykonywanych zadań, niezależnie od konkretnej postaci (stanu obiektu), zmieniającej się w trakcie wykonywanych zadań. Jak już wspomniano, obiekty są przewożone za pomocą środków transportu do stanowisk produkcyjnych, gdzie realizatory wykonują na nich zadania technologiczne – zgodnie z kolejnością określoną przez ograniczenia technologiczne. Po wykonaniu wszystkich zadań obiekty są umieszczane w centralnym magazynie wyjściowym $\overline{h}$. Zakładamy, że pojemności magazynów centralnych dla obiektów są nieograniczone, czyli mogą one jednocześnie pomieścić wszystkie obiekty. Podobnie przyjmujemy, że pojemność centralnego magazynu dla środków transportu jest nieograniczona. W systemie występuje L typów realizatorów. Realizatory jednego typu tworzą zbiór $\boldsymbol{R}_l = \{1,2, ..., h_{l,r}, ..., R_l\}$, gdzie l, $l = 1, 2, ..., L$, $h_{l,r}$, $r = 1, 2, ..., R_l$ oraz R_l, to odpowiednio indeks typu realizatora, *r*-ty realizator typu *l* oraz liczba realizatorów typu *l*. Każdy realizator $h_{r,l}$ wraz z magazynem wejściowym $\underline{h}_{r,l}$ i magazynem wyjściowym $\overline{h}_{r,l}$ tworzą stanowisko produkcyjne $h'_{l,r}$, czyli $h'_{l,r} = \langle h_{l,r}, \underline{h}_{l,r}, \overline{h}_{l,r} \rangle$. Przez $\underline{H}_{l,r}$ i $\overline{H}_{l,r}$ oznaczamy pojemności odpowiednio magazynu wejściowego $\underline{h}_{r,l}$ oraz wyjściowego $\overline{h}_{r,l}$. Aktualne liczby obiektów znajdujących się w tych magazynach zapisujemy jako $|\underline{h}_{l,r}|$ i $|\overline{h}_{l,r}|$, gdzie $|\underline{h}_{l,r}| \in \{0,1,2, ..., \underline{H}_{l,r}\}$ i $|\overline{h}_{l,r}| \in \{0,1,2, ..., \overline{H}_{l,r}\}$. Analogicznie oznaczamy liczby elementów znajdujących się w magazynach centralnych. Przyjmujmy, że w systemie występuje W identycznych środków transportu, tworzących zbiór $\boldsymbol{W} = \{1, 2, ..., w, ..., W\}$, gdzie *w* to indeks bieżącego środka transportu. Każdy środek transportu w jednej chwili może przewozić co najwyżej jeden obiekt. Przyjmujemy, że środki transportu mają mechanizmy do załadunku i rozładunku obiektów. W systemie może być produkowanych *K* typów obiektów. Przed rozpoczęciem produkcji wszystkie obiekty – na których nie wykonano jeszcze żadnego zadania technologicznego – znajdują się w magazynie $\underline{h}$. Typ obiektu oznaczamy przez *k*, gdzie $k = 1, 2, .., K$. W każdym typie jest produkowanych M_k obiektów. Tworzą one zbiór $\boldsymbol{V}_k$. Wytworzenie obiektu $v_{k,m}$ dla $m = 1, 2, ..., M_k$ wymaga wykonania N_k zadań technologicznych $z_{k,n}$, gdzie $k = 1, 2, .., K$, $n = 1, 2, ..., N_k$. Zadania te tworzą ciąg $\boldsymbol{z}_k = (z_{k,1}, z_{k,2}, ..., z_{k,n}, ..., z_{k,N_k})$, czyli kolejność ich wykonania jest ustalona i nie podlega wyborowi. W dalszym ciągu przyjmujemy, że dla każdego zadania jest określony jednoznacznie typ realizatora, na którym można to zadanie wykonać. Wyborowi podlega natomiast numer realizatora danego typu, który ma wykonać zadanie. Czas wykonania zadania nie zależy jednak od wybranego realizatora, ponieważ jest on taki sam dla każdego realizatora danego typu. Typ realizatora przewidzianego do wykonania zadania $z_{k,n}$ oznaczamy jako $l_{k,n}$. Moż-

liwa jest sytuacja, w której realizatory pewnego typu wykonują więcej niż jedno zadanie dla obiektów jednego typu. Wykonywanie zadania $z_{k,n}$ rozpoczyna się w momencie czasu $\underline{t}_{z_{k,m,n}}$, w którym następuje pobranie obiektu $v_{k,m}$ z magazynu wejściowego realizatora, który to zadanie wykonuje. Podobnie $\bar{t}_{z_{k,m,n}}$ jest momentem czasu, w którym obiekt $v_{k,m}$ znajdzie się w magazynie wyjściowym realizatora, który jest zlokalizowany na tym samym stanowisku. Wówczas $\tau_{z_{k,n}} = \bar{t}_{z_{k,m,n}} - \underline{t}_{z_{k,m,n}}$ jest czasem wykonania zadania $z_{k,n}$ na obiekcie $v_{k,m}$. Ze względu na przyjęte ograniczenia kolejnościowe

$$\underline{t}_{z_{k,m,n}} \geq \bar{t}_{z_{k,m,n-1}}, \ k = 1, 2, ..., K, \ m = 1, 2, ..., M_k \ \ n = 2, 3, ..., N_k . \qquad (3.31)$$

Przed wykonaniem każdego zadania technologicznego obiekty muszą zostać przewiezione do stanowiska, w którym znajduje się realizator, mogący wykonać to zadanie. Obiekty są przewożone przez środki transportu. Przemieszczenie obiektu miedzy magazynami lokalnymi, znajdującymi się na stanowiskach oraz magazynami centralnymi dla obiektów, nazywamy zadaniem transportowym. Tak więc w systemie produkcyjnym występują dwa rodzaje zadań: zadanie technologiczne wykonywane przez realizatory i zadania transportowe wykonywane przez środki transportu. W chwili rozpoczęcia procesu produkcyjnego wszystkie obiekty znajdują się w magazynie $\underline{h}$. Każdy z nich musi zostać przewieziony do stanowiska z realizatorem, którego typ jest odpowiedni do wykonania pierwszego zadania na transportowanym obiekcie. Następnie na obiekcie jest wykonywane przez realizator zadanie technologiczne. Zakładamy, że dla dowolnego typu obiektu żadne dwa sąsiednie zadania z ciągu $\boldsymbol{z}_k$ nie mogą być wykonane na realizatorze tego samego typu. W przeciwnym razie zadania takie można zastąpić przez jedno zadanie o odpowiednio dłuższym czasie wykonywania przez realizator tego typu. Po tym założeniu na każdym z obiektów są wykonywane na przemian zadania transportowe i technologiczne, przy czym pierwszym zadaniem jest zadanie transportowe, polegające na przewiezieniu obiektu z magazynu $\underline{h}$ do stanowiska, a ostatnim zadaniem jest zadanie transportowe, polegające na przewiezieniu obiektu, na którym wykonano już wszystkie zadania technologiczne, do magazynu $\bar{h}$. Liczba zadań transportowych jest więc o jeden większa od liczby zadań technologicznych i dla obiektu typu k wynosi $N_k + 1$. Zadanie transportowe oznaczamy jako $\zeta_{k,j}$, $j = 0,1, ..., N_k$, $k = 1, 2, .., K$, gdzie zadanie $\zeta_{k,0}$ polega na transporcie obiektu z magazynu $\underline{h}$. Oba rodzaje zadań tworzą ciąg

$$(\zeta_{k,0}, z_{k,1}, \zeta_{k,1}, z_{k,2}, \zeta_{k,2}, ..., z_{k,n}, \zeta_{k,j}, ..., z_{k,N_k}, \zeta_{k,N_k}), \qquad (3.32)$$

gdzie $n = j$.

Podobnie jak dla zadań technologicznych dla zadania $\zeta_{k,j}$ wykonywanego na obiekcie $v_{k,m}$ przez $\underline{t}_{\zeta_{k,m,j}}$, $\bar{t}_{\zeta_{k,m,j}}$ oraz $\tau_{\zeta_{k,j}} = \bar{t}_{\zeta_{k,m,j}} - \underline{t}_{\zeta_{k,m,j}}$ oznaczamy odpowiednio moment rozpoczęcia zadania, moment jego zakończenia oraz czas trwania. Mamy wówczas

$$\bar{t}_{z_{k,m,n}} \geq \bar{t}_{\zeta_{k,m,j}},\ n = j+1,\ k = 1,2,...,K,$$
$$m = 1,2,...,M_k,\ n = 1,2,...,N_k,\ j = 0,1,\ ...,N_k. \tag{3.33}$$

Wykonywanie obu rodzajów zadań odbywa się dla założenia, że zadania są niepodzielne i nie można przerywać ich wykonywania oraz każdy realizator lub środek transportu w jednym momencie czasu może wykonywać co najwyżej jedno zadanie.

Wprowadźmy jeszcze oznaczenia dla położeń nieruchomych elementów systemu produkcyjnego. Będą one istotne dla dalszych rozważań, zwłaszcza podczas określania kryterium jakości. Zbiór takich elementów, czyli stanowisk i magazynów centralnych oznaczamy jako

$$\boldsymbol{H} = \{h'_{l,r},\ l = 1,2,...,L,\ r = 1,2,...,R\} \cup \{\underline{h}\} \cup \{\bar{h}\} \cup \{\hat{h}\}.$$

Niech $\boldsymbol{a}_h = [a_h^{(1)}, a_h^{(2)}]^{\mathrm{T}}$, gdzie $h \in \boldsymbol{H}$ będą położeniami stanowisk i magazynów w kartezjańskim układzie współrzędnych. Nie rozpatrujemy wymiarów elementów zbioru $\boldsymbol{H}$ i przyjmujemy, że są one punktami materialnymi. Wtedy $d(a_{h_1}, a_{h_2})$ jest odległością miedzy dwoma elementami zbioru $\boldsymbol{H}$. Zakładamy, że $d(a_{h_1}, a_{h_1}) = 0$, $d(a_{h_1}, a_{h_2}) = d(a_{h_{21}}, a_{h_1})$, $h_1, h_2 \in \boldsymbol{H}$. W przypadku, gdy przejazd między pewnymi elementami zbioru $\boldsymbol{H}$ jest niemożliwy, wówczas odpowiednia odległość jest równa nieskończoności. Jak już wspomniano, wszystkie środki transportu są jednakowe. Oznacza to m.in., że wszystkie zadania są wykonywane ze stałą prędkością. Bez straty ogólności możemy przyjąć, że prędkość ta ma wartość jednostkową. Wówczas wartość czasu przejazdu między dwoma elementami zbioru $\boldsymbol{H}$ jest równa wartości odległości między tymi elementami.

3.3.2. Sformułowanie problemu i algorytmy rozwiązania

Operacyjne problemy decyzyjne, które są ważne dla tak opisanego dyskretnego systemu produkcyjnego, polegają faktycznie na określeniu zmiennych w czasie przyporządkowań między różnymi elementami systemu. Na przykład zagadnienie wyboru realizatorów do wykonania zadań technologicznych polega na określeniu przyporządkowania elementów zbioru realizatorów elementom zbioru zadań technologicznych. Podobnie przy wyborze środków transportu do wykonania zadań transportowych należy wyznaczyć związki między elementami zbiorów zadań transpor-

towych, środków transportu oraz magazynów jako miejsc, między którymi poruszają się środki transportu. Z kolei podczas rozważania kolejności wykonywania zadań na różnych obiektach przez ten sam realizator należy ustalić związek między tym realizatorem a podzbiorem zadań i obiektów oczekujących na obsługę. Jak już wspomniano w punkcie 3.1, kompleksowe rozpatrzenie problemów decyzyjnych dla rozważanego kompleksu z ruchomymi obiektami wymaga również rozwiązania zagadnień sterowania wykonaniem zadań, a także sterowania jazdą środków transportu przewożących obiekty. Obecnie zawęzimy zakres naszych rozważań jedynie do niektórych problemów szeregowania, czyli do wybranych operacyjnych problemów decyzyjnych.

Specyficzną cechą rozpatrywanego zagadnienia jest podział zadań na zadania technologiczne i transportowe. Rozróżnienie to jest przede wszystkim spowodowane innym sposobem wykonywania tych zadań. Zakładamy, że dla zadań technologicznych czasy ich wykonania są a priori znane i niezmienne. W przypadku zadań transportowych odpowiednie czasy nie mogą być z góry znane. Środek transportu musi bowiem najpierw dojechać do miejsca, gdzie znajduje się obiekt, który należy przewieźć, a czas tego dojazdu zależy od jego aktualnego położenia, czyli od dotychczasowej kolejności zadań technologicznych i transportowych. Ograniczenie rozważań do problemów szeregowania oznacza, że przedstawiany do rozwiązania problem decyzyjny polega na wyznaczeniu dopuszczalnego uszeregowania obu rodzajów zadań.

Uszeregowaniem dopuszczalnym Ψ zadań technologicznych i transportowych jest przyporządkowanie każdemu zdaniu technologicznemu i transportowemu dla wszystkich obiektów – przedziału czasu, w którym wykonywane jest zadanie oraz odpowiednio: dla zadań technologicznych realizatora, a dla zadań transportowych środka transportu, który wykonuje to zadanie. Spełnione muszą być przy tym następujące warunki:

A. Każde zadanie jest wykonywane tylko przez jeden realizator albo środek transportu.
B. Wykonywanie żadnego z zadań nie jest przerywane.
C. W jednej chwili każdy realizator i środek transportu wykonuje co najwyżej jedno zadanie.
D. Aż do zakończenia wykonania wszystkich zadań nie istnieje czas, w którym nie pracuje żaden z realizatorów i(albo) środków transportu, mimo że istnieją nie wykonane zadania.
E. Są spełnione ograniczenia kolejnościowe (3.31) i (3.33) oraz ograniczenia wynikające ze skończonej pojemności magazynów.

Zgodnie z przytoczonym określeniem, wyznaczenie uszeregowania można sprowadzić do podania, dla każdego zadania transportowego $\zeta_{k,j}$ wykonywanego na obiekcie $v_{k,m}$, wartości czterech następujących wielkości: numeru środka transportu oznaczanego dalej jako $w_{k,m,j} \in \boldsymbol{W}$, momentu rozpoczęcia wykonywania zadania $s_{k,m,j}$ oraz odpowiednio początkowego i końcowego położenia obiektu, czyli $\underline{a}_{k,m,j}$ i $\overline{a}_{k,m,j}$. Wielkości te wraz z zadaniem tworzą operację transportową

$$o_{k,m,j} = \left\langle \zeta_{k,m,j}, w_{k,m,j}, s_{k,m,j}, \underline{a}_{k,m,j}, \overline{a}_{k,m,j} \right\rangle . \tag{3.34}$$

Operacja transportowa jest wykonaniem zadania transportowego przez określony środek transportu. Zadanie różni się od operacji tym, że znany jest tylko typ, a nie numer realizatora (stanowiska), gdzie znajduje się obiekt i tylko typ stanowiska (a nie numer), do którego obiekt należy przewieźć. Dla zadania nie jest sprecyzowany też środek transportu, który ma przewieźć obiekt. Ponieważ środek transportu w momencie rozpoczęcia wykonywania operacji $o_{k,m,j}$ nie musi znajdować się w położeniu $\underline{a}_{k,m,j}$, w ogólnym więc przypadku wykonanie operacji składa się z dwóch faz. W pierwszej fazie środek transportu jedzie pusty do miejsca, gdzie znajduje się obiekt, a w drugiej fazie – po załadowaniu – obiekt jest przewożony przez środek transportu do zadanego położenia. Oznaczmy położenie środka transportu, który ma wykonać operację, w chwili jej rozpoczęcia przez $\widetilde{a}_{k,m,j}$. Jeśli obiekt, który ma zostać przewieziony, znajduje się gotowy do przewiezienia w magazynie wyjściowym stanowiska, to czas wykonania operacji jest równy sumie czasów przejazdu z $\widetilde{a}_{k,m,j}$ do $\underline{a}_{k,m,j}$ oraz z $\underline{a}_{k,m,j}$ do $\overline{a}_{k,m,j}$. Czas załadowania obiektu na środek transportu przyjmujemy równy zeru. Ponieważ prędkość środka transportu przyjęliśmy jako jednostkową, więc czas wykonania operacji jest równy

$$d(\widetilde{a}_{k,m,j}, \underline{a}_{k,m,j}) + d(\underline{a}_{k,m,j}, \overline{a}_{k,m,j}) .$$

Zauważmy jednak, ze pierwsza faza operacji, tj. przejazd środka transportu do miejsca, z którego ma zostać przewieziony obiekt, może się odbyć jeszcze przed umieszczeniem obiektu w magazynie wyjściowym stanowiska, to znaczy w czasie, gdy realizator wykonuje zadanie technologiczne na obiekcie. Następnie środek transportu może oczekiwać na zakończenie wykonywania zadania technologicznego przez realizator na obiekcie, bezpośrednio potem załadować obiekt z magazynu wyjściowego i rozpocząć drugą fazę operacji transportowej. W takim przypadku moment zakończenia wykonywania operacji jest równy sumie momentu zakończenia wykonywania zadania technologicznego $z_{k,n}$, gdzie $n = j$ oraz czasu przejazdu do zadanego położenia końcowego. Zatem moment zakończenia wykonywania operacji w tym przypadku jest równy $\bar{t}_{z_{k,m,n}} + d(\underline{a}_{k,m,j}, \overline{a}_{k,m,j})$, gdzie $n = j$. Pierwszą fazę

operacji transportowej możemy potraktować jako czas przygotowania środka transportu do wykonania zadania transportowego. W rozważanym systemie produkcyjnym czas ten jest zależny od kolejności wykonywania zadań. Ogólnie, moment zakończenia wykonywania operacji transportowej $o_{k,m,j}$ możemy zapisać jako

$$\bar{t}_{\zeta_{k,m,j}} = \max\{s_{k,m,j} + d(\tilde{a}_{k,m,j}, \underline{a}_{k,m,j}), \bar{t}_{z_{k,m,n}} + d(\underline{a}_{k,m,j}, \bar{a}_{k,m,j})\}, \quad (3.35)$$

gdzie $n = j$.

Drogę przebytą przez środek transportu oznaczymy przez $d_{k,m,j}$, przy czym

$$d_{k,m,j} = d(\tilde{a}_{k,m,j}, \underline{a}_{k,m,j}) + d(\underline{a}_{k,m,j}, \bar{a}_{k,m,j}). \quad (3.36)$$

Analogiczne rozważania można przeprowadzić dla uszeregowania zadań technologicznych. Wyznaczenie uszeregowania polega w tym przypadku na podaniu numeru realizatora typu $l_{k,n}$, wykonującego zadanie $z_{k,n}$, oznaczanego jako $\mu_{k,m,n}$ oraz momentu $\underline{t}_{z_{k,m,n}}$ rozpoczęcia wykonywania zadania. Moment zakończenia wykonywania zadania może być wtedy obliczony jako $\bar{t}_{z_{k,m,n}} = \underline{t}_{z_{k,m,n}} + \tau_{k,n}$. Operacje technologiczną $\bar{o}_{k,m,n}$ tworzą trzy wielkości, to znaczy

$$\bar{o}_{k,m,n} = \left\langle z_{k,n}, \mu_{k,m,n}, \underline{t}_{z_{k,m,n}} \right\rangle. \quad (3.37)$$

Zgodnie z wprowadzonymi oznaczeniami, moment zakończenia wykonywania ostatniego zadania transportowego na obiekcie $v_{k,m}$ jest równy $\bar{t}_{\zeta_{k,m,N_k}}$. Moment zakończenia wykonywania ostatniego zadania nawiązuje do długości uszeregowania i jest wówczas równy

$$T = \max_{\substack{k=1,2,\ldots,K \\ m=1,2,\ldots,M_k}} \{\bar{t}_{\zeta_{k,m,N_k}}\}. \quad (3.38)$$

Drogę przebytą przez wszystkie środki transportu w trakcie realizacji zadań oznaczamy przez *D*, gdzie

$$D = \sum_{k=1}^{K} \sum_{m=1}^{M_k} \sum_{j=0}^{N_k} d_{k,m,j}. \quad (3.39)$$

W celu dokonania oceny wyznaczanego uszeregowania, wprowadzamy dwukryterialny wskaźnik jakości, będący ważoną sumą czasu wykonania zadań oraz drogi przebytej przez środki transportu, czyli

$$Q = \lambda T + (1 - \lambda)D, \quad (3.40)$$

gdzie $\lambda \in [0, 1]$ jest współczynnikiem wagowym.

Problem sterowania rozważanym kompleksem operacji z uwzględnieniem przemieszczania obiektów można teraz sformułować następująco.

Dla danych:

A. realizatorów $h_{l,r}$ tworzących zbiory $\boldsymbol{R}_l = \{1, 2, ..., R_l\}$, $l = 1, 2, ..., L$, $r = 1, 2, ..., R_l$,
B. stanowisk $h'_{l,r}$ oraz magazynów centralnych $\underline{h}$, $\overline{h}$, $\hat{h}$ o położeniach określonych przez wektory $\boldsymbol{a}_h$, $h \in \boldsymbol{H} = \{h'_{l,r},\ l = 1, 2, ..., L,\ r = 1, 2, ..., R_l\} \cup \{\underline{h}\} \cup \{\overline{h}\} \cup \{\hat{h}\}$; przy czym stanowiska $h'_{l,r} = \langle h_{l,r}, \underline{h}_{l,r}, \overline{h}_{l,r} \rangle$ składają się z realizatorów $h_{l,r}$, lokalnych magazynów wejściowych $\underline{h}_{l,r}$ o pojemnościach $\underline{H}_{r,l}$ i lokalnych magazynów wyjściowych $\overline{h}_{l,r}$ o pojemnościach $\overline{H}_{l,r}$, a magazyny centralne maja nieograniczoną pojemność,
C. jednakowych środków transportu *w* tworzących zbiór $\boldsymbol{W} = \{1, 2, ..., W\}$, poruszających się ze stałymi prędkościami,
D. obiektów $v_{k,m}$, na których są wykonywane zadania, tworzących zbiory $\boldsymbol{V}_k = \{1, 2, ..., v_{k,m}, ..., V_{M_k}\}$, gdzie $k = 1, 2, ..., K$,
E. zadań technologicznych $z_{k,n}$, $k = 1, 2, ..., K$, $n = 1, 2, ..., N_k$, tworzących ciągi $\boldsymbol{z}_k = (z_{k,1}, z_{k,2}, ..., z_{k,N_k})$, $k = 1, 2, ..., K$, które określają ograniczenia kolejnościowe w zbiorze zadań,
F. typu realizatora $l_{k,n}$ przewidzianego do wykonania zadania $z_{k,n}$, $k = 1, 2, ..., K$, $n = 1, 2, ...N_k$,
G. czasów wykonywania zadań technologicznych $\tau_{z_{k,n}}$, $k = 1, 2, ..., K$, $n = 1, 2, ...N_k$,
H. odległości $d(h_1, h_2)$, $h_1, h_2 \in \boldsymbol{H}$ między elementami systemu,
I. wartości współczynnika wagowego λ

należy wyznaczyć

A. uszeregowanie wszystkich zadań technologicznych $z_{k,n}$, tworzących ciągi $\boldsymbol{z}_k$, dla których są spełnione warunki (3.31), czyli należy podać wartości zmiennych $\mu_{k,m,n}$ oraz $\underline{t}_{z_{k,m,n}}$, $k = 1, 2, ..., K$, $m = 1, 2, ..., M_k$, $n = 1, 2, ..., N_k$ określających operacje (3.37),
B. uszeregowanie wszystkich zadań transportowych $\zeta_{k,j}$, tworzących wraz z zadaniami technologicznymi ciągi (3.32) i dla których są spełnione warunki (3.33), czyli należy podać wartości zmiennych $w_{k,m,j}$, $s_{k,m,j}$, $\underline{a}_{k,m,j}$, $\overline{a}_{k,m,j}$, $k = 1, 2, ..., K$, $m = 1, 2, ..., M_k$, $j = 0, 1, ..., N_k$.

tak aby minimalizować wskaźnik jakości (3.40).

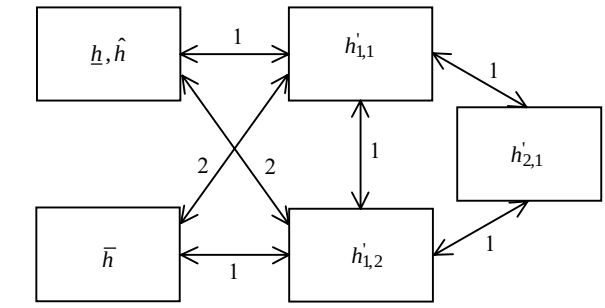
Przedstawione sformułowanie problemu odpowiada sytuacji deterministycznej, w której zakładamy aprioryczną znajomość czasów wykonywania wszystkich zadań

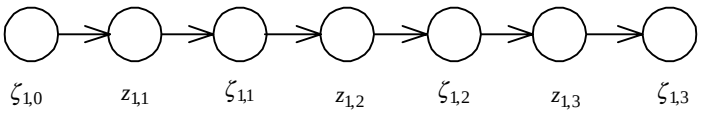
technologicznych i odległości między wszystkimi elementami systemu produkcyjnego, a także pełną gotowość do pracy i bezawaryjność wszystkich realizatorów i środków transportu. Zagadnienie takie w teorii szeregowania zadań jest określane jako ogólny problem obsługi (ang. *job-shop problem*), np. [31] i jest NP-trudnym problemem optymalizacyjnym. Warto dodać, że liczba wszystkich zadań do wykonania wynosi $\sum_{k=1}^{K} M_k(2N_k+1)$. Prezentację algorytmu rozwiązania poprzedzimy przedstawieniem przykładu ilustrującego rozwiązania dopuszczalne oraz charakter stosowanego wskaźnika jakości.

Przykład 3.1

Rozważmy następujący prosty dyskretny system produkcyjny. W systemie są realizatory dwóch typów, $L = 2$, przy czym $R_1 = 2$ i $R_2 = 1$, czyli do dyspozycji są dwa realizatory pierwszego typu i jeden realizator drugiego typu. Jest jeden środek transportu, a więc $W = 1$. Pojemność magazynów lokalnych jest nieograniczona, czyli $\underline{\underline{H}}_{1,1} = \underline{\underline{H}}_{1,2} = \underline{\underline{H}}_{2,1} = \overline{\overline{H}}_{1,1} = \overline{\overline{H}}_{1,2} = \overline{\overline{H}}_{2,1} = +\infty$. Odległości między elementami systemu są podane na rys. 3.12, przy czym położenia magazynów $\underline{h}$ i $\hat{h}$ są takie same. Do wykonania jest jeden typ obiektów, a więc $K = 1$.

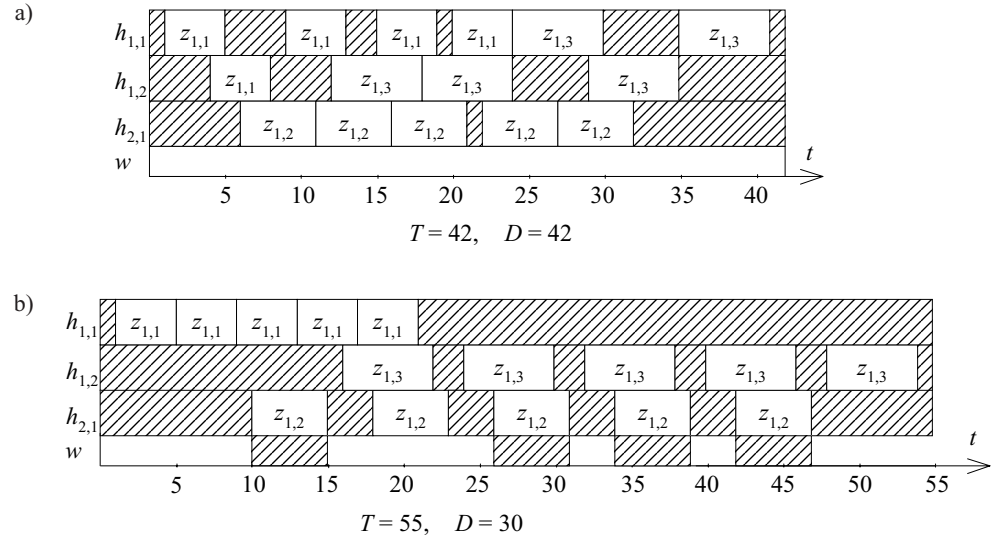
Na każdym z pięciu obiektów należy wykonać po trzy zadania, czyli $M_1 = 5$, $N_1 = 3$. Przyporządkowanie realizatorów do zadań technologicznych jest następu-





Rys. 3.12. Dyskretny system produkcyjny z przykładu 3.1: a) położenia elementów systemu z podanymi odległościami między nimi, b) sekwencja zadań technologicznych i transportowych

jące: $l_{1,1} = l_{1,3} = 1$, $l_{1,2} = 2$, a więc realizatory pierwszego typu wykonują pierwsze i trzecie zadania technologiczne, a realizator drugiego typu – drugie zadania technologiczne. Ograniczenia kolejnościowe dla obu typów zadań przedstawiono w sposób graficzny na rys. 3.12. Czasy wykonania zadań technologicznych przyjęto jako $\tau_{z_{1,1}} = 4$, $\tau_{z_{1,2}} = 5$, $\tau_{z_{1,3}} = 6$. Dwa przykładowe uszeregowania dopuszczalne dla zadań technologicznych przedstawiono na rys. 3.13. Pierwsze z nich, przedstawione na rys. 3.13.a, zapewnia minimalizację T ($\lambda = 1$). Drugie uszeregowanie, zaprezentowane na rys. 3.13.b i wyznaczone dla $\lambda = 0$, pozwala na uzyskanie minimalnej drogi przejazdu środków transportu D. Oba składniki we wskaźniku jakości (3.40) są przeciwstawne w tym sensie, że zmniejszanie jednego z nich powoduje wzrost wartości drugiego.



Rys. 3.13. Kolejność wykonywania zadań technologicznych: a) w przypadku minimalizacji T, b) w przypadku minimalizacji D

■

Algorytm rozwiązania

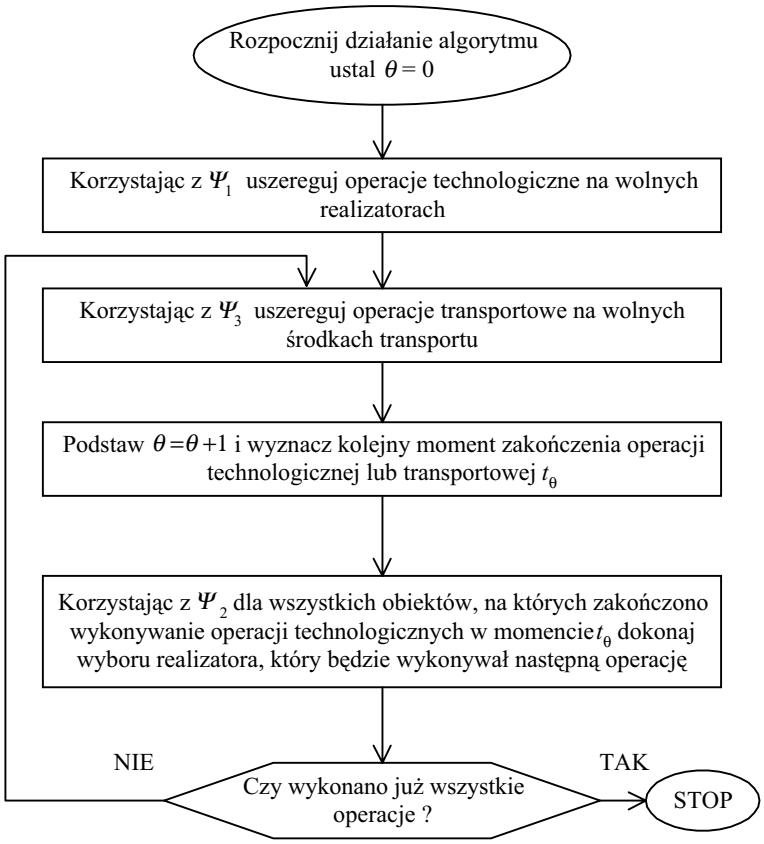
Przejdźmy do omówienia algorytmu rozwiązania sformułowanego problemu sterowania kompleksem operacji z uwzględnieniem przemieszczania obiektów. Ze względu na wykładniczą złożoność obliczeniową problemu zaproponowano algorytm heurystyczny. W dalszym ciągu dla rozróżnienia jednakowych zadań wykony-

wanych na różnych obiektach będziemy używać pojęcia operacji. Będziemy więc mówić o szeregowaniu operacji, gdy został określony już realizator lub środek transportu wykonujący zadanie albo o szeregowaniu zadań w przeciwnym przypadku. W proponowanym algorytmie decyzje są podejmowane w momencie, w którym następuje zakończenie wykonywania co najmniej jednej operacji. Decyzje te polegają na dołączeniu do istniejącej już sekwencji uszeregowanych operacji pewnej liczby nowych operacji. Następnie wyznaczany jest kolejny moment zakończenia dowolnej operacji i podejmowane są kolejne decyzje o dołączeniu nowych operacji do uszeregowania. Procedura ta jest powtarzana do chwili, w której wszystkie operacje są uszeregowane. Dla ułatwienia rozwiązania problemu, decyzje dotyczące operacji technologicznych i transportowych nie są podejmowane łącznie. Zastosowano dekompozycję na następujące prostsze algorytmy decyzyjne, realizujące różne szczegółowe funkcje algorytmu wyjściowego, oznaczanego jako Ψ:

A. Algorytm Ψ_1 wyboru operacji technologicznej, jaką wykona aktualnie wolny realizator, czyli algorytm wyboru z magazynu wejściowego stanowiska obiektu do wykonania operacji.
B. Algorytm Ψ_2 wyboru realizatora do wykonania kolejnej operacji technologicznej na obiekcie, na którym zakończono właśnie wykonywanie operacji technologicznej. Dokonywany jest wybór jednego z R_l realizatorów odpowiedniego typu. Wybór ten determinuje miejsce, do którego należy przewieźć obiekt, czyli następną operację transportową.
C. Algorytm Ψ_3 wyboru operacji transportowej, jaką wykona wolny w danej chwili środek transportu, czyli algorytm p r z e m i e s z c z a n i a o b i e k t ó w.

Dokonana dekompozycja polega na oddzielnym wyznaczaniu uszeregowań dla operacji technologicznych i transportowych. Dekompozycja taka prowadzi do pogorszenia jakości rozwiązania, ponieważ obydwa problemy szeregowania są od siebie zależne, to znaczy określone uszeregowanie operacji technologicznych wpływa na możliwości uszeregowania operacji transportowych i odwrotnie.

Ogólny schemat działania algorytmu wyjściowego Ψ jest przedstawiony na rys. 3.14. Formalny zapis tego algorytmu, który jest zawarty w pracy [96], będzie tu pominięty. Szczegółowo omówimy jedynie algorytm Ψ_3, który umożliwia rozważane w tym punkcie przemieszczanie obiektów. Zakładamy, że algorytmy Ψ_1 i Ψ_2 są ustalone. Będziemy poszukiwać takiego algorytmu Ψ_3, który zastosowany w przedstawionym na rys. 3.14 algorytmie, prowadzi do wyznaczenia uszeregowań operacji transportowych z uwzględnieniem wymagań zawartych we wskaźniku (3.40). Przyjmujemy następujące postaci algorytmów Ψ_1 i Ψ_2.



Rys. 3.14. Ogólny schemat działania algorytmu rozwiązania problemu sterowania kompleksem operacji z uwzględnieniem przemieszczania obiektów

Algorytm Ψ_1

Działanie algorytmu polega na przydzieleniu każdemu wolnemu w chwili t_θ, $\theta = 0,1,...$ realizatorowi $h_{l,r}$ typu $l_{k,n}$ operacji technologicznej, polegającej na wykonaniu zadania $z_{k,n}$ na jednym z obiektów typu *k* oczekujących w magazynie wejściowym $\underline{h}_{l,r}$, jeśli $\left|\underline{h}_{l,r}\right| \geq 1$. Wynikiem działania algorytmu jest wartość $\mu^*_{k,m,n}$. Moment $\underline{t}_{z_{k,m,n}}$ rozpoczęcia wykonywania zadania jest równy t_θ. Jest to równoznaczne szeregowaniu operacji $\bar{o}^*_{k,m,n} = \left\langle z_{k,n}, \mu^*_{k,m,n}, t_\theta \right\rangle$ na realizatorze $h_{l,r}$. Sposoby określania wartości $\mu^*_{k,m,n}$ mogą być różne. Podamy trzy heurystyczne reguły wyboru operacji (obiektu).

A. Wybieramy operację, która najdłużej oczekuje w magazynie $\underline{h}_{l,r}$ – reguła $\Psi_{1,1}$.
B. Wybieramy operację, której czas wykonania jest najdłuższy – reguła $\Psi_{1,2}$.
C. Wybieramy operacje, dla której numer typu obiektu *k* jest najmniejszy, a w przypadku równych numerów typów – obiekt o najmniejszym numerze – reguła $\Psi_{1,3}$.

Algorytm Ψ_2

Algorytm ten umożliwia wybór jednego realizatora *r* typu $l_{k,n+1}$ ze zbioru $\boldsymbol{R}_{l_{k,n+1}}$, który wykona zadanie $z_{k,n+1}$ na obiekcie $v_{k,m}$, gdzie w momencie t_θ inny realizator odpowiedniego typu – oznaczany jako $h^{\mathrm{I}}_{l,r}$ – zakończył wykonywanie zadania technologicznego $z_{k,n}$. Można go przedstawić w dwóch krokach.

1. Wyznacz podzbiór $\hat{\boldsymbol{R}} \subset \boldsymbol{R}_{l_{k,n+1}}$ takich realizatorów typu $l_{k,n+1}$, dla których wartość $\sum_{i=1}^{\left|h_{l_{k,n+1},r}\right|} \hat{\tau}_{z_{k,n}(i)}$ jest najmniejsza, gdzie $z_{k,n}(i)$ oznacza zadanie do wykonania na *i*-tym obiekcie odpowiedniego typu, znajdującym się w magazynie $\underline{h}_{l,r}$.
2. W przypadku, gdy zbiór $\hat{\boldsymbol{R}}$ jest jednoelementowy, wówczas zawarty w nim numer wskazuje na szukany realizator *r*. W przeciwnym razie, dla wszystkich elementów $h_{l_{k,n+1},r}$ zbioru $\hat{\boldsymbol{R}}$ obliczamy odległości $d(h_{l_{k,n+1},r}, h^{\mathrm{I}}_{l,r})$ i jako szukany realizator *r* przyjmujemy ten, dla którego odległość ta jest najmniejsza.

Algorytm Ψ_3

Podejmowanie decyzji dotyczących szeregowania operacji transportowych następuje w momencie początkowym działania systemu i następnie po zakończeniu wykonywania poszczególnych zadań transportowych. W momentach tych, oznaczanych przez t_θ, dla wszystkich środków transportu gotowych do wykonania zadań, tworzących zbiór $\overline{\boldsymbol{W}}_\theta$, dokonuje się wyboru zadania transportowego $\zeta^*_{k,j}$ ze zbioru zadań możliwych do wykonania w momencie t_θ, czyli ze zbioru $\overline{\boldsymbol{Z}}_\theta$. Oznaczmy dodatkowo przez $\zeta_{k,j}(\theta)$ i $w(\theta)$ bieżące elementy zbiorów $\overline{\boldsymbol{Z}}_\theta$ i $\overline{\boldsymbol{W}}_\theta$, a także przez $\underline{a}_{k,m,j}(\theta)$ i $\overline{a}_{k,m,j}(\theta)$ oraz $\widetilde{a}_{k,m,j}(\theta)$ odpowiednio położenia początkowe i końcowe obiektów do przewiezienia oraz położenie środka transportu $w(\theta)$. Przy wyznaczaniu algorytmu Ψ_3 kierujemy się wymaganiami określonymi we wskaźniku jakości (3.40). Dla różnych czasów wykonywania zadań problem ten jest NP-trudnym zagadnieniem optymalizacyjnym [83, 107]. Skoncentrujemy się więc na przedstawieniu algorytmu heurystycznego. Omówimy wpierw przesłanki stanowiące podstawę do opracowania algorytmu, wynikające z analizy problemu i własności uszeregowań. Ponieważ w skład wskaźnika jakości (3.40) wchodzą dwa kryteria: moment

T zakończenia wykonywania ostatniego zadania oraz droga D przebyta przez wszystkie środki transportu, więc dla różnych wartości λ poszczególne kryteria mają różne znaczenie. Dla małych λ decydująca w ocenie uszeregowania będzie długość drogi przejechanej przez środki transportu, natomiast dla dużych λ decydujący będzie moment T. Algorytm Ψ_3 powinien zatem uwzględniać różne wartości Ψ_3 i odpowiednio dawać w rezultacie uszeregowania o małych wartościach T lub D albo o wartościach pośrednich. Wynika stąd sugestia opracowania algorytmu z pewną liczbą parametrów, które można byłoby tak zmieniać, aby wpływać na wartości T i D. Można przypuszczać, że algorytm minimalizujący D powinien wybierać zadanie, którego wykonanie spowoduje przebycie przez środek transportu najkrótszej drogi. W ten sposób nie będą wybierane zadania transportu obiektów oddalonych, ponieważ środek transportu musiałby wpierw dotrzeć do takiego obiektu pokonując długą drogę. Zadanie takie powinno być wykonane przez środek transportu znajdujący się bliżej. Z drugiej strony, można przypuszczać, że algorytm minimalizujący T powinien tak szeregować zadania, aby jak najefektywniej wykorzystywać wszystkie realizatory. Regułą wyboru, która wydaje się do tego prowadzić, jest wybór zadania transportowego o najdłuższym czasie oczekiwania na wykonanie, gdyż żaden z realizatorów nie powinien wtedy czekać zbyt długo na obiekt do wykonania zadania. W pracy [96] przeprowadzono szczegółową analizę różnych dyskretnych systemów produkcyjnych w celu określenia czynników powodujących zmniejszenie wartości T. Zauważono, że wcześniejsze wykonywanie końcowych zadań technologicznych na pewnych obiektach w stosunku do początkowych zadań na innych obiektach, sprzyja zmniejszeniu T. Wynika stąd wniosek, że dla zmniejszania T wcześniej należy wybierać zadania transportowe dla obiektów o dużej liczbie wykonanych zadań. Zaobserwowano również, że zmniejszenie drogi D następuje w sytuacji, gdy w systemie kolejno są wykonywane zadania technologiczne dla tego samego typu obiektów przed wykonywaniem zadań dla obiektów innego typu. Wówczas trasy przejazdów środków transportu są mniej różnorodne, co powoduje zmniejszanie długości drogi przebytej bez przewożenia obiektów.

Na podstawie przytoczonych przesłanek opracowano następujący heurystyczny algorytm wyboru operacji transportowych (algorytm przemieszczania obiektów). Ma on charakter lokalny, to znaczy decyzje są podejmowane w kolejnych momentach czasu t_θ, w których zaszły zdarzenia, polegające na zakończeniu wykonywania co najmniej jednego zadania technologicznego i należy przydzielić środek transportu do przewiezienia obiektu, na którym wykonano to zadanie. Algorytm podajemy dla bieżącej chwili t_θ i dla dowolnego środka transportu $w(\theta) \in \overline{W}_\theta$. Jeżeli zbiór $\overline{Z}(\theta)$ nie jest pusty, to należy wykonać następujące kroki.

1. Dla wszystkich zadań $\zeta_{k,j}(\theta) \in \overline{\boldsymbol{Z}}(\theta)$ oblicz wartość wskaźnika jakości

$$q_{k,m,j}(\zeta_{k,j}(\theta)) = \alpha_1(t_\theta - \bar{t}_{k,m,j}) + \alpha_2[d(\tilde{a}_{k,m,j}(\theta), \underline{a}_{k,m,j}(\theta)) + d(\underline{a}_{k,m,j}(\theta), \overline{a}_{k,m,j}(\theta))] + \alpha_3(N_k - j) + \alpha_4 k, \quad \text{dla} \quad n = j \tag{3.41}$$

gdzie $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ – nieujemne współczynniki liczbowe będące parametrami algorytmu.

2. Wybierz do wykonania przez środek transportu $w(\theta)$ zadanie $\zeta^*_{k,j}(\theta)$, dla którego jest spełniona zależność

$$q_{k,m,j}(\zeta^*_{k,j}(\theta)) < q_{k,m,j}(\zeta_{k,j}(\theta)),$$
$$\text{dla} \quad \zeta^*_{k,j}(\theta), \zeta_{k,j}(\theta) \in \overline{\boldsymbol{Z}}_\theta, \zeta^*_{k,j}(\theta) \neq \zeta_{k,j}(\theta). \tag{3.42}$$

W przypadku większej liczby takich zadań wybierz dowolne z nich. Następnie podstaw $w_{k,m,j} = w(\theta)$ oraz $s_{k,m,j} = t_\theta$. Wielkości te wraz z zadaniem $\zeta^*_k(\theta)$ oraz położeniami $\underline{a}_{k,m,j}(\theta)$ i $\overline{a}_{k,m,j}(\theta)$ określają operację transportową $o^*_{k,m,j}$.

W algorytmie występują cztery parametry, przyjmujące wartości w zbiorze liczb rzeczywistych. Parametr α_1 wpływa na wartość składnika związanego z różnicą momentu podejmowania decyzji i momentu zakończenia wykonywania ostatniego zadania technologicznego na danym obiekcie. Wyraża on czas oczekiwania danego zadania na wykonanie. Kolejny parametr jest związany z drogą, jaką musi przebyć środek transportu, aby wykonać dane zadanie. Następny parametr, czyli α_3, ma wpływ na składnik związany z liczbą dotychczas wykonanych zadań na obiekcie. Wartość składnika jest odwrotnie proporcjonalna do liczby wykonanych zadań. Dlatego są preferowane zadania transportowe na obiektach, na których wykonano najwięcej zadań. Parametr α_4 jest związany ze składnikiem określającym typ obiektu. W tym przypadku są preferowane zadania transportu obiektów o najmniejszych numerach typów. Tak więc, podczas wyboru zadania transportowego są uwzględniane wszystkie przesłanki, które zostały wcześniej opisane.

Odpowiedni dobór parametrów umożliwia kształtowanie algorytmu rozwiązania i uzyskiwanie określonych uszeregowań. Przykładowo, jeśli założymy, że $\alpha_1 = \alpha_2 = 0$, $\alpha_3 = -1$, $\alpha_4 > \max_{k=1,2,...,K}\{N_k\}$, to

$$q_{k,m,j}(\zeta_{k,j}) < q_{k+1,m,j}(\zeta_{k,j}), \quad k = 1,2,...,K-1, \quad m = 1,2,...,M_k, \quad j = 0,1,...,N_k$$

oraz

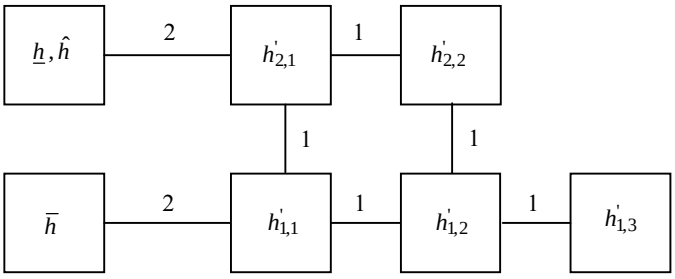
$$q_{k,m,j}(\zeta_{k,j}) < q_{k,m,j+1}(\zeta_{k,j}), k = 1,2,...,K, \quad m = 1,2,...,M_k, \quad j = 0,1,...,N_k - 1.$$

Stosując opisany algorytm, uzyskujemy uszeregowanie, w którym środki transportu przewożą kolejno obiekty poszczególnych typów, a w ramach jednego typu wszystkie obiekty, na których ma być wykonane pierwsze zadanie technologiczne, następnie wszystkie obiekty w celu wykonania drugiego zadania technologicznego itd.

Z prowadzonych badań symulacyjnych [96, 97] wynika różny wpływ parametrów algorytmu na wartość wskaźnika jakości Q. Zilustrujmy to na przykładzie.

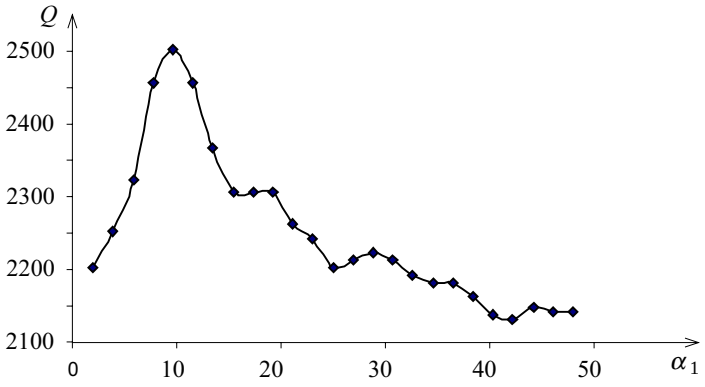
Przykład 3.2

Rozważmy prosty system produkcyjny jak na rys. 3.15, gdzie $L = 2$, $R_1 = 3$, $R_2 = 2$, $W = 3$, $K = 2$, $M_1 = M_2 = 50$, $N_1 = 5$, $N_2 = 3$, $\tau_{z_{1,1}} = \tau_{z_{1,3}} = \tau_{z_{1,5}} = \tau_{z_{2,1}} = \tau_{z_{2,3}} = 1$, $\tau_{z_{1,2}} = \tau_{z_{1,4}} = \tau_{z_{2,2}} = 2$.



Rys. 3.15. Położenia elementów dyskretnego systemu produkcyjnego z przykładu 3.2 z podanymi odległościami między nimi

Komputerowe badania symulacyjne wykazały, że największy i przeciwstawny wpływ na wartość wskaźnika jakości Q, będącego ważoną sumą dwóch kryteriów T i D, mają parametry α_1 i α_2. Znaczenie parametrów α_3 i α_4 okazało się znacznie mniejsze i nie będzie tu prezentowane. Dla małych wartości α_1 momenty T, uzyskane w wyniku działania algorytmu, są duże, a ze wzrostem α_1 szybko maleją. Jednak w pewnym momencie dalszy wzrost α_1 nie powoduje zmniejszania T. Inaczej α_1 wpływa na długość drogi D przejechanej przez środki transportu. Dla małych, w stosunku do pozostałych parametrów, wartości α_1 droga D jest niewielka, by szybko wzrosnąć dla α_1 zbliżonych do wartości pozostałych parametrów. Dalsze zwiększanie α_1 nie wpływa w sposób istotny na D, która utrzymuje się w przybliżeniu na jednakowym poziomie, przy czym ze zwiększaniem α_1 występują duże wahania D wokół wartości tego poziomu. Dla kryterium Q najkorzystniejsze są duże

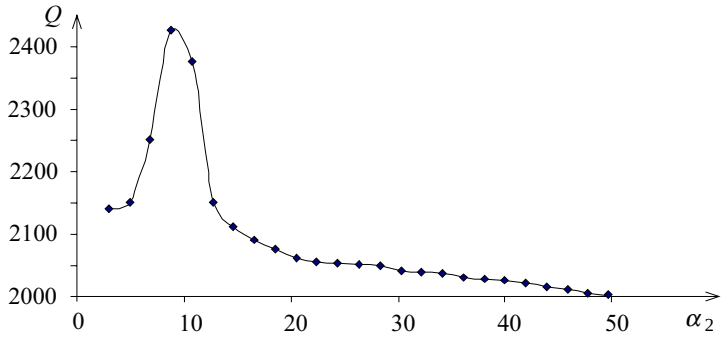


Rys. 3.16. Zależność Q od α_1 w przykładzie 3.2

wartości α_1 dla dużych λ. Natomiast dla małych λ korzystne jest ustalenie małych, w stosunku do innych parametrów, wartości α_1. Ich wartości zbliżone do wartości innych parametrów nie są korzystne. Na rys. 3.16 przedstawiono zależność Q od α_1 dla $\lambda = 0{,}5$ oraz ustalonych wartości pozostałych parametrów. Wpływ wartości parametru α_2, związanego z długością drogi przebytej przez środek transportu podczas wykonywania danej operacji, na wartości kryteriów T i D, jest odwrotny niż dla α_1, co jest zgodne z intuicyjnymi przypuszczeniami. Dla małych, w stosunku do innych parametrów, wartości α_2 wartość momentu T jest stosunkowo mała. Gdy zwiększa się α_2 do wartości innych parametrów, osiąga ona wyraźne maksimum (dynamika wzrostu T jest duża). Dalsze zwiększanie α_2 nie powoduje już istotnej zmiany T. Wartość drogi D bardzo maleje ze zwiększaniem się α_2 i ustala się dopiero dla wartości α_2 wielokrotnie większych od pozostałych parametrów. Dla kryterium Q najlepsze są małe wartości α_2 dla dużych λ oraz bardzo małe w porównaniu z innymi parametrami – dla małych λ. Zależność Q od α_2 dla ustalonych wartości innych parametrów oraz dla $\lambda = 0{,}5$ przedstawiono na rys. 3.17.

■

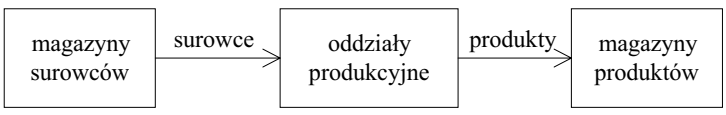
Ogólnym wnioskiem z przykładu 3.2 jest spostrzeżenie, że zmiany wartości parametrów – dotyczy to zwłaszcza α_1 i α_2 – umożliwiają uzyskanie uszeregowań o różnych wartościach Q. Cecha ta umożliwia poszukiwanie coraz lepszych rozwiązań w procesie minimalizacji Q względem α_1, α_2, α_3 i α_4. Procedura taka została opracowana i jest przedstawiona w pracy [96]. Jest w niej wykorzystywana jedna z popularnych bezgradientowych numerycznych metod optymalizacji.



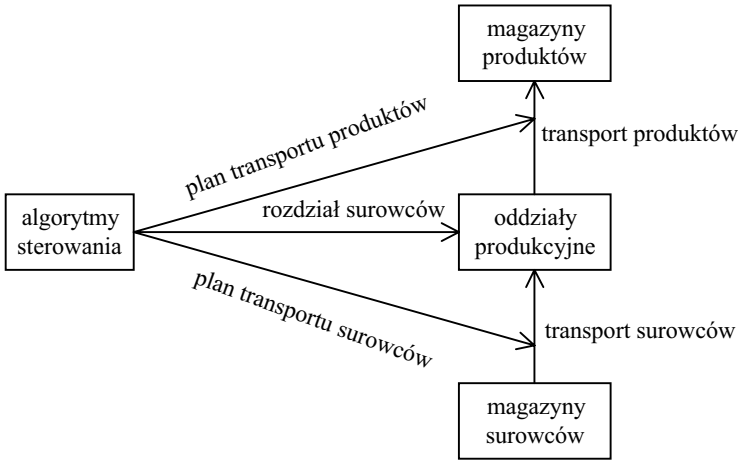
Rys. 3.17. Zależność Q od α_2 w przykładzie 3.2

3.4. Rozdział zadań w kompleksie operacji z uwzględnieniem transportu surowców i produktów

Podobnie jak w poprzednich dwóch punktach niniejszego rozdziału, tematyka prezentowana w bieżącym punkcie może być traktowana jako uogólnienie klasycznego problemu decyzyjnego dla kompleksu operacji. Obecnie nawiązujemy do zagadnienia czasowo-optymalnego rozdziału zadań w kompleksie operacji równoległych, wprowadzonego w podpunkcie 1.2.2. Rozważmy system produkcyjny (przedsiębiorstwo produkcyjne), w którym okresowo należy wyznaczać wielkość produkcji z uwzględnieniem kosztów transportu oraz podejmować decyzje dotyczące transportu z uwzględnieniem wielkości produkcji. Składa się on z trzech powiązanych ze sobą części: magazynów surowców (dostawców surowców), pracujących równolegle oddziałów produkcyjnych oraz magazynów produktów (odbiorców produktów). Ogólna struktura systemu produkcyjnego jest przedstawiona na rys. 3.18. W konsekwencji system sterowania składa się z trzech podsystemów: sterowania transportem surowców od dostawców do oddziałów produkcyjnych, sterowania produkcją w oddziałach produkcyjnych i sterowania transportem produktów z oddziałów produkcyjnych do odbiorców (rys. 3.19). W pierwszym podsystemie są wyzna-



Rys. 3.18. Ogólna struktura systemu produkcyjnego



Rys. 3.19. Schemat systemu sterowania

czane ilości surowców transportowanych między poszczególnymi magazynami surowców a oddziałami produkcyjnymi, czyli jest określany plan transportu surowców. Podobnie w trzecim podsystemie jest wyznaczany plan transportu produktów, czyli ilości produktów przewożonych między poszczególnymi oddziałami produkcyjnymi a magazynami produktów. Zadaniem drugiego podsystemu sterowania jest rozdział dostarczonych surowców między oddziały produkcyjne. Ponieważ założyliśmy niezależne działanie oddziałów produkcyjnych, problem dla tego podsystemu może być sprowadzony do rozdziału zadań, traktowanych jako surowce, w kompleksie operacji niezależnych. Decyzje oraz ich wyniki dla trzech podsystemów sterowania są zależne, ponieważ rozdział surowców ma bezpośredni wpływ na dane dla planowania transportu surowców oraz wpływ pośredni na dane dla planowania transportu produktów.

3.4.1. Model kompleksu operacji i sformułowanie problemów sterowania

Rozpocznijmy od niezależnego rozpatrzenia poszczególnych podsystemów sterowania.

Rozdział surowców

Korzystamy z oznaczeń używanych w podpunkcie 1.2.2. Tak więc wielkość V jest całkowitą ilością surowców przetwarzanych przez R równolegle działających

oddziałów produkcyjnych (operacji), v_r, $r = 1, 2, ..., R$ – ilością surowców przydzieloną *r*-temu oddziałowi produkcyjnemu (*r*-tej operacji), a T_r – czasem pracy *r*-tego oddziału. Zmienne v_r tworzą wektor $\boldsymbol{v} = [v_1, v_2, ..., v_R]^T$. Zgodnie z zależnością (1.44) przyjmujemy naturalne założenia, że każdej operacji można przydzielić nieujemną ilość surowców oraz że cały surowiec będzie rozdzielony, czyli $v_r \geq 0$, $r = 1, 2, ..., R$ oraz $\sum_{r=1}^{R} v_r = V$.

Modele poszczególnych operacji, czyli zależności między czasem pracy oddziału a wielkością przydzielonych surowców przyjmujemy w ogólnej postaci $T_r = \bar{\gamma}_r(v_r)$, $r = 1, 2, ..., R$. Zakładamy, że wszystkie oddziały produkcyjne rozpoczynają pracę w tym samym momencie czasu. Wówczas ich czas pracy, zgodnie z (1.45a) może być wyrażony jako

$$T = \max_{r=1, 2, ..., R} \{\bar{\gamma}_r(v_r)\}. \tag{3.43}$$

Dla danej ilości surowców problem polega na ich rozdziale między oddziały produkcyjne, czyli na wyznaczeniu dopuszczalnego wektora $\boldsymbol{v}^*$ tak, aby minimalizować (3.43).

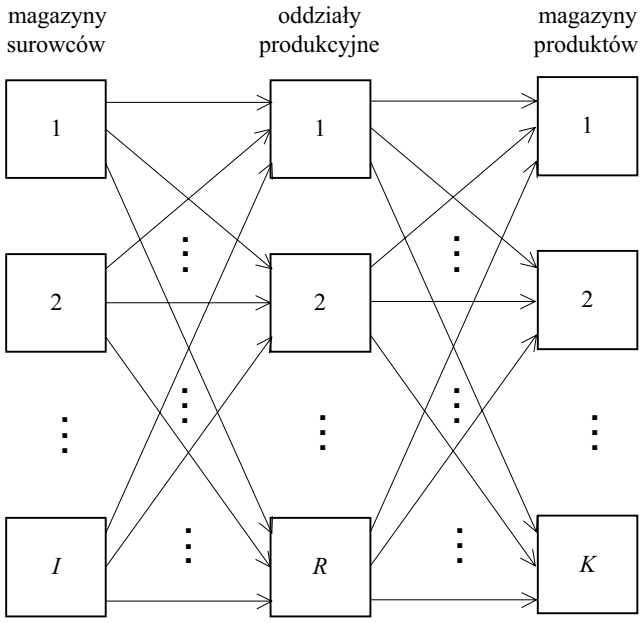
Przyjmujemy, że koszt produkcji $J_2(\boldsymbol{v})$ wszystkich oddziałów jest związany z czasem produkcji *T* prostą zależnością

$$J_2(\boldsymbol{v}) = \pi T, \tag{3.44}$$

gdzie π jest danym nieujemnym współczynnikiem proporcjonalności.

Transport surowców i produktów

Niech *I* będzie liczbą magazynów surowców a *i* indeksem bieżącego magazynu. Analogicznie przez *K* oraz *k* oznaczamy liczbę magazynów produktów oraz indeks bieżącego magazynu produktów. Szczegółowa struktura trzech części systemu produkcyjnego z zaznaczonymi połączeniami transportowymi jest przedstawiona na rys. 3.20. Przyjmujemy, że surowce mogą być transportowane z dowolnego magazynu *i* do dowolnego oddziału produkcyjnego *r*, a produkty – z każdego oddziału *r* do każdego magazynu produktów *k*. Jednostkowe koszty transportu surowców oraz produktów oznaczamy odpowiednio przez $c'_{i,r}$, $i = 1,2, ..., I$, $r = 1, 2, ..., R$ oraz $\bar{c}_{r,k}$, $r = 1, 2, ..., R$, $k = 1, 2, ..., K$, gdzie $c'_{i,r}$ jest jednostkowym kosztem transportu surowców z magazynu surowców *i* do *r*-tego oddziału produkcyjnego, natomiast $\bar{c}_{r,k}$ to jednostkowy koszt transportu produktów z *r*-tego oddziału do *k*-tego magazynu produktów. Ilości surowców $x'_{i,r}$ przewożone z *i*-tego magazynu do *r*-tego oddziału



Rys. 3.20. Struktura systemu produkcyjnego z połączeniami transportowymi

tworzą macierz decyzyjną $\boldsymbol{x}' = [x'_{i,r}]_{\substack{i=1,2,...,I \\ r=1,2,...,R}}$. Również w przypadku transportu produktów określamy macierz decyzyjną $\bar{\boldsymbol{x}} = [\bar{x}_{r,k}]_{\substack{r=1,2,...,R \\ k=1,2,...,K}}$, gdzie $\bar{x}_{r,k}$ jest ilością produktów przewożonych z oddziału *r* do magazynu *k*. Wszystkie magazyny mają skończone pojemności. Niech teraz w_i, $i = 1, 2, ..., I$ oznaczają pojemności magazynów surowców, a $\bar{v}_k$, $k = 1, 2, ..., K$ – pojemności magazynów produktów. W przypadku, gdy magazynom nadamy interpretację dostawców i odbiorców, wtedy mówimy nie o pojemnościach, lecz o podaży surowców w_i i o popycie (zapotrzebowaniu) na produkty $\bar{v}_k$. Problemy dla podsystemów są powiązane w ten sposób, że cała ilość surowców znajdujących się w magazynach ma być dostarczona do oddziałów, to znaczy

$$\sum_{i=1}^{I} w_i = \sum_{r=1}^{R} v_r = V, \tag{3.45}$$

a z każdego magazynu surowców należy wywieźć tyle surowców, ile się tam znajduje, czyli

$$\sum_{r=1}^{R} x'_{i,r} = w_i, \quad i = 1,2, ..., I, \tag{3.46}$$

a także do każdego oddziału produkcyjnego należy dowieźć tyle surowców, ile zostało wyznaczone w podsystemie rozdziału, a więc

$$\sum_{i=1}^{I} x'_{i,r} = v_r, r = 1, 2, ..., R. \tag{3.47}$$

Przyjmujemy, że r-ty oddział wytwarza produkt o wielkości $\overline{w}_r = e_r v_r$, $r = 1, 2, ..., R$, gdzie e_r dodatni współczynnik produkcji. Wszystkie produkty o łącznej ilości

$$\overline{W} = \sum_{r=1}^{R} \overline{w}_r = \sum_{r=1}^{R} e_r v_r = \sum_{k=1}^{K} \overline{v}_k \tag{3.48}$$

muszą być przekazane do magazynów produktów. Ponadto zakładamy, że

$$\sum_{k=1}^{K} \overline{x}_{r,k} = \overline{w}_r, \quad r = 1, 2, ..., R, \tag{3.49}$$

czyli całą produkcję każdego oddziału należy rozdysponować między magazyny oraz

$$\sum_{r=1}^{R} \overline{x}_{r,k} = \overline{v}_k, \quad k = 1, 2, ..., K \tag{3.50}$$

co oznacza, że dostawy produktu do każdego magazynu muszą być równe zapotrzebowaniu. Koszty transportu definiujemy tradycyjnie, czyli

$$J_1(\boldsymbol{x}') = \sum_{i=1}^{I} \sum_{r=1}^{R} c'_{i,r} \, x'_{i,r} \tag{3.51}$$

dla transportu surowców oraz

$$J_3(\overline{\boldsymbol{x}}) = \sum_{r=1}^{R} \sum_{k=1}^{K} \overline{c}_{r,k} \overline{x}_{r,k} \tag{3.52}$$

w przypadku transportu produktów. Wówczas problemy przewozu surowców i produktów można sformułować jako klasyczne zbilansowane problemy transportowe.

T r a n s p o r t s u r o w c ó w. Dla danych: całkowitej ilości surowców V, I magazynów, R oddziałów produkcyjnych, jednostkowych kosztów transportu $c'_{i,r}$,

$i = 1, 2, ..., I$, $r = 1, 2, ..., R$, pojemności magazynów surowców w_i, $i = 1, 2, ..., I$ spełniających warunek (3.45), żądanych wielkości surowców dla oddziałów v_r, r = 1, 2, ..., R należy wyznaczyć dopuszczaną w sensie (3.46) i (3.47) macierz decyzyjną $\boldsymbol{x}'$ tak, aby minimalizować kryterium (3.51).

Transport produktów. Dla danych: całkowitej ilości produktów $\overline{W}$, R oddziałów produkcyjnych, K magazynów, jednostkowych kosztów transportu $\overline{c}_{r,k}$, $r = 1, 2, ..., R$, $k = 1, 2, ..., K$, wielkości produkcji $\overline{w}_r$, $r = 1, 2, ..., R$ spełniających warunek (3.48) oraz pojemności magazynów produktów $\overline{v}_k$, $k = 1, 2, ..., K$ należy wyznaczyć dopuszczalną w sensie (3.49) i (3.50) macierz decyzyjną $\overline{\boldsymbol{x}}$ tak, aby minimalizować (3.52).

Całkowity koszt ponoszony w omawianym systemie produkcyjnym jest sumą trzech wprowadzonych kosztów częściowych i wynosi

$$\tilde{J}(\boldsymbol{v}, \boldsymbol{x}', \overline{\boldsymbol{x}}) = J_1(\boldsymbol{v}, \boldsymbol{x}') + J_2(\boldsymbol{v}) + J_3(\boldsymbol{v}, \overline{\boldsymbol{x}}) . \tag{3.53}$$

Zależność J_1 i J_3 od $\boldsymbol{v}$ jest określona w sposób niejawny poprzez ograniczenia (3.47) i (3.49). W przypadku ogólnym wyznaczenie wektora $\boldsymbol{v}^*$ minimalizującego J_2, a następnie szukanie $\boldsymbol{x}'^*$ i $\overline{\boldsymbol{x}}^*$ minimalizujących $J_1(\boldsymbol{v}^*, \boldsymbol{x}')$ i $J_3(\boldsymbol{v}^*, \overline{\boldsymbol{x}})$ nie daje minimalnego kosztu (3.53). Decyzje podejmowane w podsystemach sterowania są zależne w tym sensie, że dla wartości $\boldsymbol{v}$ innych niż wyznaczone w trakcie rozdziału surowca ($\boldsymbol{v} \neq \boldsymbol{v}^*$), czas, a więc w konsekwencji i koszt, może być wprawdzie większy, ale łączny koszt transportu może być mniejszy. Jeśli zwiększenie kosztu produkcji jest mniejsze niż zmniejszenie kosztów transportu, to dla nowych wartości $\boldsymbol{v}$ suma obu rodzajów kosztu będzie mniejsza.

Problem wyznaczania optymalnego rozdziału surowców z uwzględnieniem kosztów transportu surowców i kosztów transportu produktów można zatem sformułować następująco.

Dla danych jak dla trzech podsystemów, należy wyznaczyć takie wartości $\boldsymbol{v} = \hat{\boldsymbol{v}}$, które minimalizują łączny koszt

$$J(\boldsymbol{v}) = \min_{\boldsymbol{x}'} J_1(\boldsymbol{v}, \boldsymbol{x}') + J_2(\boldsymbol{v}) + \min_{\overline{\boldsymbol{x}}} J_3(\boldsymbol{v}, \overline{\boldsymbol{x}}), \tag{3.54}$$

gdzie składniki sumy, pierwszy i trzeci, oznaczają minimalny koszt transportu odpowiednio surowców i produktów dla danego $\boldsymbol{v}$.

Jest to trudny problem optymalizacyjny, nie posiadający rozwiązania analitycznego. Wyznaczenie decyzji sterujących wymaga zastosowania metod numerycznych.

3.4.2. Badanie własności rozwiązania

Opracowanie efektywnego algorytmu rozwiązania dla ogólnego przypadku pozostaje sprawą otwartą i jest obecnie przedmiotem badań. Podamy teraz algorytm rozwiązania dla przypadku dwóch oddziałów produkcyjnych ($R = 2$), przedstawiony w pracy [3]. Stosowana procedura – w przypadku ogólnym – rozpoczyna działanie od wartości $\boldsymbol{v}^*$, a następnie w kolejnych krokach jest dokonywana korekta wektora $\boldsymbol{v}$. Jest stosowany algorytm kolejnych przybliżeń, w którym wyznacza się kolejne η-te przybliżenie $\boldsymbol{v}^{\eta}$ na podstawie porównania wartości $J(\boldsymbol{v})$ dla dwóch poprzednich przybliżeń, to znaczy

$$\boldsymbol{v}^{\eta+1} = G[J(\boldsymbol{v}^{\eta}) - J(\boldsymbol{v}^{\eta-1})],\ \eta = 0,1,... \tag{3.55}$$

z warunkiem początkowym $\boldsymbol{v}^0 = \boldsymbol{v}^*$ oraz $J(\boldsymbol{v}^{-1}) = 0$, gdzie $\boldsymbol{v}^{\eta}$ jest przybliżeniem optymalnego rozwiązania $\hat{\boldsymbol{v}}$ w η -tej iteracji. Istnieją różne sposoby określania konkretnych algorytmów G. Przykładowo, algorytm kroków próbnych polega na zmianie aktualnej wartości $\boldsymbol{v}$ o stały krok próbny $\Delta \boldsymbol{v}$. Dla przypadku dwóch oddziałów produkcyjnych, w którym wystarczy go stosować do wyznaczania v_1, bo $v_2 = V - v_1$, ma on następującą postać

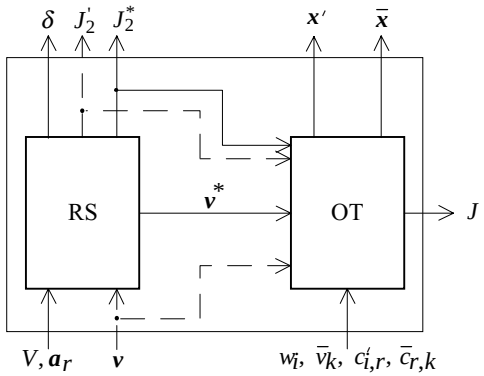
$$v_1^{\eta+1} = v_1^{\eta} + \Delta v_1 \operatorname{sgn}[J(v_1^{\eta-1}) - J(v_1^{\eta})],\ \eta = 1,2,... \tag{3.56}$$

z warunkiem początkowym $v_1^0 = v_1^*$ oraz $v_1^1 = v_1^* + \Delta v_1$. Obliczenia kończymy, gdy dla $\eta > 2$ jest spełniony warunek stopu $J(v_1^{\eta-1}) - J(v_1^{\eta}) \le 0$. Jako rozwiązanie przyjmujemy $\hat{v}_1 \approx v_1^{\eta-1}$ oraz $\hat{v}_2 = V - \hat{v}_1$.

Badania symulacyjne

Ze względu na trudności ze znalezieniem efektywnego algorytmu rozwiązania dla rozdziału surowców z uwzględnieniem kosztów transportu, szczególnego znaczenia nabiera sprawa oceny wpływu kosztów transportu na zmianę wartości $\boldsymbol{v}$ w stosunku do sytuacji, gdy kryterium jakości jest tylko koszt produkcji. Aby sprawdzić ten wpływ, wykonano szczegółowe badania symulacyjne [24]. System programowy do symulacji (rys. 3.21) składa się z dwóch modułów: rozdziału surowców (RS) i optymalizacji transportu (OT). Do wyznaczania rozdziału surowca dla modeli operacji w postaci ogólnej (1.42), dla której nie można otrzymać rozwiązania analitycznego, wykorzystano algorytm neuropodobny. Opis takiego algorytmu szczegółowo przedstawiono w punkcie 4.1.

Ważne są co najmniej dwa cele badań symulacyjnych. Pierwszy z nich polega na ocenie wpływu parametrów modeli operacji na koszt produkcji J_2. Przyjęto do



Rys. 3.21. Schemat systemu programowego do symulacji

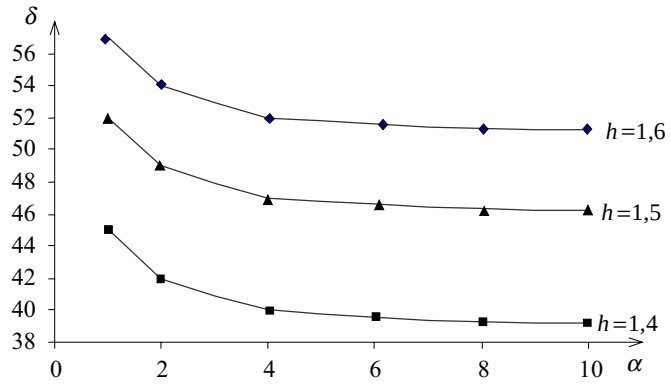
badań modele operacji (1.43), czyli $T_r = k_r v_r^{\alpha}$, $k_r, \alpha > 0$. Ocenie faktycznie podlega zysk z wyznaczania decyzji optymalnej $\boldsymbol{v}^*$ minimalizującej J_2 w stosunku do sytuacji, gdy rozdział zadań jest dokonywany w sposób równomierny, to znaczy $v_1 = v_2 = ... v_R = V/R$. Z drugą sytuacją często można się spotkać w sytuacjach praktycznych, zwłaszcza gdy modele operacji niewiele się od siebie różnią. Zysk ten jest oceniany przez wskaźnik

$$\delta = \frac{J_2' - J_2^*}{J_2'} 100\%, \tag{3.57}$$

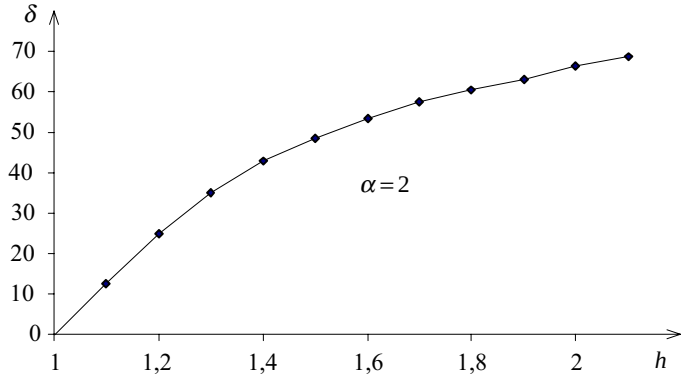
gdzie J_2' jest wartością kosztu produkcji dla równomiernego rozdziału zadań, a J_2^* – dla rozdziału optymalnego $\boldsymbol{v}^*$. W badaniach przyjęto, że $k_{r+1} = h k_r$, $r = 1, 2, ..., R-1$, gdzie współczynnik h określa rozrzut parametrów k_r. Dla danych $R = 4$, $k_1 = 4$, $V = 100$ wyniki przedstawiono na rys. 3.22 i 3.23.

Rozrzut parametrów k_r ma znaczący wpływ na zysk uzyskiwany w przypadku stosowania optymalnego rozdziału zadań. Ponadto, wrażliwość δ na zmiany α jest bardzo mała dla $\alpha > 3$. Oznacza to, że dokładna znajomość modelu w takim przypadku nie jest konieczna.

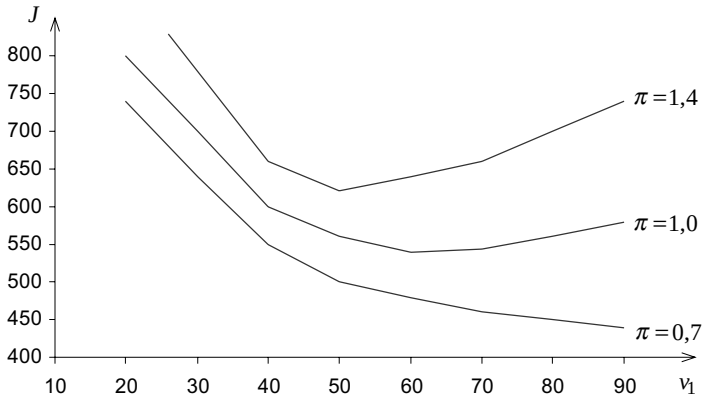
Drugi ważny cel badań symulacyjnych polega na określeniu charakteru zależności między J a $\boldsymbol{v}$. W badaniach, których rezultaty przedstawiono na rys. 3.24, przyjęto, że $J_3 \equiv 0$ oraz $R = 2$, $I = 3$, $k_1 = \frac{1}{2}$, $k_2 = \frac{1}{3}$, $\alpha_1 = 10$, $\alpha_2 = 60$, $V = 100$, $c'_{1,1} = 1{,}1$, $c'_{1,2} = 8{,}1$, $c'_{2,1} = 1{,}2$, $c'_{2,2} = 6{,}3$, $c'_{3,1} = 0{,}8$, $c'_{3,2} = 7{,}2$. Na badaną zależność duży wpływ ma wartość współczynnika π oraz wzajemne stosunki między jed-



Rys. 3.22. Zależność δ od α



Rys. 3.23. Zależność δ od h



Rys. 3.24. Zależność J od v_1 dla $J_3 \equiv 0$

nostkowymi kosztami transportu surowca $c'_{i,r}$ a parametrami modeli operacji k_r. Jeśli koszt produkcji w *r*-tym oddziale jest większy niż koszt transportu surowca do tego oddziału, to dla odpowiednio małych π zwiększenie v_r z wartości v_r^* może spowodować znaczną redukcję kosztu *J*. Na rys. 3.24 $v_1^* = 43$ oraz

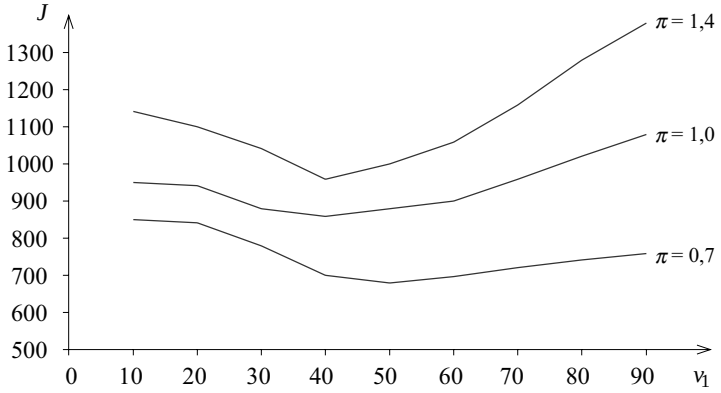
dla $\pi = 1{,}4$ – $\hat{v}_1 = 50$, $J(v_1^*) - J(\hat{v}_1) = 10$,

dla $\pi = 1{,}0$ – $\hat{v}_1 = 64$, $J(v_1^*) - J(\hat{v}_1) = 42$,

dla $\pi = 0{,}7$ – $\hat{v}_1 = 100$, $J(v_1^*) - J(\hat{v}_1) = 92$.

Dla $\pi = 0{,}7$, gdy J_2 jest wystarczająco małą częścią *J*, zmiana wielkości zadania v_1^* minimalizującego J_2 na wielkość $\hat{v}_1$ minimalizującą *J* powoduje 17% zmniejszenie globalnego kosztu.

W kolejnym badaniu uwzględniono już koszt transportu produktów. Przyjęto dane jak w poprzednim badaniu oraz $K = 3$, $e_1 = e_2 = 1$. Dla $\pi = 0{,}7$ i $\pi = 1{,}4$ jednostkowe koszty transportu produktów są następujące: $\bar{c}_{1,1} = 8{,}1$, $\bar{c}_{1,2} = 6{,}3$, $\bar{c}_{1,3} = 7{,}2$, $\bar{c}_{2,1} = 1{,}1$, $\bar{c}_{2,2} = 1{,}2$, $\bar{c}_{2,3} = 0{,}8$. Dla $\pi = 1{,}0$ wartości tych kosztów są dwa razy mniejsze (rys. 3.25).



Rys. 3.25. Zależność *J* od v_1 dla $J = J_1 + J_2 + J_3$

W tym przypadku wzrost v_1 z wartości v_1^* powoduje zmniejszenie kosztu transportu surowców J_1 oraz zwiększenie kosztu transportu produktów J_3. Postać korekty zmiennej $\boldsymbol{v}^*$ do wartości $\hat{\boldsymbol{v}}$ zależy od wartości składowych J_1, J_2, J_3 kosztu całkowitego *J*. Badania prowadzone dla różnych danych pokazały, że jeżeli korekta $\boldsymbol{v}$ z wartości $\boldsymbol{v}^*$ powoduje zmniejszenie (zwiększenie) J_1 oraz zwiększenie (zmniejszenie) J_3, to obszar, w którym znajduje się minimum $J(\boldsymbol{v})$, jest bardziej płaski niż w przypadku $J_3 \equiv 0$.

■

Równoważność rozwiązań

Badania symulacyjne umożliwiają między innymi sformułowanie wniosku, że wartość różnicy $J(\boldsymbol{v}^*) - J(\hat{\boldsymbol{v}})$ w dużym stopniu zależy od parametrów problemu, a zwłaszcza od wartości parametrów modeli k_r i jednostkowych kosztów transportu. Zerowa wartość tej różnicy oznaczałaby, że bez utraty optymalności jest możliwa dekompozycja problemu na niezależnie rozwiązywane: zagadnienie rozdziału surowców minimalizującego J_2 oraz problemy transportowe. Interesujące byłoby też w tym kontekście postawienie pytania o jeszcze większej wadze: Czy można pominąć koszt transportu bez straty optymalności rozwiązania problemu (3.54)? Okazuje się, że dla pewnych założeń odpowiedź na to pytanie może być twierdząca. Oznacza to, że można sformułować warunki, dla których zachodzą równości $v_r^* = \hat{v}_r$, $r = 1, 2, ..., R$, nazywane własnościami równoważności. Równoważność w tym sensie oznacza, że można zastąpić rozwiązywanie trudnego problemu optymalizacyjnego (3.54) w celu wyznaczenia $\hat{\boldsymbol{v}}$, rozwiązywaniem klasycznego problemu rozdziału surowców z kryterium J_2 i uzyskaniem $\boldsymbol{v}^*$. Jak się okazuje, jest ona prawdziwa dla szczególnego przypadku z jednym magazynem surowców (I = 1) i jednym magazynem produktów (K = 1). Wtedy oczywiście nie ma problemu transportowego, ale uproszczony koszt transportu nadal pozostaje. Kryterium (3.54) możemy przedstawić jako

$$\hat{J}(\boldsymbol{v}) = \max_{r=1, 2, ..., R} \{\hat{\gamma}_1(v_1), \hat{\gamma}_2(v_2), ..., \hat{\gamma}_R(v_R)\} + \sum_{r=1}^{R} c_r \, v_r \,, \tag{3.58}$$

gdzie $c_r = c'_{1,r} + \bar{c}_{r,1} \, e_r$ oraz $\hat{\gamma}_r(v_1) = \pi \, \bar{\gamma}_r(v_r)$, dla $r = 1, 2, ..., R$. Załóżmy, że funkcje $\bar{\gamma}_r$ są różniczkowalne i istnieją pochodne $\dfrac{d\hat{\gamma}_r(v_r)}{dv_r} = \hat{\gamma}'_r(v_r)$ oraz $\hat{\gamma}''_r(v_r) > 0$ dla każdego r, a także $v_r \in [0, V]$. Wprowadźmy dodatkowo oznaczenia

$$\max_{0 \le v_r \le V} \hat{\gamma}'_r(v_r) = \hat{\gamma}'_r(V) = \bar{\kappa}_r \,,$$

$$\min_{0 \le v_r \le V} \hat{\gamma}'_r(v_r) = \hat{\gamma}'_r(0) = \underline{\kappa}_r \,.$$

Wtedy jest prawdziwe twierdzenie [27].

Twierdzenie 3.1

Jeżeli

$$(R-1)^2 \cdot \frac{\max\limits_{r=1,2,...,R}\{c_r\}}{\min\limits_{r=1,2,...,R}\{\underline{\kappa}_r\}} - \min_{r=1,2,...,R}\left\{\frac{c_r}{\overline{\kappa}_r}\right\} \leq 1, \tag{3.59}$$

to $\hat{v}_r = v_r^*$, $r = 1, 2, ..., R$.

Dowód

Załóżmy, że $v_r = v_r^* + \varepsilon_r$, gdzie $\varepsilon_r > 0$. Wystarczy wykazać, że dla dowolnych ε_r, $r = 1, 2, ..., R$ jest prawdziwa nierówność

$$\hat{J}(\boldsymbol{v}) - \hat{J}(\boldsymbol{v}^*) \stackrel{\Delta}{=} \Delta \geq 0, \tag{3.60}$$

gdzie $\hat{J}$ jest określone przez (3.58). Niech

$$\hat{\gamma}_s(v_s^* + \varepsilon_s) = \max_{r=1,2,...,R}\{\hat{\gamma}_r(v_r^* + \varepsilon_r)\}. \tag{3.61}$$

Dla $v_1^*, v_2^*, ..., v_R^*$ czasy T_r są równe, to znaczy $\hat{\gamma}_1(v_1^*) = \hat{\gamma}_1(v_2^*) = \cdots = \hat{\gamma}_1(v_R^*)$. Wtedy (3.60) można przedstawić jako

$$\Delta = \hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*) + \sum_{r=1}^{R} c_r\, \varepsilon_r\ .$$

Korzystając z założenia o funkcjach $\hat{\gamma}_r$ oraz (3.61) otrzymujemy

$$\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*) \geq \hat{\gamma}_r(v_r^* + \varepsilon_r) - \hat{\gamma}_r(v_r^*) \geq \hat{\gamma}_r'(v_r^*)\varepsilon_r$$

oraz

$$\varepsilon_r \leq \frac{\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)}{\underline{\kappa}_r},\ r = 1, 2, ..., R. \tag{3.62}$$

Z warunku $\sum_{r=1}^{R} v_r = V$ wynika, że $\sum_{r=1}^{R} \varepsilon_r = 0$, to znaczy

$$\varepsilon_r = -\sum_{\substack{l=1 \\ l \neq r}}^{R} \varepsilon_l\ . \tag{3.63}$$

Z zależności (3.62) i (3.63) uzyskujemy

$$\varepsilon_r \geq -(R-1)\frac{\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)}{\min\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{\underline{\kappa}_r\}}, \quad r = 1,2,\ldots,R.$$

Wówczas

$$\Delta \geq \hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*) + c_s\varepsilon_s - (R-1)\frac{\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)}{\min\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{\underline{\kappa}_r\}} \sum_{\substack{l=1 \\ l\neq s}} c_l$$

$$\geq \hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*) + c_s\varepsilon_s - [\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)](R-1)^2 \frac{\max\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{c_r\}}{\min\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{\underline{\kappa}_r\}}.$$

Stąd

$$\frac{\Delta}{\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)} \geq 1 + \frac{c_s\varepsilon_s}{\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)} - (R-1)^2 \frac{\max\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{c_r\}}{\min\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{\underline{\kappa}_r\}}.$$

Łatwo zauważyć, że dla założeń dotyczących $\hat{\gamma}_s$

$$\frac{\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)}{\varepsilon_s} \leq \hat{\gamma}_s'(V) = \overline{\kappa}_s,$$

wtedy

$$\frac{\Delta}{\hat{\gamma}_s(v_s^* + \varepsilon_s) - \hat{\gamma}_s(v_s^*)} \geq 1 + \frac{c_s}{\overline{\kappa}_s} - (R-1)^2 \frac{\max\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{c_r\}}{\min\limits_{\substack{r=1,2,\ldots,R \\ r\neq s}} \{\underline{\kappa}_r\}},$$

czyli gdy nierówność (3.59) jest spełniona, wówczas $\Delta \geq 0$, a więc $\hat{v}_r = v_r^*$, $r = 1, 2, \ldots, R$.

■

Dla modeli liniowych $T_r = k_r\, v_r$, $r = 1, 2, ..., R$ kryterium (3.58) można sprowadzić do postaci

$$\hat{J}(\boldsymbol{v}) = \pi \max_{r=1,2,...,R} \{k_1 v_1, k_2 v_2, ..., k_R v_R\} + \sum_{r=1}^{R} c_r v_r \,. \tag{3.64}$$

Pochodne funkcji $\hat{\gamma}_r$ względem v_r są niezależne od v_r i równe k_r, a więc $\underline{\kappa}_r = \overline{\kappa}_r \triangleq k_r$. Stąd warunek (3.59) przyjmuje postać

$$(R-1)^2 \frac{\max\limits_{r=1,2,...,R} \{c_r\}}{\min\limits_{r=1,2,...,R} \{k_r\}} - \min_{r=1,2,...,R} \left\{\frac{c_r}{k_r}\right\} \le 1. \tag{3.65}$$

Dla tego szczególnego przypadku oraz dla dwóch oddziałów produkcyjnych ($R = 2$) łatwo można wyznaczyć rozwiązanie optymalne $\hat{\boldsymbol{v}}$, wykorzystując proste rozważania geometryczne, bez stosowania omówionego poprzednio iteracyjnego algorytmu kolejnych przybliżeń.

Przykład 3.3

Dla $R = 2$ kryterium (3.64) zapisujemy jako

$$\hat{J}(\boldsymbol{v}) = \pi \max\{k_1 v_1, k_2 v_2\} + c_1 v_1 + c_2 v_2 \,.$$

Zgodnie z (1.55)

$$v_1^* = V \frac{k_2}{k_1 + k_2}, \quad v_2^* = V \frac{k_1}{k_1 + k_2} \,.$$

Wystarczy znaleźć minimum funkcji

$$F(v_1) = \max\{k_1 v_1, k_2 (V - v_1)\} + c_1 v_1 + c_2 (V - v_1) \,.$$

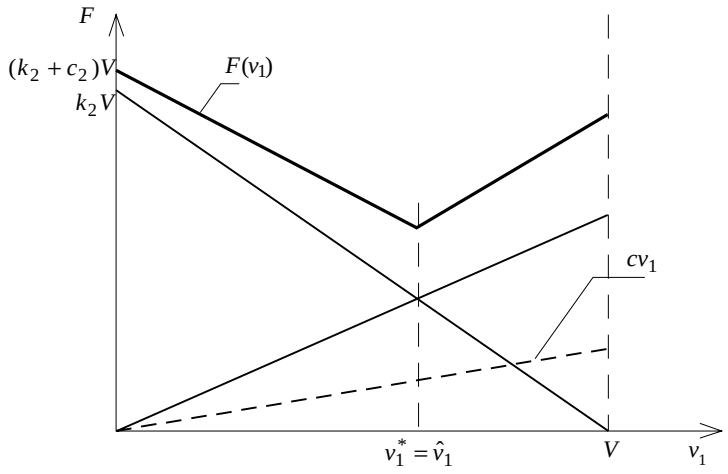
Przykładowy przebieg tej funkcji przedstawiono na rys. 3.26.

Można sformułować następujące reguły dotyczące funkcji $F(v_1)$ [27]:

1. Jeśli $F(v_1)$ jest rosnąca dla $v_1 < v_1^*$, to $\hat{v}_1 = 0$.

2. Jeśli $F(v_1)$ jest malejąca dla $v_1 < v_1^*$ i rosnąca dla $v_1 > v_1^*$ (rys. 3.26), to $\hat{v}_1 = v_1^*$.

3. Jeśli $F(v_1)$ jest malejąca dla $v_1 > v_1^*$, to $\hat{v}_1 = V$.



Rys. 3.26. Wykres funkcji $F(v_1)$

Jeśli $F(v_1)$ jest rosnąca dla $v_1 < v_1^*$, to jest również rosnąca dla $v_1 > v_1^*$ oraz jeśli jest malejąca dla $v_1 > v_1^*$, to jest też malejąca dla $v_1 < v_1^*$. Biorąc pod uwagę wartości parametrów k_1, k_2, $c_2 - c_1 \triangleq c$, od których zależy monotoniczność funkcji, reguły te można zapisać jako:

1. Jeśli $c - k_2 > 0$, to $\hat{v}_1 = 0$, $\hat{v}_2 = V$.
2. Jeśli $c - k_2 < 0$ i $c + k_1 > 0$, to $\hat{v}_1 = v_1^*$, $\hat{v}_2 = v_2^*$.
3. Jeśli $c + k_1 < 0$, to $\hat{v}_1 = V$, $\hat{v}_2 = 0$.

Ponadto, jeśli $c - k_2 = 0$, to $\hat{v}_1$ może przyjąć dowolną wartość z przedziału $[0, v_1^*]$. W pracy [27] podano analogiczne reguły dla operacji o modelach ogólnych $T_r = \bar{\gamma}_r(v_r)$.

■

Zakończmy te rozważania uwagą na temat innej możliwości uwzględniania kosztów transportu podczas rozdziału surowca między oddziały produkcyjne. Przedstawiony dotychczas sposób polegał na rozszerzeniu kryterium jakości bez zmiany modeli operacji. Istnieje możliwość odwrotna, która polega na pozostawieniu kryterium w postaci jedynie kosztu produkcji i rozszerzeniu modeli operacji. Według tej koncepcji modele operacji dla $I = K = 1$ mają postać

$$T_r = \bar{\gamma}_r(v_r) + \tilde{c}_r\, v_r \triangleq \tilde{\gamma}_r(v_r), \tag{3.66}$$

gdzie $\tilde{c}_r = c''_{1,r} + e_r\, \bar{c}''_{r,1}$ oraz $c''_{1,r}$ i $\bar{c}''_{r,1}$ są jednostkowymi czasami transportu surowców i produktów. Wówczas otrzymujemy klasyczne zagadnienie rozdziału surowców z modelami $\tilde{\gamma}_r$ w miejsce modeli $\bar{\gamma}_r$ oraz kryterium

$$\tilde{J}(\boldsymbol{v}) = \pi \max_{r=1,2,...,R} \{\tilde{\gamma}_1(v_1), \tilde{\gamma}_2(v_2), ..., \tilde{\gamma}_R(v_R)\}. \tag{3.67}$$

Jeśli wynik minimalizacji (3.67) oznaczymy jako $\boldsymbol{v}_{\min}$, a wynik minimalizacji J_2 jako $\boldsymbol{v}^*$, to dla przyjętych założeń o różniczkowalności $\bar{\gamma}_r$, jest prawdziwe twierdzenie o równoważności rozwiązań [27] – analogiczne do twierdzenia 3.1.

Twierdzenie 3.2

Jeżeli

$$\min_{r=1,2,...,R} \left\{ \frac{\tilde{c}_r}{\tilde{\kappa}_r} \right\} \le 1, \tag{3.68}$$

to

$$v_{\min,r} = v_r^*, \quad r = 1, 2, ..., R,$$

gdzie $\tilde{\kappa}_r = \max_{0 \le v_r \le V} \tilde{\gamma}'_r(v_r) = \tilde{\gamma}'_r(V)$.

Dowód

Dowód jest podobny jak w twierdzeniu 3.1. Utworzona jak w (3.60) różnica $\tilde{\Delta}$ między $\tilde{J}_2(\boldsymbol{v})$ a $\tilde{J}_2(\boldsymbol{v}^*)$ wynosi teraz

$$\tilde{\Delta} = \tilde{\gamma}_s(v_s^* + \varepsilon_s) - \tilde{\gamma}_s(v_s^*) + \tilde{c}_s\, \varepsilon_s. \tag{3.69}$$

Korzystając z nierówności $\tilde{\gamma}_s(v_s^* + \varepsilon_s) - \tilde{\gamma}_s(v_s^*) \le \bar{\kappa}_s\, \varepsilon_s$, wyrażenie (3.69) możemy przekształcić i oszacować w sposób następujący

$$\frac{\tilde{\Delta}}{\tilde{\gamma}_s(v_s^* + \varepsilon_s) - \tilde{\gamma}_s(v_s^*)} = 1 + \frac{\tilde{c}_s\, \varepsilon_s}{\tilde{\gamma}_s(v_s^* + \varepsilon_s) - \tilde{\gamma}_s(v_s^*)} \ge 1 + \frac{\tilde{c}_s}{\bar{\kappa}_s}$$

$$\ge 1 + \min_{r=1,2,...,R} \left\{ \frac{\tilde{c}_r}{\tilde{\kappa}_r} \right\}.$$

Jeśli jest prawdziwy warunek (3.68), to zachodzi nierówność $\tilde{\Delta} \ge 0$, a więc $v_{\min,r} = v_r^*$, $r = 1, 2, ..., R$.

■

Podobnie jak dla (3.64), również i teraz dla modeli liniowych warunek (3.68) jest prostszy, a mianowicie

$$\min_{r=1,2,\ldots,R}\left\{\frac{\tilde{c}_r}{k_r}\right\}\leq 1.$$

Ocena analityczna własności rozwiązania dla bardziej złożonych struktur systemu produkcyjnego jest bardzo trudna. Pewne wyniki można uzyskać, przeprowadzając komputerowe badania symulacyjne. Wszechstronne badania przedstawionego w tym punkcie problemu rozdziału zadań w kompleksie operacji z uwzględnieniem transportu surowców i produktów są obecnie prowadzone. Dotyczą one nie tylko własności rozwiązań, ale przede wszystkim wyznaczania efektywnych algorytmów sterowania. Jest rozpatrywany zarówno przypadek deterministyczny, jak i wersje niedeterministyczne problemu. Do modelowania kompleksu operacji w warunkach niedeterministycznych są stosowane oprócz klasycznych modeli probabilistycznych opisy wykorzystujące zmienne niepewne oraz wybraną logikę rozmytą. Krótka charakterystyka tych i innych metod sztucznej inteligencji w zastosowaniu do wybranych problemów decyzyjnych dla kompleksów operacji jest przedstawiona w punkcie 4.1.

4. Metody sztucznej inteligencji w kompleksach operacji

4.1. Wprowadzenie

Wzrost technicznych możliwości realizacji coraz bardziej złożonych algorytmów rozwiązywania różnych problemów decyzyjnych spowodował w ostatnim okresie gwałtowny rozwój metod opracowywania takich algorytmów. Dotyczy to między innymi metod tradycyjnych, na przykład takich, jakie przedstawiono w poprzednich rozdziałach. W tym przypadku rozwój nowoczesnych środków informatyki umożliwia przede wszystkim efektywną realizację algorytmów dla problemów bardziej złożonych, a więc bliższych zastosowaniom praktycznym oraz dla problemów o większej wymiarowości. Co ważniejsze jednak, rozwój współczesnych środków informatyki był i jest inspiracją do opracowywania nowych metod rozwiązywania trudnych problemów decyzyjnych oraz do powrotu i twórczego rozwijania pomysłów, które opracowane przed wielu laty, nie miały wtedy możliwości praktycznego zastosowania. Należą do nich między innymi metody często opatrywane wspólną nazwą metod sztucznej inteligencji (ang. *artificial intelligence methods*). Pojęcie to nie jest ściśle określone, ale zwykle wiąże się z algorytmizacją i formalizacją różnych procesów rozumowania i wnioskowania oraz uczenia, percepcji, rozpoznawania, kojarzenia itp. Stosowanie tego typu metod wiąże się zwykle z nietradycyjnym opisem obiektu podejmowania decyzji i (lub) algorytmu decyzyjnego w postaci tak zwanej reprezentacji wiedzy o obiekcie i (lub) systemie decyzyjnym, np. [19, 22].

Spośród wielu szczegółowych metod w tym zakresie należy wymienić najbardziej rozpowszechnione, a mianowicie: metody ewolucyjne (ang. *evolutionary methods*), prowadzące do wyznaczania najbardziej rozpowszechnionych algorytmów genetycznych lub szerzej algorytmów ewolucyjnych, np. [87, 46, 34]; metody wykorzystujące sztuczne sieci neuronowe (ang. *neural nets*) (inne powszechnie stosowane nazwy to: sieci neuronowe, np. [108, 109, 81, 30, 37, 101, 91], sieci neuronalne) oraz metody oparte na logice i zbiorach rozmytych (ang. *fuzzy logics and sets*), np. [117, 79, 35, 94, 116, 32, 112] lub inne nieklasyczne sposoby opisu niepewności.

W metodach ewolucyjnych, których rozwój był inspirowany przez procesy zachodzące w organizmach biologicznych i które przejęły nomenklaturę z takich systemów, zmienne decyzyjne są przedstawiane w postaci chromosomów i są oceniane za pomocą funkcji przystosowania, będących odpowiednikiem tradycyjnych funkcji celu. Są rozpatrywane zbiory chromosomów, tworzące populacje. Wyznaczane konkretne algorytmy rozwiązania (algorytmy genetyczne, algorytmy ewolucyjne) operują na populacjach chromosomów i wartościując je z wykorzystaniem funkcji przystosowania, prowadzą w kolejnych przybliżeniach (iteracjach) do odpowiednich zmian (ewolucji) populacji w celu poszukiwania najlepszych rozwiązań.

Sztuczna sieć neuronowa, w dużym przybliżeniu wzorowana na biologicznej sieci neuronów (tkance nerwowej), składa się z elementów – neuronów, które przekazują między sobą informacje. Odpowiednio zaprojektowana sieć (jedno- lub wielowarstwowa, jednokierunkowa lub rekurencyjna) jest traktowana jako model rozwiązywanego problemu decyzyjnego. Istotą przetwarzania informacji przez sztuczne sieci neuronowe jest dopasowywanie takiego modelu rozwiązania w kolejnych cyklach na drodze uczenia z nauczycielem lub bez nadzoru.

Metody oparte na zbiorach rozmytych umożliwiają z kolei bardziej ogólny i uniwersalny sposób opisu obiektów i (lub) systemów podejmowania decyzji oraz modelowanie takich zjawisk i pojęć, które mają charakter nieprecyzyjny i wieloznaczny. Uogólnieniu podlegają również sposoby wnioskowania oraz podejmowania decyzji (sterowania), wykorzystujące logikę rozmytą, a więc bardziej ogólną niż tradycyjna logika dwuwartościowa. Argumentami funkcji logicznych są zwykle tak zwane zmienne lingwistyczne. W procesach podejmowania decyzji można pominąć wyznaczanie modeli obiektów i wprost zastosować techniki zbiorów rozmytych do określania decyzji. Wtedy formułowane w logice rozmytej reguły postępowania dotyczą wprost sposobu podejmowania decyzji, np. sterowania, z pominięciem modelu obiektu. W tym przypadku w sposób szczególny jakość decyzji zależy od posiadanych informacji, zawartych w wiedzy o prawidłowościach podejmowania decyzji, określanej przez eksperta.

Wymienione trzy grupy metod dla konkretnych zastosowań mogą być łączone [102]. Na przykład można wykorzystać sztuczne sieci neuronowe do tworzenia reguł wnioskowania w logice rozmytej. Powszechnie stosowane jest też jednoczesne korzystanie z algorytmów genetycznych i sztucznych sieci neuronowych. Algorytmy genetyczne są stosowane wówczas w procesie uczenia sieci lub do wyznaczania jej topologii. Innym przykładem jest zastosowanie sztucznych sieci neuronowych do tworzenia populacji początkowych w algorytmach genetycznych. To wzajemne przenikanie rozważanych metod sprawia, że stosuje się na nie łącznie wspólną na-

zwę angielską *computational intelligence* lub *soft computing*, która nie ma jeszcze w języku polskim ustalonego odpowiednika. Nazwy angielskie wskazują na nowe, uogólnione i bardziej uniwersalne metody przetwarzania informacji, których duża efektywność jest spowodowana coraz to nowymi rezultatami metodologicznymi, ale również rozwojem środków informatyki. W tym drugim przypadku wymienimy dla przykładu przetwarzanie współbieżne i równoległe w systemach wielokomputerowych i rozproszonych, a także neurokomputery.

Zwróćmy jeszcze krótko uwagę na trzy pojęcia: system ekspertowy, niepewność i uczenie.

Jest oczywiste, że obecnie nowoczesne środki informatyki (systemy informatyczne) pełnią rolę wspomagającą lub doradczą w różnych zagadnieniach podejmowania decyzji. Niektóre cechy systemów informatycznych w rozważanych zastosowaniach, które są charakterystyczne dla omawianych metod sztucznej inteligencji, powodują, że takie systemy są określane mianem systemów ekspertowych. Są to przede wszystkim: wbudowane mechanizmy wnioskowania na podstawie przyjętych reguł oraz ogólny opis wiedzy o obiekcie w postaci reprezentacji wiedzy. Tak więc, konkretna realizacja wspomnianych metod sztucznej inteligencji odbywa się w systemie ekspertowym. Warto zauważyć, że system ekspertowy to nie tylko specjalizowane środki informatyki (sprzęt, oprogramowanie), ale również użytkownik, jeśli współdziała z systemem informatycznym, a także ekspert, jeśli w trakcie działania systemu podaje aktualną wiedzę o obiekcie podejmowania decyzji.

Jak już wspomniano, metody sztucznej inteligencji umożliwiają formułowanie ogólnych opisów również dla takich obiektów, które nie są dokładnie znane, lub które podlegają szybkim zmianom, czyli dla których nie jest możliwy lub wygodny opis deterministyczny. Systemy, w których występują takie obiekty, nazywamy systemami niepewnymi (ang. *uncertain systems*). Istnieje wiele sposobów opisu (modeli) niepewności. Niektóre z nich, jak na przykład modele probabilistyczne, modele growe, są stosowane od dawna. Inne, również dobrze znane, w metodach sztucznej inteligencji znalazły nowe możliwości realizacji. Przykładem mogą być na przykład modele relacyjne, w których wiedza o obiekcie i (lub) systemie podejmowania decyzji jest dana nie w postaci funkcyjnej, ale za pomocą relacji. Uogólnione sposoby rozumowania stosowane w algorytmach sztucznej inteligencji umożliwiają efektywne wykorzystanie takich opisów. W omawianych metodach pojawiły się jednak nowe sposoby opisu niepewności. Najbardziej znanym z nich są zbiory rozmyte z funkcją przynależności, która w praktycznych zastosowaniach jest wygodniejsza niż rozkłady prawdopodobieństwa dla modeli probabilistycznych. Z innych nowych opisów niepewności warto wspomnieć o zbiorach przybliżonych [93] i zmiennych niepewnych [26].

Wspomnianą już i bardzo ważną cechą metod sztucznej inteligencji jest uczenie, umożliwiające poprawianie na bieżąco wyznaczanego rozwiązania, na podstawie informacji uzyskiwanych w trakcie procesu podejmowania decyzji. Można tu krótko wspomnieć o dwóch koncepcjach uczenia: o uczeniu z nauczycielem i o uczeniu bez nauczyciela. W pierwszej z nich decyzje podejmowane według algorytmu, którego postać ma podlegać ustaleniu w trakcie uczenia, w kolejnych cyklach są konfrontowane z poprawnymi informacjami, które ma tak zwany nauczyciel. Sposób ustalania postaci algorytmu decyzyjnego nosi nazwę algorytmu uczenia, które odbywa się na podstawie dużej liczby przykładów, tworzących tak zwany ciąg uczący, opracowany przez nauczyciela. Elementami ciągu są pary składające się z informacji wejściowej (w szczególności wyników obserwacji) oraz poprawnych wyników lub odpowiedzi podawanych przez eksperta. Jest to tak zwana koncepcja uczenia na przykładach. Drugi sposób uczenia, a właściwie samouczenia już bez udziału nauczyciela polega na korekcie algorytmu decyzyjnego w kolejnych krokach, na podstawie obserwacji skutków podjętych wcześniej decyzji, ocenianych przez przyjęte kryterium jakości. Ten sposób uczenia można by nazwać uczeniem na własnych błędach.

Zakres zastosowań scharakteryzowanych metod rozwiązywania problemów decyzyjnych jest bardzo szeroki. Obejmuje on również problemy decyzyjne dla systemów dyskretnych i operacyjnych, w tym również dla kompleksów operacji, chociaż z pewnością nie jest to ich główny obszar zastosowań. Większość problemów decyzyjnych – również dla systemów operacyjnych – polega na rozwiązywaniu zagadnień optymalizacyjnych. W tym celu można oczywiście wykorzystywać algorytmy ewolucyjne lub sztuczne sieci neuronowe, ale warto wspomnieć i o innych sposobach poszukiwania optimów lokalnych, na przykład: metody przeszukiwań losowych *tabu search*, symulowanego wyżarzania.

W ramach jednego rozdziału nie jest oczywiście możliwe przedstawienie całej problematyki zastosowania metod sztucznej inteligencji do podejmowania decyzji w kompleksach operacji. Dlatego dokonano wyboru trzech problemów, opisanych we wcześniejszych rozdziałach, które potraktowano jako przykłady. Postanowiono zilustrować zastosowanie również wybranych metod sztucznej inteligencji do rozwiązania tych problemów. I tak, w punkcie 4.2 przedstawiono prosty model sztucznej sieci neuronowej oraz algorytm uczenia bez nauczyciela w zastosowaniu do rozwiązania problemu alokacji dla szczególnych przypadków. Przedmiotem rozważań w punkcie 4.3 jest zagadnienie szeregowania z ruchomymi realizatorami. Minimalizacja długości uszeregowania jest przeprowadzana z wykorzystaniem algorytmu ewolucyjnego. Wreszcie w punkcie 4.4 jest rozważany problem koordynacji jazdy

grupy pojazdów autonomicznych, szczegółowo przedstawiony w punkcie 3.2. Algorytm koordynacji jazdy wykorzystuje procedurę rozpoznawania z reprezentacją wiedzy.

4.2. Sztuczne sieci neuronowe i algorytmy uczenia w alokacji

4.2.1. Algorytm neuropodobny

Rozważania w tym punkcie ograniczymy do przypadku czasowo-optymalnego rozdziału zasobów lub zadań w kompleksie operacji niezależnych, który szczegółowo przedstawiono w podpunktach 1.2.1 i 1.2.2. Generalnie utrzymamy oznaczenia wprowadzone w rozdziale 1. Jedyna różnica będzie polegać na rozróżnieniu zapisu modeli oraz algorytmów rozwiązania, które w rozdziale 1. dla uproszczenia przyjmowaliśmy jako takie same. I tak, przypomnijmy, że dla modeli operacji $T_r = \gamma_r(u_r; \boldsymbol{a}_r)$ – w przypadku rozdziału zasobów oraz $T_r = \gamma'_r(v_r; \boldsymbol{a}_r)$ – w przypadku rozdziału zadań, ogólnie analityczne postaci algorytmów rozwiązania przedstawialiśmy odpowiednio jako $u_r^* = \eta_r(U, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R) = \overline{\eta}_r(\boldsymbol{a})$ oraz $v_r^* = \eta'_r(V, \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_R) = \overline{\eta}'_r(\boldsymbol{a})$, gdzie $\boldsymbol{a} = [\boldsymbol{a}_1^{\mathrm{T}}, \boldsymbol{a}_2^{\mathrm{T}}, ..., \boldsymbol{a}_R^{\mathrm{T}}]^{\mathrm{T}}$. Sztuczne sieci neuronowe można stosować do rozwiązywania takich problemów alokacji w dwóch przypadkach. Wtedy, gdy nie istnieje analityczne rozwiązanie, czyli gdy nie można wyznaczyć algorytmów η_r oraz η'_r i należałoby stosować procedury numeryczne, tak jak to opisano w punkcie 1.2, oraz gdy wprawdzie istnieje rozwiązanie analityczne, ale ze względu na rozmiar problemu czas jego wyznaczenia jest zbyt długi. Przedstawione tu rozważania zostały oparte na pracach [23, 90]. Ogólna idea stosowania sztucznych sieci neuronowych do rozwiązywania tego typu zagadnień decyzyjnych polega na zastąpieniu algorytmów η_r (η'_r) lub odpowiednich algorytmów numerycznych przez algorytmy neuropodobne, aproksymujące te algorytmy. Prosty algorytm neuropodobny może mieć postać

$$y_r = f_r(\boldsymbol{w}_r^{\mathrm{T}} \boldsymbol{x}_r + b_r) \triangleq g_r(\boldsymbol{x}_r, \overline{\boldsymbol{a}}_r), \tag{4.1}$$

gdzie: y_r – jednowymiarowe wyjście,
$\boldsymbol{x}_r = [x_{r,1}, x_{r,2}, ..., x_{r,I}]^{\mathrm{T}}$ – I-wymiarowy wektor wejściowy,
$\boldsymbol{w}_r = [w_{r,1}, w_{r,2}, ..., w_{r,I}]^{\mathrm{T}}$ – I-wymiarowy wektor wag,
b_r – wartość progowa,
$\overline{\boldsymbol{a}}_r = [\boldsymbol{w}_r^{\mathrm{T}}, b_r]^{\mathrm{T}}$ – wektor parametrów.

Funkcje f_r mogą mieć typowe postaci, na przykład liniową lub sigmoidalną. W sieciach wielowarstwowych, odpowiadających bardziej złożonym algorytmom

neuropodobnym, wejścia elementów (neuronów) z warstw dalszych są wyjściami elementów (neuronów) z warstw bezpośrednio poprzedzających. Na przykład dla dwóch warstw (4.1) opisuje warstwę wyjściową, a dla warstwy poprzedniej można podać zależność

$$x_{r,i} = f_{r,i}(\boldsymbol{w}_r^{(i)\mathrm{T}} \boldsymbol{z}_r^{(i)} + b_r^{(i)}) \stackrel{\Delta}{=} g_{r,i}(\boldsymbol{z}_r^{(i)}, \tilde{\boldsymbol{a}}_r^{(i)}),\ i = 1, 2, ..., I\ , \tag{4.2}$$

gdzie: $\boldsymbol{z}_r^{(i)} = [z_{r,1}^{(i)}, z_{r,2}^{(i)}, ..., z_{r,L}^{(i)}]^{\mathrm{T}}$ – L-wymiarowy wektor wejściowy i-tego elementu,

$\boldsymbol{w}_r^{(i)} = [w_{r,1}^{(i)}, w_{r,2}^{(i)}, ..., w_{r,L}^{(i)}]^{\mathrm{T}}$ – L-wymiarowy wektor wag i-tego elementu,

$b_r^{(i)}$ – wartość progowa i-tego elementu,

$\tilde{\boldsymbol{a}}_r^{(i)} = [\boldsymbol{w}_r^{(i)\mathrm{T}}, b_r^{(i)}]^{\mathrm{T}}$ – wektor parametrów i-tego elementu.

Podobnie jak w warstwie wyjściowej, liczba wejść każdego elementu tej warstwy jest taka sama i wynosi L. Łącząc algorytmy (4.1) i (4.2) otrzymujemy

$$y_r = g_r(\boldsymbol{z}_r, \hat{\boldsymbol{a}}_r)\,, \tag{4.3}$$

gdzie: $\boldsymbol{z}_r = [\boldsymbol{z}_r^{(1)\mathrm{T}}, \boldsymbol{z}_r^{(2)\mathrm{T}}, ..., \boldsymbol{z}_r^{(I)\mathrm{T}}]^{\mathrm{T}}$ – wektor wejściowy,

$\hat{\boldsymbol{a}}_r = [\tilde{\boldsymbol{a}}_r^{(1)\mathrm{T}}, \tilde{\boldsymbol{a}}_r^{(2)\mathrm{T}}, ..., \tilde{\boldsymbol{a}}_r^{(I)\mathrm{T}}]^{\mathrm{T}}$ – wektor parametrów.

W dalszym ciągu rozważamy przypadek typowych modeli (1.43a), czyli $T_r = k_r v_r^{\alpha_r}$. Przyjmujemy też, że sztuczna sieć neuronowa jest jednowarstwowa i składa się z $R-1$ elementów z funkcjami f_r postaci

$$y_r = \left| \sum_{i=1}^{R-1} (w_{r,i} x_{r,i}) + 1 \right| \stackrel{\Delta}{=} \bar{g}_r(\boldsymbol{x}_r, \boldsymbol{w}_r),\ r = 1, 2, ..., R-1 \tag{4.4}$$

oraz

$$y_R = \left(V - \sum_{r=1}^{R-1} y_r^{-1} \right)^{-1},$$

gdzie

$$x_{r,1} = \frac{k_r}{V\, k_1},\ x_{r,2} = \frac{k_r}{V\, k_2},\ ...,\ x_{r,r-1} = \frac{k_r}{V\, k_{r-1}},$$
$$x_{r,r} = \frac{k_r}{V\, k_{r+1}},\ ...,\ x_{r,R-1} = \frac{k_r}{V\, k_R} \tag{4.5}$$

Każdej z pierwszych $R-1$ operacji odpowiada jeden neuron. Inna postać y_R wynika z konieczności zagwarantowania spełnienia ograniczenia $\sum_{r=1}^{R} v_r = V$. Postaci funkcji (4.4) wynikają z analitycznego rozwiązania problemu rozdziału zadań dla modeli (1.43), czyli $T_r = k_r v_r^{\alpha}$. Algorytm rozdziału (1.55) można przekształcić do postaci

$$v_r^{*-1} = \frac{1}{V} \sum_{s=1}^{R} \left(\frac{k_r}{k_s} \right)^{\frac{1}{\alpha}} . \tag{4.6}$$

Przyjmujemy, że $v_r = y_r^{-1}$, czyli że odwrotności wyjść algorytmu neuropodobnego aproksymują szukane wartości rozmiarów zadań przydzielanych do poszczególnych operacji. W przypadku modeli (1.43) dla $\alpha = 1$ oraz wszystkich wag równych 1 bezpośrednio $v_r^* = y_r^{-1}$. W pozostałych przypadkach należy określać wartości wag $w_{r,k}$ tak, aby przybliżenie było najlepsze.

Nowe wartości wag $w_{r,k}$ są wyznaczane w procesie uczenia na podstawie uzyskiwanych wartości $\boldsymbol{k} = [k_1, k_2, ..., k_R]^{\mathrm{T}}$. Przedstawimy dwie jego wersje.

Wersja 1

Korekcja wag $\boldsymbol{w}_r$ w kolejnych cyklach uczenia (iteracjach) odbywa się na podstawie porównania wartości y_r z (4.4) oraz wartości $y_r^* = v_r^{*-1}$. Wartości y_r^* są wynikiem działania algorytmów η_r' lub odpowiednich procedur numerycznych. Mogą być też określane przez nauczyciela. Ogólna postać algorytmu uczenia jest następująca

$$\boldsymbol{w}_r(n+1) = G_n[\boldsymbol{w}_r(n), \boldsymbol{x}_r(n), y_r(n), y_r^*(n)], \quad n = 0,1,2,..., \tag{4.7}$$

gdzie n jest numerem cyklu uczenia oraz

$$y_r(n) = \bar{g}_r(\boldsymbol{x}_r(n), \boldsymbol{w}_r(n)),$$

$$y_r^*(n) = \eta_r'(\boldsymbol{k}).$$

Istnieje wiele szczegółowych procedur uczenia, np. [110], z których jako przykłady przedstawimy dwie

1. $$\boldsymbol{w}_r(n+1) = \boldsymbol{w}_r(n) + \mu_r(n)\boldsymbol{x}_r(n)[y_r^*(n) - y_r(n)], \tag{4.8}$$

gdzie współczynniki $\mu_r(n)$, takie same dla wszystkich elementów wektora $\boldsymbol{w}_r$, powinny być dobrane tak, aby były spełnione warunki

$$\mu_r(n) > 0, \qquad \sum_{n=0}^{\infty} \mu_r(n) = \infty, \qquad \sum_{n=0}^{\infty} \mu_r^2(n) < \infty. \tag{4.9}$$

2. $$\boldsymbol{w}_r(n+1) = \boldsymbol{w}_r(n) + \mu_r(n)\boldsymbol{d}_r(n), \tag{4.10}$$

gdzie *i*-ty element wektora $\boldsymbol{d}_r(n)$ ma postać

$$d_{r,i}(n) = \frac{Q(\boldsymbol{w}_r(n) + \overline{\boldsymbol{w}}_r(n)) - Q(\boldsymbol{w}_r(n))}{\delta_{r,i}}, \quad i = 1, 2, ..., R-1$$

oraz $Q(\boldsymbol{w}_r(n)) = [y_r^*(n) - y_r(n)]^2$ jest wskaźnikiem jakości, a $\overline{\boldsymbol{w}}_r(n)$ – wektorem o składowych zerowych oprócz elementu *i*-tego równego długości kroku próbnego $\delta_{r,i}$.

W obu algorytmach uczenia wykorzystano pomysł przybliżania $\overline{\eta}_r'$ przez $\overline{g}_r$ w systemie zamkniętym z degresywnym sprzężeniem zwrotnym. Warunki (4.9) zapewniają zbieżność $\boldsymbol{w}_r(n)$ w obu metodach uczenia. Należy jeszcze dodać, że zaproponowany w (4.7) sposób uczenia może być traktowany jako identyfikacja zależności między $y_r^*(n)$ oraz $\boldsymbol{k}(n)$, polegająca na wyznaczaniu parametrów $\boldsymbol{w}_r$. W tym sensie ciąg uczący $([\boldsymbol{k}(1), \boldsymbol{y}^*(1)], [\boldsymbol{k}(2), \boldsymbol{y}^*(2)], ..., [\boldsymbol{k}(j), \boldsymbol{y}^*(j)], ..., [\boldsymbol{k}(n), \boldsymbol{y}^*(n)])$, gdzie $\boldsymbol{y}^*(j) = (y_1^*(j), y_2^*(j), ..., y_{R-1}^*(j))$, $j = 1, 2, ..., n$ może być uznany jako ciąg pomiarów służących do identyfikacji.

Wersja 2

W tej wersji w procesie uczenia zastosowano modele operacji $T_r = \gamma_r'(v_r; \boldsymbol{a}_r)$ lub rzeczywiste obiekty (kompleksy operacji). Samo uczenie może być traktowane jako korekcja w procesie adaptacji parametrów $\boldsymbol{w}_r$ w (4.4). Zmiana parametrów $\boldsymbol{w}_r(n)$ w kolejnych iteracjach jest dokonywana tak, aby minimalizować czas *T* wykonania kompleksu operacji. Można wtedy stosować algorytm uczenia (4.10), gdzie $Q(\boldsymbol{w}_r(n)) = T(n)$, $T(n) = \max\limits_{r=1, 2, ..., R} \{\gamma_r'[v_r(n); \boldsymbol{a}_r(n)]\}$ oraz $v_r(n) = (\overline{g}_r[(\boldsymbol{x}_r(n), \boldsymbol{w}_r(n)])^{-1}$. Wartość czasu $T(n)$ może być w *n*-tym cyklu uczenia mierzona na obiekcie po podaniu wartości $v_r(n)$. Wtedy nie jest potrzebny model obiektu.

W podobny sposób można wykorzystać sztuczną sieć neuronową do rozwiązania problemu rozdziału zasobów. Na przykład dla kompleksu operacji o modelach (1.9), czyli $T_r = \dfrac{1}{k_r u_r^{\alpha_r}}$ można przyjąć takie postaci funkcji f_r, jak w (4.4) z wej-

ściami $x_{r,i}$ określonymi przez (4.5). Wtedy odwrotności wyjść aproksymują szukane wielkości zasobów, czyli $u_r = y_r^{-1}$.

Za pracą [90] omówimy teraz przebieg badań symulacyjnych, dotyczących przedstawionego problemu rozdziału zadań z algorytmem uczenia w wersji 2. oraz zaprezentujemy przykładowe wyniki tych badań. Celem badań jest ocena procesu uczenia sztucznej sieci neuronowej oraz efektu uczenia, czyli jakości rozdziału zadań. Jakość uczenia wiąże się z jego długością, czyli liczbą *N* cykli uczących, po których zmiany parametrów $\boldsymbol{w}_r$ były mniejsze od zadanej wartości $\varepsilon > 0$. Zmiany te są oceniane w warunku stopu

$$\sum_{j=N-m+1}^{N} \xi^{N-j} \Delta_w(j) < \varepsilon , \tag{4.11}$$

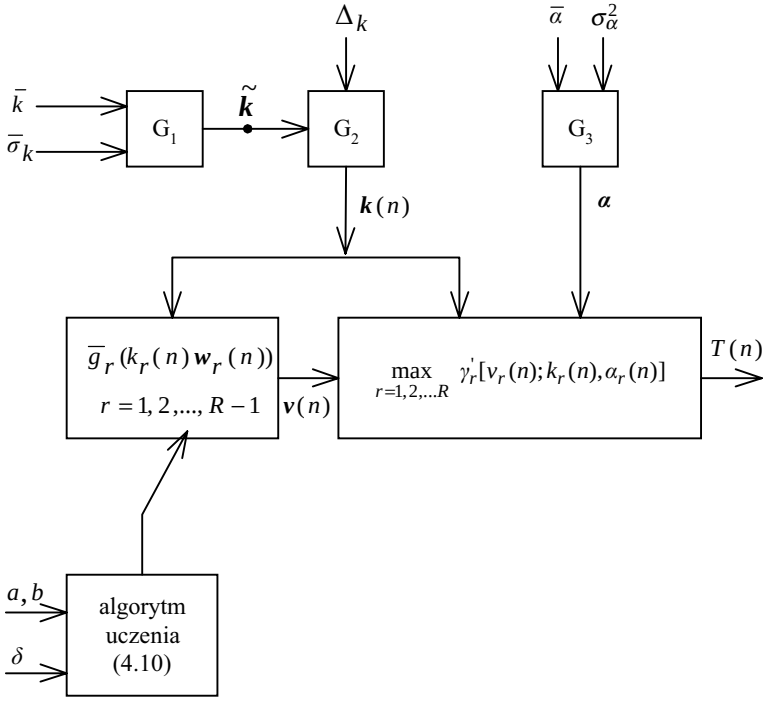
gdzie $\xi \in (0,1]$ oraz $\Delta_w(j)$ są odpowiednio współczynnikiem wagowym oraz średnią zmian parametrów $\boldsymbol{w}_r$ liczoną dla *m* ostatnich cykli uczących, wyrażoną jako

$$\Delta_w(j) = \frac{1}{(R-1)^2} \sum_{r=1}^{R-1} \sum_{i=1}^{R-1} \left| w_{r,i}(j) - w_{r,i}(j-1) \right| . \tag{4.12}$$

Dla odpowiednio dużego *m* i odpowiednio małego ε przyjęty warunek stopu dobrze określa moment, w którym należy przerwać uczenie, bo wagi już praktycznie się nie zmieniają. Badania symulacyjne prowadzono dla m =10, ξ = 0,5 i ε = 0,5. W badaniach przyjęto, że wartości elementów wektora

$$\boldsymbol{k}(n) = [k_1(n), k_2(n), ..., k_{R-1}(n)]^{\mathrm{T}}$$

są generowane niezależnie w sposób losowy według rozkładu równomiernego z przedziału $[\tilde{k}_r - \Delta_k, \tilde{k}_r + \Delta_k]$, gdzie Δ_k jest zadanym maksymalnym rozrzutem wartości wokół $\tilde{k}_r$, będącego środkiem przedziału losowanym również według rozkładu równomiernego dla zadanej wartości średniej $\bar{k}$ i dla zadanej wariancji σ_k^2. W sposób losowy generowane są również parametry $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_{R-1}]^{\mathrm{T}}$. Jest stosowany generator o rozkładzie równomiernym z wartością średnią $\bar{\alpha}$ oraz wariancją σ_α^2. Ponadto założono, że $V = 1$, $w_{r,i}(0) = 1$, r, $i = 1, 2, ..., R-1$, a współczynniki w algorytmie uczenia mają postać $\mu_r(n) = c/n^b$, $c > 0$, $b > 1$ oraz $\delta_{r,i} = \delta$, r, $i = 1, 2, ..., R-1$. Schemat blokowy układu symulacji jest przedstawiony na rys. 4.1, na którym $\boldsymbol{G}_1$, $\boldsymbol{G}_2$ i $\boldsymbol{G}_3$ oznaczają odpowiednio generatory wartości losowych $\tilde{\boldsymbol{k}} = [\tilde{k}_1, \tilde{k}_2, ..., \tilde{k}_{R-1}]^{\mathrm{T}}$, $\boldsymbol{k}(n)$ i $\boldsymbol{\alpha}$. Z wykresów przedstawionych na rys. 4.2 i 4.3 wynika, że zmiany zarówno parametrów obiektu (kompleksu operacji) jak i para-

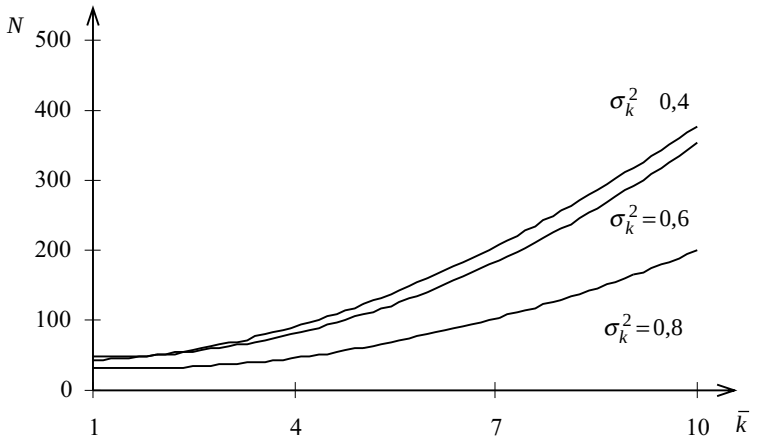


Rys. 4.1. Schemat blokowy układu symulacji

metrów algorytmu uczenia mogą miećwpływ na szybkość uczenia. Szczególnie istotny jest dobór współczynnika $\mu_r(n)$, a dla przyjętej w badaniach postaci c/n^b – wartość wykładnika *b.* Dla mniejszych wartości *b* proces uczenia wydłuża się. Podobną tendencję obserwujemy dla większych wartości parametrów k_r, zwłaszcza gdy wartości te mało różnią się od siebie. Wykres z rys. 4.2 prezentuje wyniki uśrednione, uzyskane dla wielokrotnych generacji losowych parametrów oraz dla różnych wartości $\bar{k}$ z podanego zakresu.

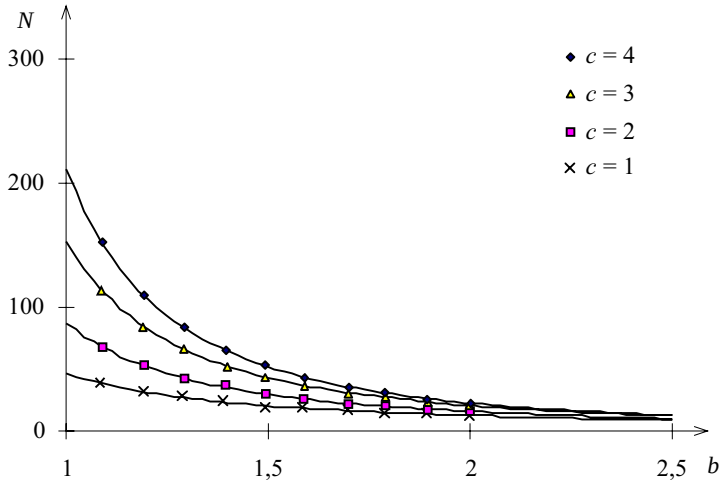
Interesująca jest również zależność między wartościami parametrów modeli operacji a jakością rozdziału zadań dokonanego po zakończeniu procesu uczenia. W celu analitycznej oceny jakości rozdziału można zaproponować następujący empiryczny wskaźnik jakości

$$Q_{\mathrm{U}} = \frac{1}{N_{\mathrm{T}}} \sum_{n=1}^{N_{\mathrm{T}}} e(n) = \frac{1}{N_{\mathrm{T}}} \sum_{n=1}^{N_{\mathrm{T}}} \frac{T(n) - \tilde{T}(n)}{\tilde{T}(n)}, \tag{4.13}$$

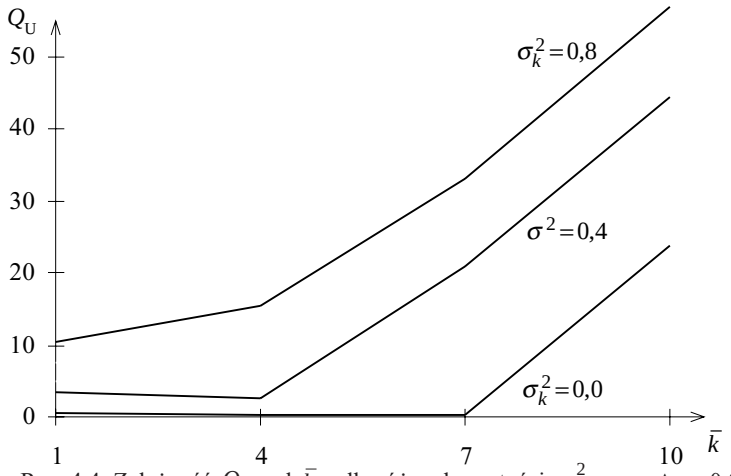


Rys. 4.2. Zależność długości uczenia N od wartości parametru $\bar{k}$, dla różnych wartości σ_k^2 oraz $\Delta_k = 0{,}05$, $c = b = 1$, $\delta = 0{,}01$, $\bar{\alpha} = 1$, $\sigma_\alpha^2 = 0{,}05$

gdzie N_T jest liczbą cykli testujących, natomiast $e(n)$ jest względnym błędem liczonym w n-tym cyklu testowania. Do obliczania błędu $e(n)$ jest wykorzystywane przybliżenie $\tilde{T}(n)$ optymalnej wartości $T^*(n)$ czasu realizacji kompleksu, uzyskiwane za pomocą odpowiedniej procedury numerycznej. Przykładowy wynik przedstawiono na rys. 4.4. Zaprezentowany tam wykres, uzyskany dla założeń i danych



Rys. 4.3. Zależność długości uczenia N od wartości parametru b, dla różnych wartości c oraz $\bar{k} = 2$, $\sigma_k^2 = 0{,}05$, $\Delta_k = 0{,}05$, $\delta = 0{,}01$, $\bar{\alpha} = 1$, $\sigma_\alpha^2 = 0{,}05$



Rys. 4.4. Zależność Q_U od $\bar{k}$, dla różnych wartości σ_k^2, oraz $\Delta_k = 0{,}05$, $c = b = 1$, $\delta = 0{,}01$, $\bar{\alpha} = 1$, $\sigma_\alpha^2 = 0{,}05$

jak w poprzednio omówionych badaniach, jest wynikiem uśrednienia uzyskanego dla wielokrotnych generacji losowych parametrów symulacji. Badania wykazują silną zależność jakości rozdziału zadań po nauczeniu od parametrów modeli operacji (np. rys. 4.4). Można stwierdzić, że jest ona akceptowalna dla wartości $\bar{k} < 7$ oraz dla $\bar{\alpha} \leq 3$. Dla innych parametrów modeli operacji należałoby zaproponować inną postać algorytmu neuropodobnego, a zwłaszcza bardziej złożoną strukturę sztucznej sieci neuronowej.

4.2.2. Uczenie dla modeli z niepewnym parametrem

Wprowadzone w podpunkcie 4.2.1 algorytmy uczenia (4.8) i (4.10) mogą być wykorzystane również i w takiej sytuacji, gdy parametry k_r poszczególnych operacji nie są znane. W przypadku, gdy ich wartości mogą być traktowane jako realizacje zmiennych losowych, wspomniane algorytmy uczenia mają interpretację algorytmów aproksymacji stochastycznej (np. [14]), a warunki (4.9) są wystarczającymi warunkami zbieżności tej procedury. Przypadek nieznajomości parametrów k_r będzie rozpatrywany w niniejszym podpunkcie. Rozważania ograniczymy do rozdziału zadań w kompleksie operacji o modelach liniowych, czyli $T_r = k_r v_r$. Jest to szczególna postać modeli (1.43), w których przyjęto, że $\alpha = 1$.

Załóżmy teraz, że

$$0 \leq k_r \leq \hat{k}_r , \tag{4.14}$$

a wartości $\hat{k}_r$ nie są znane. W konsekwencji oznacza to, że wartości parametrów k_r również nie są znane. Mogą one należeć do przedziału, którego prawy koniec nie jest znany. Na tym polega niepewność wektorowego parametru $\boldsymbol{k} = [k_1, k_2, ..., k_R]^{\mathrm{T}}$, która implikuje niepewność całego kompleksu operacji. Wówczas modele operacji przyjmują postać relacji, określonej w tym przypadku nierównościami

$$T_r \leq \hat{k}_r v_r . \tag{4.15}$$

Wymagania na czas wykonania kompleksu operacji nie możemy teraz formułować w tak „ostry" sposób jak w przypadku deterministycznym. Przyjmujemy, że $T = \max\limits_{r=1,2,...,R} \{T_1, T_2, ..., T_R\} \in \boldsymbol{D}_{\mathrm{T}} = [0, \beta]$, to znaczy

$$T \leq \beta . \tag{4.16}$$

Aby istniało rozwiązanie, β musi być odpowiednio duże. Może być ono na przykład równe minimalnemu czasowi T^* (1.54), czyli

$$T^* = \frac{V}{\sum\limits_{r=1}^{R} (\hat{k}_r)^{-1}}$$

dla $\boldsymbol{k} = \hat{\boldsymbol{k}}$, gdzie $\hat{\boldsymbol{k}} = [\hat{k}_1, \hat{k}_2, ..., \hat{k}_R]^{\mathrm{T}}$. W celu spełnienia wymagania (4.16) wystarczy, aby $T_r \leq \beta$ dla wszystkich r. Decyzje $\boldsymbol{v} = [v_1, v_2, ..., v_R]^{\mathrm{T}}$ należy wyznaczać tak, aby spełnić (4.16) oraz aby $\sum\limits_{r=1}^{R} v_r = V$ i $v_r \geq 0$, $r = 1, 2, ..., R$. Wystarczy w tym celu wybierać $\boldsymbol{v}$ ze zbioru $\hat{\Delta}_v(\hat{\boldsymbol{k}})$ określonego następująco

$$\hat{\Delta}_v(\hat{\boldsymbol{k}}) = \boldsymbol{D}_v(\hat{\boldsymbol{k}}) \cap \overline{\Delta}_{\boldsymbol{v}} , \tag{4.17}$$

gdzie

$$\boldsymbol{D}_v(\hat{\boldsymbol{k}}) = \{\boldsymbol{v} : 0 \leq v_r \leq \frac{\beta}{\hat{k}_r}\} , \tag{4.18}$$

$$\overline{\Delta}_v = \{\boldsymbol{v} : \sum_{r=1}^{R} v_r = V\} . \tag{4.19}$$

Ze względu na postać wymagania (4.16) na czas realizacji kompleksu operacji, rozwiązaniem jest zbiór możliwych wartości $\boldsymbol{v}$. W przypadku deterministycznym, w którym $\hat{\boldsymbol{k}}$ byłby znany, zbiór (4.17) stanowiłby rozwiązanie problemu rozdziału

zadań. W rozważanym przypadku z nieznanym wektorowym parametrem $\hat{\boldsymbol{k}}$ zbiór ten będzie określany w procesie uczenia. W kolejnych cyklach n, $n = 0, 1, ...$ procesu uczenia będzie wyznaczane oszacowanie parametru $\hat{\boldsymbol{k}}$. Proponowany proces uczenia w każdym cyklu polega na sprawdzeniu, czy wybrana decyzja należy do zbioru (4.17) – czynność ta nosi nazwę w a l i d a c j i, a następnie na ewentualnym u a k t u a l n i e n i u tego zbioru. Walidacji i uaktualnieniu podlega w rozważanym przypadku zbiór $\hat{\Delta}_v(\hat{\boldsymbol{k}})$, zawierający wiedzę o podejmowaniu decyzji, a nie o samym obiekcie, którego te decyzje dotyczą. Podstawą uczenia są informacje zawarte w ciągu uczącym ($[\boldsymbol{v}(1), j(1)]$, $[\boldsymbol{v}(2), j(2)]$, ..., $[\boldsymbol{v}(n), j(n)]$). Wielkości $\boldsymbol{v}(i)$, $i = 1, 2, ..., n$ są decyzjami wybieranymi w sposób losowy ze zbioru $\hat{\Delta}_v(\hat{\boldsymbol{k}})$ tak, aby był spełniony warunek (4.19), przy czym początkowa postać tego zbioru, oznaczana jako $\hat{\Delta}_v(\hat{\boldsymbol{k}}(0))$, wynosi

$$\hat{\Delta}_v(\hat{\boldsymbol{k}}(0)) = \boldsymbol{D}_v = \{\boldsymbol{v} : v_r \geq 0,\ r = 1, 2, ..., R,\ \sum_{r=1}^{R} v_r = V\}.$$

Wielkości $j(i),\ i = 1, 2, ..., n$ są podawane przez nauczyciela (eksperta, operatora kompleksu operacji) i mogą przyjmować dwie wartości: 1 – jeśli decyzja $\boldsymbol{v}(i) \in \hat{\Delta}_v(\hat{\boldsymbol{k}})$ oraz 2 – w przeciwnym przypadku. Ze względu na sposób generacji wektorów $\boldsymbol{v}(i)$, uwzględniający wymaganie (4.19), wystarczy sprawdzić przynależność $\boldsymbol{v}(i)$ do zbioru (4.18), tj. $\boldsymbol{D}_v(\hat{\boldsymbol{k}})$, a nie do zbioru (4.17), tj. $\hat{\Delta}_v(\hat{\boldsymbol{k}})$. Wartości $j(i)$ mogą być też uzyskiwane wprost z obiektu. Wówczas równość $j(i) = 1$ oznacza, że jest spełniony warunek (4.16), a sam obiekt podejmowania decyzji wraz z weryfikacją wymagania (4.16) może być potraktowany jako nauczyciel. Wprowadźmy jeszcze następujące rozróżnienie elementów $\boldsymbol{v}(i)$ ciągu uczącego. Niech $\bar{\boldsymbol{v}}(i)$ oraz $\tilde{\boldsymbol{v}}(i)$ będą podciągami ciągu $(\boldsymbol{v}(1), \boldsymbol{v}(2), ..., \boldsymbol{v}(n))$, dla których odpowiednio $j(i) = 1$ oraz $j(i) = 2$. Wtedy można określić zbiory

$$\bar{\boldsymbol{D}}_{\hat{k}}(n) = \{\hat{\boldsymbol{k}} : \bar{\boldsymbol{v}}(i) \in \hat{\Delta}_v(\hat{\boldsymbol{k}}),\ i = 1, 2, ..., n\}, \tag{4.20}$$

$$\tilde{\boldsymbol{D}}_{\hat{k}}(n) = \{\hat{\boldsymbol{k}} : \tilde{\boldsymbol{v}}(i) \in \boldsymbol{D}_v - \hat{\Delta}_v(\hat{\boldsymbol{k}}),\ i = 1, 2, ..., n\}, \tag{4.21}$$

gdzie $\boldsymbol{D}_v - \hat{\Delta}_v(\hat{\boldsymbol{k}})$ jest dopełnieniem zbioru $\hat{\Delta}_v(\hat{\boldsymbol{k}})$. Jako oszacowanie parametru $\hat{\boldsymbol{k}}$ jest proponowany zbiór

$$\Delta_{\hat{k}}(n) = \bar{\boldsymbol{D}}_{\hat{k}}(n) \cap \tilde{\boldsymbol{D}}_{\hat{k}}(n). \tag{4.22}$$

Ze względu na postać zbioru (4.20) elementy wektora $\hat{\boldsymbol{k}}$ mogą być szacowane niezależnie. Wzory (4.20)–(4.22) można zdekomponować i otrzymać

$$\overline{\boldsymbol{D}}_{\hat{k}_r}(n) = \{\hat{k}_{\boldsymbol{r}} : \bar{v}_{\boldsymbol{r}}(i) \le \frac{\beta}{\hat{k}_r},\ i = 1, 2, ..., n\} = [0, \hat{k}_r^{\max}(n)], \tag{4.23}$$

gdzie $\hat{k}_r^{\max}(n) = \dfrac{\beta}{\max\limits_{i=1, 2, ..., n} \{\bar{v}_r(i)\}}$,

$$\widetilde{\boldsymbol{D}}_{\hat{k}_r}(n) = \{\hat{k}_{\boldsymbol{r}} : \tilde{v}_{\boldsymbol{r}}(i) > \frac{\beta}{\hat{k}_r},\ i = 1, 2, ..., n\} = (\hat{k}_r^{\min}(n), \infty), \tag{4.24}$$

gdzie $\hat{k}_r^{\min}(n) = \dfrac{\beta}{\min\limits_{i=1, 2, ..., n} \{\tilde{v}_r(i)\}}$

oraz

$$\Delta_{\hat{k}_r}(n) = \overline{\boldsymbol{D}}_{\hat{k}_r}(n) \cap \widetilde{\boldsymbol{D}}_{\hat{k}_r}(n) = (\hat{k}_r^{\min}(n), \hat{k}_r^{\max}(n)]. \tag{4.25}$$

Algorytm rozdziału zadań z uczeniem dla kompleksu operacji równoległych o modelach (4.15) można przedstawić w zwartej postaci [25].

Bieżący, to znaczy n-ty, $n = 1, 2, ...$, cykl tego algorytmu składa się z sekwencji następujących kroków.

1. Wybierz $\boldsymbol{v}(n)$ w sposób losowy z $\hat{\Delta}_v(\hat{\boldsymbol{k}}(n-1))$, przydziel odpowiednie ilości zadań operacjom oraz wyznacz $T_r(n)$.

2. Powtórz krok 2. dla każdej operacji r, $r = 1, 2, ..., R$.

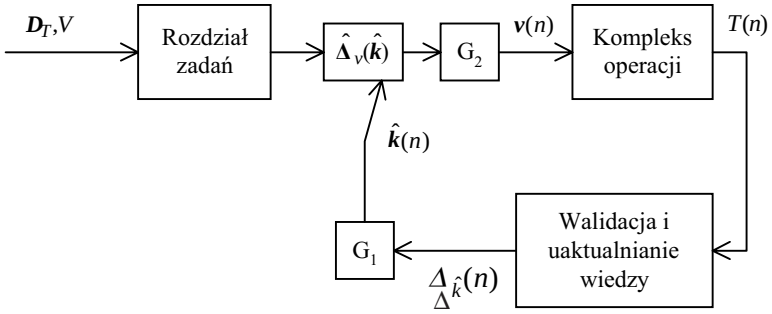
Jeżeli $T_r(n) \le \beta$, to przejdź do kroku 2A. W przeciwnym przypadku przejdź do kroku 2B.

2A. Dokonaj walidacji decyzji $v_r(n) = \bar{v}_r(n)$. W tym celu sprawdź, czy $v_r(n) \le \dfrac{\beta}{\hat{k}_r^{\max}(n-1)}$. Jeśli tak, to podstaw $\hat{k}_r^{\max}(n) = \hat{k}_r^{\max}(n-1)$. W przeciwnym razie dokonaj uaktualnienia decyzji, czyli oblicz $\hat{k}_r^{\max}(n) = \dfrac{\beta}{v_r(n)}$. Następnie podstaw $\hat{k}_r^{\min}(n) = \hat{k}_r^{\min}(n-1)$ i przejdź do kroku 3.

2B. Dokonaj walidacji decyzji $v_r(n) = \tilde{v}_r(n)$. W tym celu sprawdź, czy $v_r(n) > \dfrac{\beta}{\hat{k}_r^{\min}(n-1)}$. Jeśli tak, to podstaw $\hat{k}_r^{\min}(n) = \hat{k}_r^{\min}(n-1)$. W przeciwnym razie dokonaj uaktualnienia decyzji, czyli wyznacz nową wartość $\hat{k}_r^{\min}(n)$ jako $\hat{k}_r^{\min}(n) = \dfrac{\beta}{v_r(n)}$ i następnie podstaw $\hat{k}_r^{\max}(n) = \hat{k}_r^{\max}(n-1)$.

3. Wybierz $\hat{k}_r(n)$ w sposób losowy z przedziału $(\hat{k}_r^{\min}(n), \hat{k}_r^{\max}(n)]$.

Schemat blokowy rozdziału zadań z uczeniem przedstawiono na rys. 4.5. Występujące w nim generatory liczb losowych G_1 i G_2 służą do generacji wartości zmiennych odpowiednio $\hat{\boldsymbol{k}}(n)$ i $\boldsymbol{v}(n)$.



Rys. 4.5. Schemat blokowy algorytmu rozdziału zadań z uczeniem

4.3. Algorytmy ewolucyjne w szeregowaniu zadań z uwzględnieniem ruchu realizatorów

Problematyka szeregowania zadań z uwzględnieniem ruchu realizatorów została wszechstronnie przedstawiona w rozdziale 2. Zaprezentowano tam szczegółowo dokładne i przybliżone algorytmy rozwiązania dla różnych przypadków. Obecnie, nawiązując do tych rozważań, przedstawimy jeszcze jeden sposób rozwiązania, polegający na zastosowaniu algorytmu ewolucyjnego. Celem tych rozważań jest zilustrowanie sposobu wykorzystania tej techniki obliczeniowej do rozwiązywania specyficznego zagadnienia optymalizacji dyskretnej oraz porównanie uzyskanych wyników z rezultatami otrzymanymi przy zastosowaniu klasycznych algorytmów optymalizacji. Skoncentrujemy się na problemie minimalizacji długości uszerego-

wania i nawiążemy do pracy [71]. Na wstępie zauważmy, że algorytmy ewolucyjne można w różny sposób i w różnym stopniu stosować do rozwiązywania tego zagadnienia optymalizacyjnego. Można ich użyć bezpośrednio do rozwiązania problemu wyjściowego PC, ale można również wykorzystać dekompozycję funkcjonalną i zastosować algorytmy ewolucyjne do rozwiązania klasycznego problemu szeregowania i (lub) komiwojażera z 2. i 3. kroku algorytmu rozwiązania, uzyskanego po tej dekompozycji.

Rozpatrzmy zastosowanie algorytmu genetycznego do rozwiązania problemu PC, czyli do minimalizacji długości uszeregowania (2.9) względem macierzy γ zdefiniowanej w (2.2) dla ograniczeń danych zależnościami (2.3)–(2.8). Stosujemy klasyczny schemat algorytmu genetycznego, w którym po generacji populacji początkowej są stosowane operatory selekcji, krzyżowania i mutacji. Funkcją przystosowania jest bezpośrednio minimalizowana funkcja celu (2.9), oznaczana dalej jako $Q \stackrel{\Delta}{=} Q_{\mathrm{C}}(\gamma)$. Proponujemy następujący sposób kodowania. Osobnik, czyli chromosom, a w zagadnieniu optymalizacji zmienna optymalizacyjna, jest ciągiem genów $(g_1, g_2, .., g_M)$ o długości $M = H + R$. Ciąg ten jest podzielony na dwie części. Pierwsza składa się z H, a druga z R elementów. W drugiej części zapisano liczby zadań wykonywanych przez realizatory r, $r = 1, 2, ..., R$, czyli $g_{H+r} = M_r$. W pierwszej części kolejno dla poszczególnych realizatorów są zakodowane trasy przejazdów realizatorów, w postaci numerów zadań wykonywanych przez realizatory. Bieżący element $m_r(j)$, $j = 1, 2, ..., M_r$ trasy $\boldsymbol{M}_r$ r-tego realizatora, będący numerem odpowiedniego stanowiska (zadania) jest zapisany jako element g_m, gdzie $m = j + \sum_{i=0}^{r-1} M_i$ oraz $M_0 = 0$. Na przykład dla $H = 20$ i $R = 4$ jeden z chromosomów może mieć postać (4, 12, 1, 9, 2, 19, 20, 11, 3, 5, 8, 10, 18, 17, 15, 16, 13, 6, 7, 14, 3, 7, 4, 6) i wtedy liczby M_r oraz trasy $\boldsymbol{M}_r$ dla poszczególnych realizatorów są następujące

$M_1 = 3, \quad \boldsymbol{M}_1 = (21, 4, 12, 1, 21),$
$M_2 = 7, \quad \boldsymbol{M}_2 = (21, 9, 2, 19, 20, 11, 3, 5, 21),$
$M_3 = 4, \quad \boldsymbol{M}_3 = (21, 8, 10, 18, 17, 21),$
$M_4 = 6, \quad \boldsymbol{M}_4 = (21, 15, 16, 13, 6, 7, 14, 21).$

Trasy $\boldsymbol{M}_r$ zawierają ciągi zadań wykonywanych przez realizatory uzupełnione przez bazę $H + 1 = 21$, stanowiącą początek i koniec każdej trasy. Efektywność algorytmów ewolucyjnych polegająca na osiąganiu ekstremów globalnych funkcji w dużym stopniu zależy od wartości parametrów tych algorytmów, takich jak: liczność populacji I, liczba iteracji (liczba pokoleń) J, a przede wszystkim prawdopo-

dobieństwo krzyżowania p_1 i prawdopodobieństwo mutacji p_2. W większości zastosowań wartości tych parametrów są ustalane przed uruchomieniem algorytmu i często na podstawie intuicyjnych przesłanek lub analizy podobnych przypadków. Można jednak zaproponować bardziej racjonalne sposoby określania wartości tych parametrów. Wskażemy na dwa z nich.

1. Prawdopodobieństwa te zmieniają się w trakcie działania algorytmu genetycznego, to znaczy jest dokonywana ich adaptacja, np. [106].

2. Wartości prawdopodobieństw są wynikiem uczenia.

W pierwszym przypadku wartości tych prawdopodobieństw w bieżącej, czyli *j*-tej iteracji (populacji) algorytmu genetycznego, gdzie $j = 1, 2, ..., J$, są ustalane na podstawie wartości funkcji przystosowania osobników w bieżącej i poprzednich iteracjach. Wprowadźmy następujące oznaczenia:

$\boldsymbol{I}_j$ – *j*-ta populacja algorytmu genetycznego,

i, $i = 1, 2, ..., I$ – numer osobnika w populacji o liczności I (przyjmujemy, że wszystkie populacje są tak samo liczne),

$Q_j^{(i)}$ – wartość kryterium (funkcji przystosowania) dla *i*-tego osobnika w *j*-tej iteracji,

$\underline{Q}_j, \overline{Q}_j, \tilde{Q}_j$ – odpowiednio minimalna, maksymalna i średnia wartość kryterium w *j*-tej iteracji.

Wtedy po uwzględnieniu wyników tylko z dwóch sąsiednich iteracji j – 1 i *j*-tej, prawdopodobieństwa $p_{1,j}$ i $p_{2,j}$ w iteracji bieżącej, to znaczy *j*-tej można wyznaczyć jako

$$p_{l,j} = \Theta(Q_j^{(1)}, Q_j^{(2)}, ..., Q_j^{(I)}, Q_{j-1}^{(1)}, Q_{j-1}^{(2)}, ..., Q_{j-1}^{(I)}),\ l = 1, 2,\ \ j = 2, 3, ..., J\ . \quad (4.26)$$

Omówimy teraz kilka wersji odwzorowania Θ.

Wersja A

W wersji tej, zgodnie z powszechną interpretacją prawdopodobieństw krzyżowania i mutacji, przyjęto, że prawdopodobieństwo krzyżowania musi być tym większe, im gorsza jest populacja i tym mniejsze, im populacja jest lepsza. Jakość populacji jest tu rozumiana w sensie średniej wartości kryterium dla całej populacji $\tilde{Q}_j$. Dlatego w celu odpowiedniego ukształtowania prawdopodobieństwa krzyżowania jest rozpatrywany iloraz $\tilde{Q}_j / \hat{Q}_j$, gdzie $\hat{Q}_j = \max\limits_{1 \le m \le j} \{\tilde{Q}_m\}$ jest największą średnią wartością kryterium uzyskaną do iteracji *j* włącznie. Iloraz ten może przyjmować wartości z przedziału $(0, 1]$. Wartość jeden oznacza, że populacja składa się z jednakowych osobników, czyli jest zdegenerowana. Wówczas prawdopodobieństwo

krzyżowania powinno wynieść jeden, a więc krzyżowaniu należy poddać wszystkie osobniki. Zmniejszanie się wartości ilorazu oznacza wzrost jakości populacji. Wtedy prawdopodobieństwo krzyżowania może być mniejsze.

Prawdopodobieństwo mutacji powinno być związane ze stopniem ujednolicenia populacji. Miarą ujednolicenia populacji może być stosunek $Q_j^* / \widetilde{Q}_j$, gdzie Q_j^* jest najmniejszą wartością kryterium uzyskaną do iteracji j włącznie. W miarę działania algorytmu ewolucyjnego stosunek ten rośnie, czyli osobniki w populacji „upodobniają się" do osobnika dotychczas najlepszego (o wartości kryterium Q_j^*). Aby zapobiec temu zjawisku, prawdopodobieństwo mutacji musi być wtedy stosunkowo duże. Do konstrukcji obu prawdopodobieństw stosujemy funkcję cosinus i proponujemy obliczanie ich dla kolejnych iteracji w następujący sposób

$$p_{1,j}^{\mathrm{A}} = 1 - \cos\left(\frac{\pi}{2}\frac{\widetilde{Q}_j}{\hat{Q}_j}\right), \tag{4.27}$$

$$p_{2,j}^{\mathrm{A}} = \rho_2^{\mathrm{A}}\left[1 - \cos\left(\frac{\pi}{2}\frac{Q_j^*}{\widetilde{Q}_j}\right)\right], \tag{4.28}$$

gdzie $\rho_2^{\mathrm{A}} \in [0,1]$ jest danym współczynnikiem. Prawdopodobieństwo $p_{1,j}^{\mathrm{A}}$ przyjmuje wartości z przedziału (0, 1], a prawdopodobieństwo $p_{2,j}^{\mathrm{A}}$ – z przedziału $(0, \rho_2^{\mathrm{A}})$. Współczynnik ρ_2^{A} umożliwia ustalanie małych wartości prawdopodobieństw mutacji.

Wersja B

W tej wersji proponujemy tylko inny sposób określania prawdopodobieństw krzyżowania. Krzyżowanie jest tu połączone z selekcją. Prawdopodobieństwa krzyżowania są określone dla każdej pary osobników w populacji, czyli dla $I(I-1)/2$ przypadków. Najpierw jest tworzony ciąg takich par według nierosnących prawdopodobieństw krzyżowania, obliczanych według zależności

$$p_{1,j}^{\mathrm{B}} = \begin{cases} \rho_1^{\mathrm{B}} \dfrac{Q_j' - \underline{Q}_j}{\widetilde{Q}_j - \underline{Q}_j}, & \text{dla } Q_j' \le \widetilde{Q}_j \\ \overline{\rho}_1^{\mathrm{B}}, & \text{dla } Q_j' > \widetilde{Q}_j, \end{cases} \tag{4.29}$$

gdzie Q_j' jest wartością kryterium dla rodziców $i_1, i_2 \in \boldsymbol{I}_j$, obliczoną według jednego z wzorów

$$Q'_j = \min\{Q_j^{(i_1)}, Q_j^{(i_2)}\}, \tag{4.30}$$

$$Q'_j = 0{,}5\,(Q_j^{(i_1)} + Q_j^{(i_2)}) \tag{4.30a}$$

oraz $\rho_1^{\mathrm{B}}, \overline{\rho}_1^{\mathrm{B}} \in (0, 1]$ są danymi współczynnikami. Złożona postać wyrażenia (4.29) wynika z konieczności zachowania warunku $0 \le p_{1,j}^{\mathrm{B}} \le 1$. Następnie $[I/2]$ pierwszych par z ciągu podlega krzyżowaniu. Osobniki będące rezultatem krzyżowania tworzą nową populację o liczności I. Symbol $[\cdot]$ oznacza część całkowitą liczby. W przypadku, gdy I jest liczbą nieparzystą, krzyżowaniu podlega dodatkowo kolejna para z ciągu i tylko jeden, losowo wybrany osobnik potomny, uzupełnia nową populację. Prawdopodobieństwo mutacji można: wyznaczać według wzoru (4.28), przyjąć, że ma stałą wartość lub obliczać tak jak w wersji C.

Wersja C

Kolejny sposób adaptacyjnego ustalania obu prawdopodobieństw został zaczerpnięty z pracy [106]. Wzór na prawdopodobieństwo krzyżowania jest taki sam jak w wersji B, czyli (4.29). Inny jest jednak sposób jego wykorzystania. Teraz selekcja jest wyodrębniona tak jak w klasycznych algorytmach genetycznych lub ewolucyjnych. Porządek w populacji ustala się w dowolny sposób i w rezultacie otrzymuje się ciąg osobników. Krzyżowaniu podlegają kolejno dwa sąsiednie osobniki, począwszy od pierwszego. Po obliczeniu prawdopodobieństwa krzyżowania według (4.29), gdzie teraz i_1 oraz i_2 oznaczają dwóch sąsiednich osobników rodzicielskich, jest ono porównywane z losowo wygenerowaną liczbą z przedziału [0, 1]. Do krzyżowania dochodzi, gdy obliczone prawdopodobieństwo jest większe od wygenerowanej liczby. Wówczas, tradycyjnie, osobniki rodzicielskie są w nowej populacji zastępowane przez osobniki potomne. Prawdopodobieństwa mutacji są wyznaczane w tej wersji według wzoru

$$p_{1,j}^{\mathrm{B}} = \begin{cases} \rho_1^{\mathrm{B}} \cdot \dfrac{Q'_j - \underline{Q}_j}{\widetilde{Q}_j - \underline{Q}_j}, & \text{dla } Q'_j \le \widetilde{Q}_j \\ \overline{\rho}_1^{\mathrm{B}}, & \text{dla } Q'_j > \widetilde{Q}_j, \end{cases} \tag{4.31}$$

gdzie $\rho_2^{\mathrm{C}}, \overline{\rho}_2^{\mathrm{C}} \in (0,1]$ są danymi współczynnikami. Warunek przeprowadzenia mutacji osobnika $i \in \boldsymbol{I}_j$ jest podobny jak w przypadku krzyżowania. Oznacza to, że obliczona wartość $p_{2,j}^{\mathrm{C}}$ jest porównywana z wygenerowaną losowo liczbą z przedziału [0, 1] i jeśli jest większa od niej, to dochodzi do mutacji osobnika. Ta wersja za-

pewnia, że do następnej populacji są przenoszone osobniki o najlepszych wartościach kryterium. Pozostałe osobniki uczestniczą w poszukiwaniu rozwiązań w nowych obszarach zbioru rozwiązań dopuszczalnych.

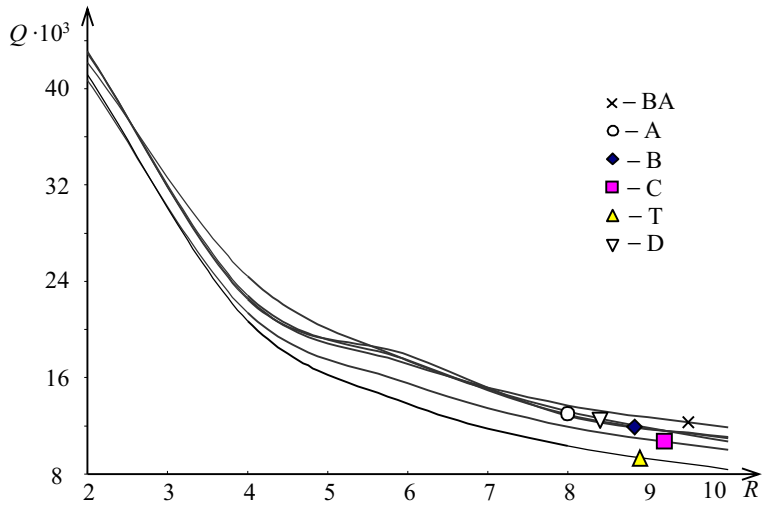
Wersja D

Prawdopodobieństwa krzyżowania są teraz uzależnione od wartości kryterium dla dwóch sąsiednich iteracji i wynoszą

$$p_{1,j}^{\mathrm{D}} = \begin{cases} \rho_1^{\mathrm{D}} \dfrac{(\tilde{Q}_j - \underline{Q}_j) - (\tilde{Q}_{j-1} - \underline{Q}_{j-1})}{\max\{\overline{Q}_j, \overline{Q}_{j-1}\} - \min\{\underline{Q}_j, \underline{Q}_{j-1}\}}, & j = 2,3,\ldots,J \\ 1, & j = 1 \end{cases} \tag{4.32}$$

gdzie $\rho_1^{\mathrm{D}} \in (0,1]$ jest danym współczynnikiem. Wartości prawdopodobieństw krzyżowania zależą tu od poprawy średniej oraz najlepszej wartości kryterium dla dwóch sąsiednich iteracji. Nie proponujemy nowego sposobu obliczania prawdopodobieństw mutacji. Można skorzystać z zależności (4.28) lub (4.31) albo przyjąć stałą wartość tego prawdopodobieństwa.

Na rysunku 4.6 przedstawiono przykładowe wyniki uruchomienia algorytmu genetycznego w wersji bez adaptacji (BA), to znaczy z a priori przyjętymi prawdopodobieństwami p_1 i p_2 oraz we wszystkich omówionych wersjach z adaptacją (A, B, C, D). Dla porównania na wykresie zaznaczono też wyniki uzyskane przez algo-



Rys. 4.6. Zależność Q od R dla $H = 250$

rytm przybliżony (T), omówiony w podpunkcie 2.3.1, w którym nie stosowano podejścia ewolucyjnego. Wykres przedstawia zależność długości uszeregowania Q od liczby realizatorów R dla $H = 250$ zadań.

Analiza wyników przedstawionych na wykresie (rys. 4.6) poparta wnioskami z innych badań pozwala na sformułowanie dwóch uwag. Algorytmy ewolucyjne, bez względu na stosowaną wersję nie zawsze dają wynik lepszy niż algorytm tradycyjny. Zaletą algorytmów ewolucyjnych jest w tym kontekście ich uniwersalizm i brak konieczności opracowywania nowych wersji dla różnych nawet znaczących modyfikacji wyjściowego problemu optymalizacji. Druga uwaga dotyczy celowości stosowania adaptacji w ustalaniu prawdopodobieństw krzyżowania i mutacji. Jej wprowadzenie, zwłaszcza w wersji C, znacznie zwiększyło jakość rozwiązania w stosunku do przypadku bez adaptacji (BA). Więcej wyników badań przedstawiono w [71].

Innym sposobem wyznaczania obu prawdopodobieństw lub tylko jednego z nich jest zastosowanie uczenia. Punktem wyjścia dla uzasadnienia celowości stosowania uczenia w rozważanym przypadku jest własność polegająca na tym, że wartości p_1 i p_2, będące parametrami algorytmu ewolucyjnego, mają z jednej strony wpływ na jakość uzyskiwanego wyniku optymalizacji, ale z drugiej strony zależą od danych problemu optymalizacyjnego i (lub) innych parametrów algorytmu ewolucyjnego. W przypadku szeregowania z ruchomymi realizatorami danymi tymi są przede wszystkim czasy dojazdów realizatorów do stanowisk, ale także czasy wykonywania czynności na stanowiskach. Dla klasycznego problemu szeregowania są to czasy wykonania zadań. Parametrem algorytmu ewolucyjnego, mającym istotny wpływ na wartości p_1 i p_2 jest wielkość populacji I. Podamy teraz prosty przykład wyznaczania prawdopodobieństw krzyżowania i mutacji w procesie uczenia w wersji bez nauczyciela. W kolejnych cyklach uczenia n, $n = 0,1,...$ wartości p_l, $l = 1,2$ są modyfikowane w sposób rekurencyjny na podstawie wartości kryterium jakości szeregowania, oceniającego kolejne zmiany p_1 i p_2. W każdym cyklu uczenia n jest uruchamiany algorytm ewolucyjny w celu wyznaczania wartości kryterium, ale za każdym razem dla nowych, generowanych losowo i niezależnych od wartości z poprzednich iteracji, danych problemu optymalizacyjnego i (lub) parametrów algorytmu ewolucyjnego. Można wykorzystać opisany już w punkcie 4.2 algorytm uczenia (4.10) lub różne jego modyfikacje, np. [91], a także algorytmy neuropodobne. Algorytm (4.10) ma teraz następującą postać

$$p_l(n+1) = p_l(n) - \mu_l(n)\, d_l(n), \tag{4.33}$$

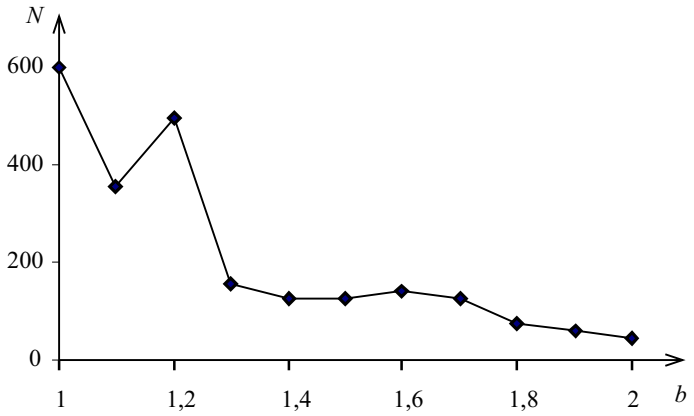
gdzie $\mu_l(n) = c/n^b$ jest zmiennym współczynnikiem o własnościach podanych w zależności (4.9),

$$d_l(n) = \frac{Q''(p_l(n) + \delta) - Q''(p_l(n))}{\delta}, \tag{4.34}$$

gdzie δ jest stałą długością kroku próbnego oraz Q'' jest unormowaną wartością kryterium liczoną według zależności $Q_j''^{(i)} \triangleq Q_j^{(i)} / \overline{Q}_j$. Unormowana postać kryterium umożliwia ograniczenie wartości licznika w zależności (4.34).

Wykres na rys. 4.7 jest ilustracją wpływu doboru współczynników $\mu_l(n)$ na proces uczenia, a dokładniej na jego długość *N*. Zastosowanie algorytmu uczenia (4.33) pozwoliło na niewielkie, bo ok. 2% zwiększenie jakości rozwiązania w stosunku do najlepszej wersji algorytmu z adaptacją, czyli wersji C (rys. 4.6).

Zakończmy rozważania w tym punkcie uwagą, że wspomniany już uniwersalizm algorytmów ewolucyjnych, pozwala na łatwe wykorzystanie opisanego algorytmu w innych problemach szeregowania, w tym również tych, które zostały przedstawione w podpunkcie 2.4.3. Dotyczy to między innymi szeregowania w warunkach niepewności z czasami dojazdu należącymi do zadanych przedziałów lub skończonych zbiorów wartości [73]. W tym przypadku, ze względu na minimaksową postać kryterium jakości szeregowania, algorytm ewolucyjny o przedstawionej konstrukcji, został zastosowany dwukrotnie: do minimalizacji względem macierzy określających algorytmy szeregowania i do maksymalizacji ze względu na możliwe realizacje czasów wykonywania zadań.



Rys. 4.7. Zależność długości uczenia *N* od parametru *b*

4.4. Rozpoznawanie z reprezentacją wiedzy w sterowaniu jazdą

W podpunkcie 3.2.1 omówiliśmy problem sterowania jazdą grupy realizatorów. Zaproponowaliśmy tam hierarchiczną strukturę systemu sterowania z wyodrębnionymi algorytmami sterowania jazdą pojedynczych realizatorów oraz z nadrzędnymi algorytmami koordynacji jazdy, wprowadzonymi w celu zapewnienia unikania ewentualnych kolizji. Istota sterowania w takim systemie polega na wyznaczaniu przez koordynator tak zwanych obszarów dopuszczalnych $\boldsymbol{Y}_{r,v}$ i przekazywaniu ich do lokalnych algorytmów dolnego poziomu (rys. 3.5). Tam, z uwzględnieniem postaci tych obszarów, następowało wyznaczanie decyzji sterujących niezależnie dla poszczególnych realizatorów. Można powiedzieć, że decyzje o ruchu realizatorów były podejmowane niezależnie przez lokalne algorytmy sterowania, a algorytm nadrzędny, zgodnie ze swoją nazwą, umożliwiał tylko koordynację ich jazdy. Jak wykazały badania symulacyjne, ten sposób sterowania nie zapewniał zadowalającej jakości sterowania we wszystkich przypadkach. Występowały sytuacje, gdy następowało wzajemne blokowanie jazdy przez realizatory uczestniczące w ruchu. Dochodziło do tego między innymi w takich sytuacjach, gdy realizatory jechały naprzeciw siebie, a tory ich jazdy pokrywały się. Było to spowodowane niedokładnością modeli opisujących warunki jazdy bezkolizyjnej, a dokładniej nie uwzględnianiem w nich wszystkich sytuacji, w których mogły się znaleźć poruszające się realizatory. Próby poprawy przedstawionych w podpunkcie 3.2.1 tradycyjnych algorytmów sterowania jazdą z unikaniem kolizji również nie dawały zadowalających rezultatów. W związku z tym została opracowana nowa metoda sterowania, którą teraz zaprezentujemy, opierając się na pracach [67, 68, 70, 78]. Zmiana dotyczy tylko sposobu zapewnienia bezkolizyjności jazdy. Algorytmy sterowania jazdą pojedynczych realizatorów pozostają bez zmian.

Proponowana metoda różni się od poprzedniej dwiema kwestiami. Inny jest sposób zapisu informacji niezbędnych do wyznaczania warunków bezkolizyjnej jazdy. Zastosowano w tym celu wyrażenia logiczne. Ponadto inna jest rola algorytmu na poziomie górnym. Nie jest to teraz tylko koordynator jazdy, ale algorytm, w którym są podejmowane decyzje, dotyczące dalszej jazdy wszystkich realizatorów. Tak więc decyzje o możliwości jazdy bezkolizyjnej są podejmowane na poziomie górnym, a nie dolnym systemu sterowania. Wspomniane podejmowanie decyzji zostało sprowadzone do procedury rozpoznawania, a ze względu na postać wykorzystywanych informacji jest to rozpoznawanie z reprezentacją wiedzy [21]. Reprezentacja wiedzy jest w tym przypadku niekonwencjonalnym opisem zależności między wekto-

rem cech i numerem klas. Podstawą do jej określenia są tak zwane formuły elementarne, czyli elementarne własności dotyczące cech oraz klas, a także ewentualnie wielkości dodatkowych. W rozpatrywanym przypadku formuły elementarne dotyczące cech są budowane na podstawie informacji zawartych w wektorach stanu $\boldsymbol{x}_{r,v}$. Oznaczmy przez $\boldsymbol{\alpha}_x^r = (\alpha_{x,1}^r, \alpha_{x,2}^r, ..., \alpha_{x,g_r}^r)$, g_r – elementowy ciąg formuł elementarnych dla realizatora r, wyznaczany na podstawie wektora $\boldsymbol{x}_{r,v}$.. W zapisie ciągu $\boldsymbol{\alpha}_x^r$ brak jest indeksu czasu v, ponieważ rozpoznawanie będzie prowadzone niezależnie dla poszczególnych momentów czasu v. Zasada ta będzie utrzymana również w oznaczeniach innych wielkości związanych z omawianą procedurą rozpoznawania. W rozpoznawaniu są wykorzystywane również formuły elementarne zawierające informacje o parze realizatorów r i s. Oznaczmy ciąg takich formuł przez $\boldsymbol{\alpha}_x^{r,s} = (\alpha_{x,1}^{r,s}, \alpha_{x,2}^{r,s}, ..., \alpha_{x,g_{r,s}}^{r,s})$. Podobnie jak w przypadku $\alpha_{x,i}^r$, $i = 1, 2, ..., g_r$ są one ustalane na podstawie wektorów stanu $\boldsymbol{x}_{r,v}$. i $\boldsymbol{x}_{s,v}$.. Formuły elementarne dla wszystkich realizatorów oraz dla wszystkich par realizatorów tworzą ciąg $\boldsymbol{\alpha}_x = (\boldsymbol{\alpha}_x^r, \boldsymbol{\alpha}_x^{r,s})$, $r, s = 1, 2, ..., R$, $r < s$. Formuły elementarne wprowadzamy również dla klas. Rozpoznawanie polega na określaniu sytuacji, w której znajduje się każdy pojazd i która ma wpływ na jego dalszy ruch. Dla każdego pojazdu wyróżniamy cztery takie sytuacje. Nazywamy je dalej klasami i oznaczamy przez j_r. Tak więc wynikiem rozpoznawania dla każdego pojazdu jest jedna z czterech klas, to znaczy:

$j_r = 1$ – pojazd r może jechać bez konieczności omijania przeszkód,

$j_r = 2$ – pojazd r może jechać z omijaniem przeszkód,

$j_r = 3$ – pojazd r zakończył jazdę,

$j_r = 4$ – pojazd r musi się zatrzymać w celu uniknięcia kolizji.

Wynikiem rozpoznawania dla wszystkich pojazdów jest ciąg $j = (j_1, j_2, ..., j_R)$. Zbiór wszystkich klas oznaczamy jako $\boldsymbol{J} = \{1, 2, ..., J\}$, gdzie $J = 4^R$ jest liczbą klas. Numer klasy $j \in \boldsymbol{J}$ można określić na podstawie elementów ciągu $(j_1, j_2, ..., j_R)$ jako $j = 1 + \sum_{r=1}^{R} 4^{r-1}(j_r - 1)$. Teraz możemy wprowadzić formuły elementarne dotyczące klas. Mają one ogólną postać $\alpha_{j,i}^r =$" $j_r = i$", $i = 1, 2, 3, 4$, co oznacza, że wynikiem rozpoznawania sytuacji pojazdu r jest klasa o numerze i. Podobnie jak w przypadku cech można wprowadzić ciągi formuł elementarnych $\boldsymbol{\alpha}_j^r = (\alpha_{j,1}^r, \alpha_{j,2}^r, \alpha_{j,3}^r, \alpha_{j,4}^r)$ oraz $\boldsymbol{\alpha}_{\mathrm{D}} = (\boldsymbol{\alpha}_j^1, \boldsymbol{\alpha}_j^2, ..., \boldsymbol{\alpha}_j^R)$. Ciągi $\boldsymbol{\alpha}_x$ i $\boldsymbol{\alpha}_{\mathrm{D}}$ są podstawą tworzenia tak zwanych faktów $F_i(\boldsymbol{\alpha}_x, \boldsymbol{\alpha}_{\mathrm{D}})$, $i = 1, 2, ..., I$. Fakty są wyrażeniami logicznymi w dwuwartościowej algebrze Boole'a, utworzonymi z podciągów formuł elementarnych $\boldsymbol{\alpha}_x$ i $\boldsymbol{\alpha}_{\mathrm{D}}$ za pomocą następujących podstawowych operatorów (funktorów) logicznych: koniunkcji ($\wedge$), alternatywy ($\vee$), negacji ($\neg$), implikacji ($\rightarrow$).

Przyjmujemy założenie o prawdziwości wszystkich faktów i reprezentację wiedzy określamy jako ich iloczyn logiczny, czyli $F = F_1 \wedge F_2 \wedge ... \wedge F_I$. Wtedy problem rozpoznawania możemy ogólnie sformułować w następujący sposób.

Dla danych: reprezentacji wiedzy $F(\boldsymbol{\alpha}_x, \boldsymbol{\alpha}_\mathrm{D})$ oraz „pomiarów cech" w postaci prawdziwej formuły logicznej (własności wejściowej) $F_x(\boldsymbol{\alpha}_x)$ należy wyznaczyć najmniejszy zbiór $\boldsymbol{D}_j \subseteq \boldsymbol{J}$ taki, że jest spełniona implikacja

$$F_x(\boldsymbol{\alpha}_x) \wedge F(\boldsymbol{\alpha}_x, \boldsymbol{\alpha}_\mathrm{D}) \rightarrow j \in \boldsymbol{D}_j . \tag{4.35}$$

Warto zauważyć, że własność wejściowa $F_x(\boldsymbol{\alpha}_x)$ jest związana z podzbiorem $\boldsymbol{D}_x$ przestrzeni cech $\boldsymbol{X}$ zależnością

$$\boldsymbol{D}_x = \{\boldsymbol{x} \in \boldsymbol{X} : F_x(\boldsymbol{\alpha}_x) = 1\} .$$

Do rozwiązania tego problemu, a także innych bardziej ogólnych zagadnień analizy, jak również podejmowania decyzji opracowano szczegółowe algorytmy rozwiązania. Opierają się one na tak zwanej metodzie logiczno-algebraicznej, np. [19, 22, 65]. Podstawowa idea tej metody polega na zastąpieniu indywidualnie dla różnych przypadków opracowywanych koncepcji rozumowania opartych na regułach wnioskowania – przez uniwersalne metody algebraiczne oparte na regułach algebry logiki dwuwartościowej. Otrzymane w ten sposób algorytmy można traktować jako ujednolicenie i uogólnienie różnych indywidualnych procedur rozumowania dla szerokiej klasy systemów z logiczną reprezentacją wiedzy. Do rozwiązania zadania analizy nie jest wymagana szczególna postać reprezentacji wiedzy (np. zestaw tak zwanych reguł w postaci implikacji, dla których stosuje się znane algorytmy wnioskowania). Co więcej, zadanie odwrotne, czyli podejmowanie decyzji nie przedstawia tu dodatkowych pryncypialnych trudności.

Istotnym składnikiem metody logiczno-algebraicznej było opracowanie oryginalnej metody dekompozycji i opartych na niej procedur rekurencyjnych oraz dwóch interpretacji: interpretacji dedukcyjnej nawiązującej do algorytmizacji wnioskowania i interpretacji systemowej lokującej całą problematykę w dobrze znanych formalizmach i metodach teorii systemów.

Po wykorzystaniu metody logiczno-algebraicznej, rozwiązanie sformułowanego problemu rozpoznawania, który jest szczególnym przypadkiem ogólnego problemu analizy, można sprowadzić do rozwiązania równania algebraicznego w algebrze logiki dwuwartościowej

$$F_x(\boldsymbol{a}_x) \wedge F(\boldsymbol{a}_x, \boldsymbol{a}_\mathrm{D}) = 1$$

względem $\boldsymbol{a}_{\mathrm{D}}$, gdzie $\boldsymbol{a}_x$ i $\boldsymbol{a}_{\mathrm{D}}$ są zero-jedynkowymi ciągami wartości logicznych elementów ciągów $\boldsymbol{\alpha}_x$ i $\boldsymbol{\alpha}_{\mathrm{D}}$. Rozwiązanie tego równania sprowadza się do wyznaczenia zbioru

$$\bar{\boldsymbol{S}}_{\mathrm{D}} = \left\{ \boldsymbol{a}_{\mathrm{D}} \in \boldsymbol{S}_{\mathrm{D}} : \bigvee_{\boldsymbol{a}_x \in \boldsymbol{S}_x} F_x(\boldsymbol{a}_x) \wedge F(\boldsymbol{a}_x, \boldsymbol{a}_{\mathrm{D}}) = 1 \right\}, \tag{4.36}$$

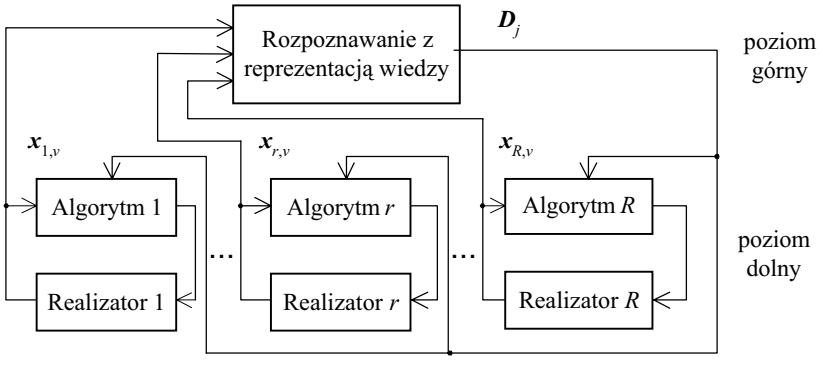
gdzie $\boldsymbol{S}_x$ i $\boldsymbol{S}_{\mathrm{D}}$ są zbiorami wszystkich $\boldsymbol{a}_x$ i $\boldsymbol{a}_{\mathrm{D}}$. Wtedy rozwiązaniem jest zbiór klas

$$\boldsymbol{D}_j = \left\{ j \in \boldsymbol{J} : \bigvee_{\boldsymbol{a}_{\mathrm{D}} \in \bar{\boldsymbol{S}}_{\mathrm{D}}} j = 1 + \sum_{r=1}^{R} \left(\sum_{i=1}^{4} i\, a_{j,i}^{r} - 1 \right) 4^{r-1} \right\}. \tag{4.37}$$

W celu poprawy efektywności obliczeniowej algorytmu rozwiązania, a tym samym skrócenia czasu generacji rozwiązania można wykorzystać procedurę dekompozycji [20]. Ostatecznie po zastosowaniu algorytmu rozpoznawania na poziomie górnym w miejsce koordynatora jazdy (rys. 3.5), struktura hierarchicznego systemu sterowania jest taka jak na rysunku 4.8.

Przypomnijmy jeszcze raz, że algorytm sterowania jazdą r-tego realizatora zlokalizowany na poziomie dolnym systemu sterowania, polega na wyznaczaniu w każdym takcie sterowania v wartości wielkości sterującej $\boldsymbol{u}_{r,v}$. W zależności od wyniku rozpoznawania j_r dla realizatora r są możliwe następujące przypadki.

1. Jeżeli $j_r = 1$, to należy rozwiązać problem sterowania jazdą pojedynczego realizatora, sformułowany w podpunkcie 3.2.1, czyli zminimalizować funkcję kry-



Rys. 4.8. System sterowania jazdą grupy realizatorów o strukturze hierarchicznej z rozpoznawaniem z reprezentacją wiedzy na poziomie górnym

terialną (3.8) względem zmiennej sterującej $\boldsymbol{u}_{r,v}$ dopuszczalnej w sensie (3.4), dla danych: stanu początkowego $\boldsymbol{x}_{r,0}$, wymagań na zakończenie jazdy zawartych w wektorze $\hat{\boldsymbol{x}}_r$, wag α_i i dokładności ε_r oraz ograniczeń na zmienne stanu (3.5), (3.6), a także obszarów $\boldsymbol{Y}_{r,v} = \boldsymbol{Y} = \boldsymbol{X}^{(1)} \times \boldsymbol{X}^{(2)}$. Należy podkreślić, że obszary $\boldsymbol{Y}_{r,v}$ są teraz stałe w czasie i niezależne od realizatora, co upraszcza problem optymalizacji (3.8).

2. Jeżeli $j_r = 2$, to należy dla realizatora r wyznaczyć punkt pośredni $\boldsymbol{y}_r^{\mathrm{P}} = [y_r^{\mathrm{P}(1)}, y_r^{\mathrm{P}(2)}]^{\mathrm{T}}$, gdzie $y_r^{\mathrm{P}(1)}$ i $y_r^{\mathrm{P}(2)}$ są współrzędnymi w kartezjańskim układzie współrzędnych i rozwiązać zagadnienie optymalizacyjne z punktu 1, z wektorem $\hat{\boldsymbol{x}}_r^{\mathrm{P}} = [y_r^{\mathrm{P}(1)}, y_r^{\mathrm{P}(2)}, x_r^{*(3)}, x_r^{*(5)}, x_r^{*(6)}]^{\mathrm{T}}$ w miejsce wektora $\hat{\boldsymbol{x}}_r$. Punkt pośredni jest wyznaczany, jeśli pojazd ma ominąć przeszkodę. Może to być przeszkoda stała (nieruchoma) lub ruchoma w postaci innego realizatora. Zakładamy, że nie ma innych przeszkód ruchomych niż realizatory. W przypadku, gdy omijaną przeszkodą dla realizatora *r* ma być inny realizator, oznaczany jako *s*, to obliczamy punkty dla obu realizatorów. Punkty pośrednie leżą na prostej prostopadłej do odcinka łączącego aktualne położenia tych pojazdów i przecinającej ten odcinek w połowie. Odległość d_r punktów pośrednich $\boldsymbol{y}_r^{\mathrm{P}}$ i $\boldsymbol{y}_s^{\mathrm{P}}$ od punktu przecięcia zależy od wymiarów geometrycznych realizatorów i od przyjętego marginesu bezpieczeństwa Δ. Jest ona obliczana według wzoru

$$d_r = \frac{\sqrt{(d_r^{(2)} + a_r)^2 + (d_r^{(1)})^2}}{2} + \Delta, \tag{4.38}$$

gdzie $d_r^{(1)}$, $d_r^{(2)}$, a_r są wymiarami geometrycznymi realizatora, podanymi na rys. 3.4. Pierwszy składnik w liczniku (4.38) oznacza odległość środka koła przedniego od najdalszego punktu realizatora. W podobny sposób wyznaczamy punkt pośredni w przypadku, gdy realizator omija przeszkodę stałą.

3. Jeżeli $j_r = 3$, to nie wyznaczamy zmiennej sterującej $u_{r,v}$.

4. Jeżeli $j_r = 4$, to ustalmy $u_{r,v} = 0$.

Algorytm rozpoznawania wieloetapowego z reprezentacją wiedzy

Do rozwiązania sformułowanego problemu rozpoznawania z reprezentacją wiedzy wykorzystamy metodę rozpoznawania wieloetapowego, która w wersji klasycznej została opisana w pracy [82]. Istota rozpoznawania wieloetapowego jest następująca. W kolejnych etapach rozpoznawania podejmuje się decyzje o tym, do jakiego podzbioru wszystkich możliwych klas należy klasa rozpoznawanego obiektu oraz o tym, jakie cechy ze zbioru wszystkich możliwych cech mają być uwzględniane

w następnym etapie rozpoznawania. Proces decyzyjny podzielony jest na $N = \frac{R(R-1)}{2}$ etapów równych liczbie par różnych realizatorów. Decyzja w etapie n, $n = 1, 2, ..., N$ obejmuje wyznaczenie podzbioru $\boldsymbol{D}_j^n \subseteq \boldsymbol{J}$. Podzbiór $\boldsymbol{D}_j^N$ jest ostatecznym wynikiem rozpoznawania. Zastosowanie rozpoznawania wieloetapowego polega, dla rozpatrywanego zagadnienia, na rozpoznawaniu w ramach etapu sytuacji, w której znajduje się para realizatorów r i s. Konieczność zapewnienia jazdy bezkolizyjnej wymaga sprawdzenia możliwości dalszej jazdy dla każdej pary realizatorów. W celu uniknięcia sytuacji polegającej na braku możliwości kontynuowania jazdy przez wszystkie realizatory naraz, gdy nie osiągnęły one jeszcze stanów końcowych, która może wystąpić w określonych specyficznych przypadkach, w zbiorze realizatorów wprowadzono priorytety. I tak, realizator mający niższy numer, ma wyższy priorytet. W n-tym etapie należy rozwiązać problem rozpoznawania dla pary realizatorów r i s, przy czym między zmiennymi n oraz r i s zachodzi związek

$$n = \begin{cases} s - r, \text{ dla } r = 1, \ s = 2, 3, ..., R \\ s - r + \sum_{i=1}^{r-1} (R - i), \text{ dla } r, s = 2, 3, ..., R, \ r < s \end{cases} . \tag{4.39}$$

Oznaczmy przez $F^{r,s}(\boldsymbol{\alpha}_x^r, \boldsymbol{\alpha}_x^s, \boldsymbol{\alpha}_x^{r,s}, \boldsymbol{\alpha}_j^r, \boldsymbol{\alpha}_j^s)$ koniunkcję wszystkich faktów z reprezentacji wiedzy F, zawierających formuły elementarne będące elementami ciągów $\boldsymbol{\alpha}_x^r, \boldsymbol{\alpha}_x^s, \boldsymbol{\alpha}_x^{r,s}, \boldsymbol{\alpha}_j^r, \boldsymbol{\alpha}_j^s$, a przez $F_x^{r,s}(\boldsymbol{\alpha}_x^r, \boldsymbol{\alpha}_x^s, \boldsymbol{\alpha}_x^{r,s})$ odpowiednią własność wejściową. Wówczas problem rozpoznawania w n-tym etapie polega dla danych $F_x^{r,s}$ i $F^{r,s}$ na wyznaczeniu takiego najmniejszego podzbioru $\boldsymbol{D}_j^n \subseteq \boldsymbol{D}_j^{n-1}$, dla którego jest prawdziwa implikacja

$$F_x^{r,s}(\boldsymbol{\alpha}_x^r, \boldsymbol{\alpha}_x^s, \boldsymbol{\alpha}_x^{r,s}) \wedge F^{r,s}(\boldsymbol{\alpha}_x^r, \boldsymbol{\alpha}_x^s, \boldsymbol{\alpha}_x^{r,s}, \boldsymbol{\alpha}_j^r, \boldsymbol{\alpha}_j^s) \rightarrow j \in \boldsymbol{D}_j^n . \tag{4.40}$$

Zastosowanie metody logiczno-algebraicznej umożliwia sprowadzenie rozwiązania tego problemu do rozwiązania równania

$$F_x^{r,s}(\boldsymbol{a}_x^r, \boldsymbol{a}_x^s, \boldsymbol{a}_x^{r,s}) \wedge F^{r,s}(\boldsymbol{a}_x^r, \boldsymbol{a}_x^s, \boldsymbol{a}_x^{r,s}, \boldsymbol{a}_j^r, \boldsymbol{a}_j^s) = 1 \tag{4.41}$$

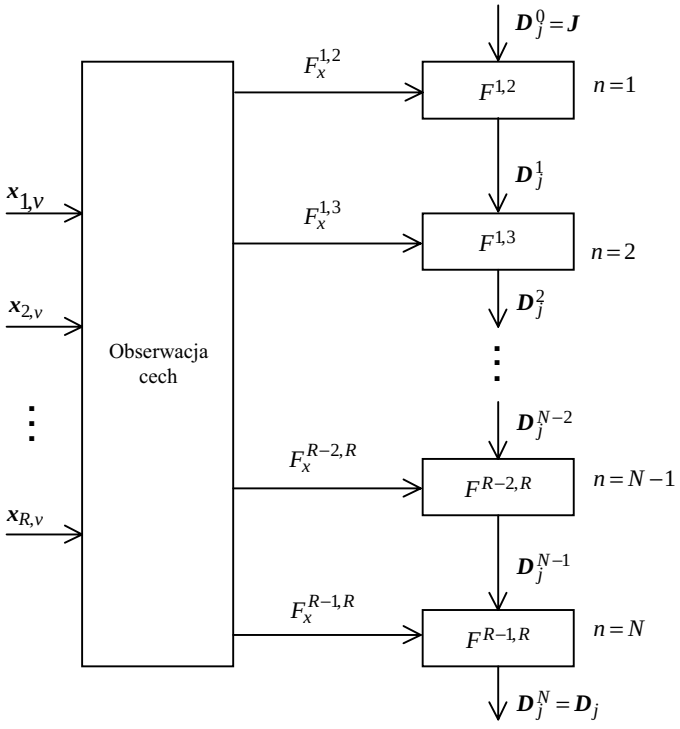
względem dopuszczalnych $(\boldsymbol{a}_j^r, \boldsymbol{a}_j^s)$, czyli takich, które zostały wyznaczone w poprzednich etapach. Jest to równoważne wyznaczeniu zbioru

$$\overline{\boldsymbol{S}}_{\mathrm{D}}^n = \left\{ \boldsymbol{a}_{\mathrm{D}} \in \overline{\boldsymbol{S}}_{\mathrm{D}}^{n-1} : \bigvee_{\boldsymbol{a}_x \in \boldsymbol{S}_x} F_x^{r,s}(\boldsymbol{a}_x^r, \boldsymbol{a}_x^s, \boldsymbol{a}_x^{r,s}) \wedge F^{r,s}(\boldsymbol{a}_x^r, \boldsymbol{a}_x^s, \boldsymbol{a}_x^{r,s}, \boldsymbol{a}_j^r, \boldsymbol{a}_j^s) = 1 \right\}, \tag{4.42}$$

gdzie $\overline{\boldsymbol{S}}_{\mathrm{D}}^{n-1} \subseteq \boldsymbol{S}_{\mathrm{D}}$ jest zbiorem dopuszczalnych wartości $\boldsymbol{a}_{\mathrm{D}}$, czyli zbiorem rozwiązań z etapu $n-1$, odpowiadającym $\boldsymbol{D}_j^{n-1}$. Wtedy zbiór $\boldsymbol{D}_j^n$ określony jest jako

$$\boldsymbol{D}_j^n = \left\{ j \in \boldsymbol{J} : \bigvee_{\boldsymbol{a}_{\mathrm{D}} \in \overline{\boldsymbol{S}}_{\mathrm{D}}^n} j = 1 + \sum_{r=1}^{R} \left(\sum_{i=1}^{4} n a_{j,i}^r - 1 \right) 4^{r-1} \right\}. \tag{4.43}$$

Ostatecznie, zbiór rozwiązań $\boldsymbol{D}_j = \boldsymbol{D}_j^N$ wyznaczany jest za pomocą procedury rekurencyjnej (4.42) i wzoru (4.43) dla $\boldsymbol{D}_j^0 = \boldsymbol{J}$ oraz $\overline{\boldsymbol{S}}_{\mathrm{D}}^0 = \boldsymbol{S}_{\mathrm{D}}$ (rys. 4.9). Jeżeli otrzymamy niejednoznaczną odpowiedź, tj. jeśli zbiór $\boldsymbol{D}_j^N$ zawiera więcej niż jeden element, to należy wybrać $j^* \in \boldsymbol{D}_j^N$ tak, aby jak najwięcej pojazdów było w ruchu. Można łatwo wykazać, że $j^* = \min\{ j \in \boldsymbol{D}_j^N \}$, czyli j^* jest numerem klasy o najmniejszej wartości, a więc w przypadku niejednoznacznego wyniku rozpoznawania wybieramy klasę o najmniejszym numerze.



Rys. 4.9. Rozpoznawanie wieloetapowe z reprezentacją wiedzy w bieżącym takcie sterowania

Zgodnie z (4.39) rozpoznawanie odbywa się kolejno, począwszy od realizatorów o najniższych numerach, czyli najwyższych priorytetach. Priorytet oznacza, że w przypadku wystąpienia niebezpieczeństwa kolizji – gdy zachodzi taka konieczność – zatrzymuje się realizator o niższym priorytecie (w stosowanej notacji jest to realizator *s*).

Reprezentacja wiedzy w n-tym etapie i eksperymenty obliczeniowe

Reprezentacja wiedzy dotycząca pojazdów *r* i *s*, wykorzystywana w takcie sterowania ν i *n*-tym etapie algorytmu rozpoznawania, składa się z ośmiu faktów. Zgodnie z ogólną zasadą są one zbudowane z formuł elementarnych $\boldsymbol{\alpha}_{\mathrm{D}}$ dotyczących klas oraz z następujących formuł elementarnych dotyczących cech:

1. $\alpha_{x,1}^{r,s} = \sqrt{\sum_{i=1}^{2} (x_{r,\nu}^{(i)} - x_{s,\nu+1}^{(i)})^2} < d$,

2. $\alpha_{x,2}^{r,s} = \sqrt{\sum_{i=1}^{2} (x_{r,\nu+1}^{(i)} - x_{s,\nu}^{(i)})^2} < d$,

3. $\alpha_{x,3}^{r,s} = \sqrt{\sum_{i=1}^{2} (x_{r,\nu+1}^{(i)} - x_{s,\nu+1}^{(i)})^2} < d$,

4. $\alpha_{x,4}^{r,s} = \pi - \theta < \left| x_{r,\nu}^{(5)} - x_{s,\nu}^{(5)} \right| < \pi + \theta$,

5. $\alpha_{x,1}^{r} = \sqrt{\sum_{i=1}^{2} (x_{r,\nu}^{(i)} - \hat{x}_{r}^{(i)})^2} < \varepsilon_r$,

6. $\alpha_{x,1}^{s} = \sqrt{\sum_{i=1}^{2} (x_{s,\nu}^{(i)} - \hat{x}_{s}^{(i)})^2} < \varepsilon_s$.

Występujące w formułach wielkości $x_{r,\nu+1}^{(i)}$ i $x_{s,\nu+1}^{(i)}$, $i = 1, 2$ odnoszą się do przewidywanych położeń realizatorów wyznaczonych na przykład na podstawie zależności (3.10). Parametr *d* jest stałą związaną z odległością (4.38) między realizatorami i jest przyjmowany jako

$$d > \max_{\substack{r,s=1,2,\ldots,R \\ r \neq s}} \{d_r + d_s\}.$$

Określa on minimalną odległość między realizatorami, przy której jest stwierdzane zagrożenie wystąpienia kolizji. Z kolei parametr θ jest graniczną wartością kąta wyznaczonego przez kierunki jazdy realizatorów *r* i *s*, przy której zbliżające się realizatory mogą się ominąć bez potrzeby zmiany swoich trajektorii jazdy. Jeśli kierunki jazdy realizatorów przecinają się pod kątem mniejszym niż θ, to musi zostać dokonana modyfikacja ich trajektorii przez wyznaczenie punktów pośrednich $\boldsymbol{y}_r^{\mathrm{P}}$ i $\boldsymbol{y}_s^{\mathrm{P}}$. W przeciwnym przypadku dla uniknięcia kolizji wystarczy, aby realizator o niższym priorytecie zatrzymał się. Parametry ε_r i ε_s są zadanymi dokładnościami osiągnięcia stanów końcowych, wprowadzonymi w warunku stopu (3.9). Poszczególne formuły elementarne mają następującą interpretację:

$\alpha_{x,1}^{r,s}$ – odległość między położeniem realizatora *r* w bieżącym takcie ν a prognozowanym położeniem realizatora *s* w takcie następnym $\nu+1$ jest mniejsza od wartości parametru *d*,

$\alpha_{x,2}^{r,s}$ – odległość między prognozowanym położeniem realizatora *r* w takcie następnym $\nu+1$ a położeniem realizatora *s* w bieżącym takcie ν jest mniejsza od wartości parametru *d*,

$\alpha_{x,3}^{r,s}$ – odległość między prognozowanymi położeniami realizatorów *r* i *s* w takcie następnym $\nu+1$ jest mniejsza od wartości parametru *d*,

$\alpha_{x,4}^{r,s}$ – kąt między kierunkami jazdy realizatorów *r* i *s* zawiera się w przedziale $(\pi-\theta, \pi+\theta)$,

$\alpha_{x,1}^{r}$ – realizator *r* osiągnął cel jazdy,

$\alpha_{x,1}^{s}$ – realizator *s* osiągnął cel jazdy.

Przyjęto następujące fakty, czyli reprezentację wiedzy:

$$F_1^{r,s} = \alpha_{x,1}^{r} \rightarrow \alpha_{j,3}^{r},$$

$$F_2^{r,s} = \neg\alpha_{x,1}^{r} \rightarrow \left(\alpha_{j,1}^{r} \vee \alpha_{j,2}^{r} \vee \alpha_{j,4}^{r}\right),$$

$$F_3^{r,s} = \left(\alpha_{x,1}^{r,s} \vee \alpha_{x,2}^{r,s} \vee \alpha_{x,3}^{r,s}\right) \wedge \left(\alpha_{x,4}^{r,s} \vee \alpha_{j,4}^{s}\right) \rightarrow \left(\alpha_{j,2}^{r} \vee \alpha_{j,4}^{r}\right),$$

$$F_4^{r,s} = \alpha_{x,1}^{s} \rightarrow \alpha_{j,3}^{s},$$

$$F_5^{r,s} = \neg\alpha_{x,1}^{s} \rightarrow \left(\alpha_{j,1}^{s} \vee \alpha_{j,2}^{s} \vee \alpha_{j,4}^{s}\right),$$

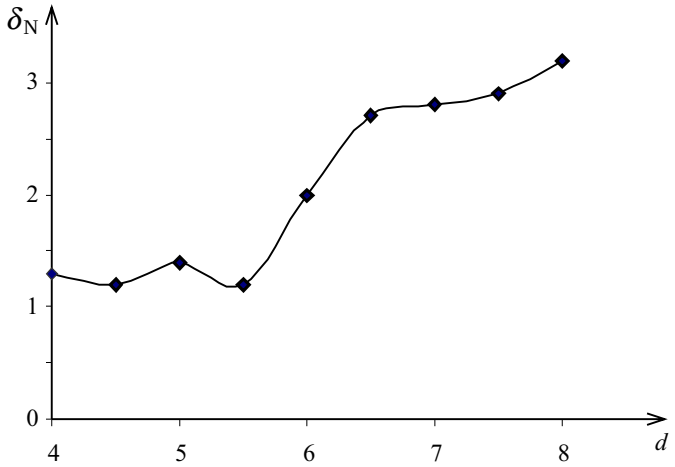
$$F_6^{r,s} = \left(\alpha_{j,1}^{r} \vee \alpha_{j,2}^{r}\right) \wedge \left(\alpha_{x,1}^{r,s} \vee \alpha_{x,2}^{r,s} \vee \alpha_{x,3}^{r,s} \vee \alpha_{x,4}^{r,s}\right) \rightarrow \left(\alpha_{j,2}^{s} \vee \alpha_{j,4}^{s}\right),$$

$$F_7^{r,s} = \left(\alpha_{j,1}^{r} \wedge \neg\alpha_{j,2}^{r} \wedge \neg\alpha_{j,3}^{r} \wedge \neg\alpha_{j,4}^{r}\right) \vee \left(\neg\alpha_{j,1}^{r} \wedge \alpha_{j,2}^{r} \wedge \neg\alpha_{j,3}^{r} \wedge \neg\alpha_{j,4}^{r}\right)$$
$$\vee \left(\neg\alpha_{j,1}^{r} \wedge \neg\alpha_{j,2}^{r} \wedge \alpha_{j,3}^{r} \wedge \neg\alpha_{j,4}^{r}\right) \vee \left(\neg\alpha_{j,1}^{r} \wedge \neg\alpha_{j,2}^{r} \wedge \neg\alpha_{j,3}^{r} \wedge \alpha_{j,4,}^{r}\right),$$

$$F_8^{r,s} = \left(\alpha_{j,1}^s \wedge \neg\alpha_{j,2}^s \wedge \neg\alpha_{j,3}^s \wedge \neg\alpha_{j,4}^s\right) \vee \left(\neg\alpha_{j,1}^s \wedge \alpha_{j,2}^s \wedge \neg\alpha_{j,3}^s \wedge \neg\alpha_{j,4}^s\right)$$
$$\vee \left(\neg\alpha_{j,1}^s \wedge \neg\alpha_{j,2}^s \wedge \alpha_{j,3}^s \wedge \neg\alpha_{j,4}^s\right) \vee \left(\neg\alpha_{j,1}^s \wedge \neg\alpha_{j,2}^s \wedge \neg\alpha_{j,3}^s \wedge \alpha_{j,4}^s\right).$$

W faktach $F_1^{r,s}$ i $F_2^{r,s}$ są odpowiednio zawarte informacje o warunkach zakończenia i możliwości prowadzenia dalszej jazdy przez realizator *r*, który ma wyższy priorytet. Fakty $F_4^{r,s}$ i $F_5^{r,s}$ odnoszą się do realizatora *s* i mają analogiczne znaczenie. Wystąpienie sytuacji, w której należy podjąć odpowiednie decyzje w celu uniknięcia kolizji, dla realizatorów *r* i *s* jest opisane w formie implikacji odpowiednio w faktach $F_3^{r,s}$ i $F_6^{r,s}$. Przykładowo, dla realizatora *r* (fakt $F_3^{r,s}$) sytuacja taka oznacza, że realizator ten może jechać z omijaniem przeszkód ($\alpha_{j,2}^r$) lub musi się zatrzymać ($\alpha_{j,4}^r$). Aby tak się stało, muszą zajść dwa warunki. Pierwszy polegający na tym, że co najmniej jedna z odległości między położeniami realizatorów *r* i *s* w taktach ν i $\nu+1$ jest mniejsza niż *d* (pierwsza, druga lub trzecia formuła elementarna). Drugi warunek oznacza, że kąt między kierunkami jazdy realizatorów jest na tyle mały, że należy wyznaczać punkty pośrednie lub realizator musi się zatrzymać. Fakty $F_7^{r,s}$ i $F_8^{r,s}$ zapewniają rozpoznanie tylko jednej klasy odpowiednio dla realizatora *r* i *s*.

Przeprowadzone badania symulacyjne zaproponowanego algorytmu rozwiązania problemu sterowania jazdą grupy realizatorów z unikaniem kolizji, w których jazda realizatorów była symulowana komputerowo – wykazały większą jego przydatność od algorytmu opisanego w podpunkcie 3.2.1. Wyników tych badań nie będziemy tu



Rys. 4.10. Zależność δ_N od *d*

omawiać. W charakterze przykładu przedstawimy jedynie przebieg zależności empirycznego wskaźnika

$$\delta_{\mathrm{N}} = \frac{N_f}{N} 100\%$$

od wartości parametru *d* (rys. 4.10). Wskaźnik δ_{N} jest wyrażonym w procentach stosunkiem liczby przypadków, w których algorytm zadziałał nieprawidłowo, to znaczy, gdy jazda nie została pomyślnie zakończona do liczby wszystkich przypadków, w których sprawdzano działanie algorytmu. Wyniki prezentowane na rys. 4.10 są średnimi wartościami dla tysiąca różnych przypadków, czyli dla tysiąca wygenerowanych losowo par stanów ($\boldsymbol{x}_{r,0}$, $\boldsymbol{x}_r^*$). Badanie prowadzone dla dwóch realizatorów wskazało na silną zależność jakości algorytmu od wartości parametru *d*. Okazało się, że ustalenie zbyt dużych wartości *d* wcale nie prowadzi do wzrostu jakości algorytmu.

5. Uwagi końcowe

W pracy dokonano przeglądu wybranych problemów decyzyjnych dla systemów typu kompleks operacji, czyli takich, których elementy (operacje) są powiązane ze sobą na zasadzie kolejności czasowych, wyrażonych w postaci ograniczeń kolejnościowych. Zasadnicze znaczenie metodologiczne mają rozdziały 2., 3. i 4., ponieważ w rozdziale 1. zaprezentowano przede wszystkim te dobrze znane zagadnienia, których znajomość jest niezbędna do lektury dalszych rozdziałów. Dotyczy to problemów alokacji oraz szeregowania.

W rozdziale 2. rozszerzono, w stosunku do [58], problematykę szeregowania zadań z uwzględnieniem ruchu realizatorów o przypadek szeregowania zadań niezależnych i niepodzielnych na realizatorach dowolnych w celu minimalizacji maksymalnego opóźnienia. Do rozwiązania tego problemu zastosowano analogiczne metody badawcze, jak w przypadku minimalizacji długości uszeregowania. Uzyskane w wyniku zastosowania dekompozycji funkcjonalnej różne wersje algorytmów rozwiązania, pozwalają na wyznaczenie tras przejazdów realizatorów między stanowiskami produkcyjnymi w postaci dróg Hamiltona, a nie cykli Hamiltona jak w przypadku minimalizacji długości uszeregowania. Sprawa kontynuacji tej problematyki, polegającej na rozpatrywaniu nowych, bardziej złożonych przypadków, bliższych zastosowaniom praktycznym, pozostaje otwarta. Obecnie są opracowywane problemy szeregowania z ruchomymi realizatorami dla minimalizacji sumy momentów zakończenia zadań oraz dla minimalizacji długości uszeregowania z uwzględnieniem ograniczeń kolejnościowych w zbiorze zadań.

Problematyka polegająca na łącznym modelowaniu i rozwiązywaniu tradycyjnych problemów decyzyjnych dla kompleksów operacji oraz sterowania ruchem lub transportem różnych elementów kompleksu operacji produkcyjnych po raz pierwszy została przedstawiona w monografii. Analiza sklasyfikowanych w punkcie 3.1 szczegółowych problemów z tego zakresu wskazuje, że podjęcie w monografii niektórych z nich oraz prezentacja algorytmów ich rozwiązania są zaledwie początkiem tej ciekawej i niezwykle ważnej z praktycznego punktu widzenia problematyki badawczej. Uzyskane wyniki dla szeregowania oraz sterowania jazdą realizatorów i środków transportu przewożących obiekty, a także alokacji i transportu zasobów

oraz surowców i(lub) produktów wskazują na konieczność poszukiwania nowych, bardziej efektywnych algorytmów rozwiązania dla tych i innych przypadków przewidzianych dopiero do rozwiązania. Bardzo ważne jest podjęcie w tym zakresie prac związanych z zastosowaniem opracowanych algorytmów rozwiązania w specjalizowanych systemach informatycznych do analizy i symulacji, ale również do wspomagania decyzji, przede wszystkim w dyskretnych systemach produkcyjnych. Pojawiające się już obecnie wspomniane systemy informatyczne posiadają możliwości łączenia w jednym systemie metod symulacyjnych oraz procedur umożliwiających rozwiązywanie różnych problemów decyzyjnych, na przykład rozdziału zasobów lub ustalania kolejności wykonywania operacji. Pogłębionej analizy i w następstwie ewentualnego rozpatrzenia wymaga sprawa rozszerzenia problematyki podejmowania decyzji w kompleksach operacji z uwzględnieniem ruchu na systemy informatyczne, w których rolę poruszających się realizatorów pełniłyby specjalne procedury przemieszczające się w sieciach informatycznych. Łączy się to z zagadnieniami tak zwanych systemów wieloagentowych.

O ile w rozdziałach 2. i 3. skoncentrowano się na prezentacji oryginalnych problemów badawczych oraz algorytmów ich rozwiązania uzyskiwanych w wyniku zastosowania dobrze znanych metod i narzędzi, o tyle w rozdziale 4., dla wybranych i sformułowanych wcześniej zagadnień, przedstawiono efekty zastosowania nowych metod rozwiązywania problemów decyzyjnych. Chodzi o bardzo popularne obecnie i intensywnie rozwijane metody sztucznej inteligencji. Ponieważ omawiane problemy decyzyjne są specyficznymi problemami optymalizacji dyskretnej, i tak były przedstawiane, naturalne było wykorzystanie algorytmów ewolucyjnych, które generalnie są przeznaczone do rozwiązywania problemów optymalizacyjnych. Uzyskane dotychczas wyniki, których przykłady przedstawiono dla szczególnego przypadku szeregowania zadań, wskazują na konieczność dalszego doskonalenia tego narzędzia. Obiecujące wydaje się być stosowanie rozwijanych już od kilku lat metod hybrydowych, polegających na połączeniu różnych technik sztucznej inteligencji, na przykład algorytmów ewolucyjnych i sztucznych sieci neuronowych. Uwaga o możliwości wykorzystania metod hybrydowych ma większe znaczenie i nie dotyczy tylko rozwiązywania problemów szeregowania, ale również pozostałych problemów decyzyjnych dla kompleksów operacji.

Bez wątpienia warte dalszych studiów i badań są jeszcze dwa zagadnienia dla kompleksów operacji, które w monografii zostały jedynie zasygnalizowane. Pierwsze z nich dotyczy rozpatrywania określonych problemów w warunkach niepewności, to znaczy wtedy, gdy opis kompleksu operacji nie jest w pełni znany lub posiadane informacje nie są pewne. Sprawa ta została zasygnalizowana pod koniec rozdziału 2. dla szeregowania, a także w rozdziale 4. dla rozdziału zadań. Pierwsze,

wstępne rezultaty wskazują, że obiecujące jest stosowanie do opisu i rozwiązywania problemów decyzyjnych sposobów i metod wykorzystujących wybraną logikę wielowartościową lub rozmytą. Drugie ze wspomnianych zagadnień, które należy rozwijać dla kompleksów operacji, polega na wykorzystaniu algorytmów uczenia, w celu precyzowania informacji o systemie, będącym przedmiotem podejmowania decyzji, zwłaszcza w przypadkach niedeterministycznych. Specyfika kompleksów operacji wymaga tu opracowywania odrębnych algorytmów uczenia, pozwalających na uzyskiwanie rozwiązań zbieżnych do rozwiązań optymalnych. W monografii podano przykłady wykorzystania algorytmów uczenia w dwóch przypadkach: do określania niepewnych parametrów w modelach czasowych kompleksu operacji dla rozdziału zadań oraz do wyznaczania niektórych parametrów algorytmu ewolucyjnego, stosowanego do rozwiązania problemu szeregowania zadań z ruchomymi realizatorami.

Monografia została opracowana w ramach realizacji projektu badawczego nr 7 T11A 039 20 pt. *Algorytmy podejmowania decyzji i uczenia w systemach niepewnych i kompleksach operacji z reprezentacją wiedzy*, finansowanego przez Komitet Badań Naukowych.

Literatura

[1] ACKHOF R.L., SASIENI M.W., *Fundamentals of operations research*, Wiley, New York 1968.

[2] BADACH A., MOLISZ W., SEIDLER J., *Metody rozwiązywania zadań optymalizacji*, WNT, Warszawa 1980.

[3] BŁASIAK P., *Analiza i projekt informatycznego systemu zarządzania transportem w przedsiębiorstwie produkcyjnym* (praca dyplomowa), Instytut Sterowania i Techniki Systemów Politechniki Wrocławskiej, Wrocław 1995.

[4] BŁAŻEWICZ J., *Złożoność obliczeniowa problemów kombinatorycznych*, WNT, Warszawa 1988.

[5] BŁAŻEWICZ J., CELLARY W., SŁOWIŃSKI R., WĘGLARZ J., *Algorytmy sterowania rozdziałem zadań i zasobów w kompleksie operacji*, Wydawnictwo Politechniki Poznańskiej, Poznań 1978.

[6] BŁAŻEWICZ J., CELLARY W., SŁOWIŃSKI R., WĘGLARZ J., *Badania operacyjne dla informatyków*, WNT, Warszawa 1983.

[7] BŁAŻEWICZ J., DROR M., WĘGLARZ J., *Mathemetical programming formulations for maschine scheduling: A survey*, European Journal of Operational Research, vol. 51, 1991, s. 283–300.

[8] BŁAŻEWICZ J., DROZDOWSKI M., WĘGLARZ J., *Scheduling multiprocessor tasks – a survey*, Microcomputer Applications, vol. 13, no. 2, 1994, s. 89–97.

[9] BŁAŻEWICZ J., ECKER K.H., PESCH E., SCHMIDT G., WĘGLARZ J., *Scheduling computer and manufacturing processes*, Springer Verlag, Berlin, Heidelberg 1996.

[10] BŁAŻEWICZ J., LENSTRA J.K., RINOOY KAN A.H.G., *Scheduling subject to resource constraints: classification and complexity*, Disctete Applied Mathematics, vol. 5, 1983, s. 11–24.

[11] BŁAŻEWICZ J., WĘGLARZ J., *O pewnym problemie szeregowania maszyn i wózków w elastycznym systemie produkcyjnym*, Zeszyty Naukowe Politechniki Śląskiej, seria: Automatyka, z. 101, Gliwice 1990.

[12] BUBNICKI Z., *Optimal control of the complex of operations with random parameters*, Podstawy Sterowania, t. 1, z. 1, 1971.

[13] BUBNICKI Z., *O pewnych problemach czasowo-optymalnego sterowania kompleksem operacji w warunkach probabilistycznych*, Podstawy Sterowania, t. 2, nr 3, 1972.

[14] BUBNICKI Z., *Identyfikacja obiektów sterowania*, PWN, Warszawa 1974.

[15] BUBNICKI Z., *Time optimal control of dependent operations with random parameters*, Systems Science, vol. 3, no. 3, 1977.

[16] BUBNICKI Z., *Two-level optimization and control of the complex of operations*, Materiały VII Kongresu IFAC, vol. 2, Helsinki, Pergamon Press, Oxford 1978.

[17] BUBNICKI Z., *Multilevel optimization and control of the complex of operations*, [w:] Systems Analysis Applications of Complex Programs, Pergamon Press, Oxford–New York 1978.

[18] BUBNICKI Z., *Optymalizacja kompleksów operacji w sterowaniu dyskretnymi procesami produkcyjnymi*, Prace VII Krajowej Konferencji Automatyki, t. 3 (referaty plenarne), Rzeszów 1979.

[19] BUBNICKI Z., *Wstęp do systemów ekspertowych*, PWN, Warszawa 1990.

[20] BUBNICKI Z., *Rekurencyjne algorytmy rozwiązywania zadań z reprezentacją wiedzy w systemie ekspertowym*, Archiwum Informatyki Teoretycznej i Stosowanej, t. 3, z. 1–4, 1992.

[21] BUBNICKI Z., *Knowledge-based approach as a generalization of pattern recognition problems and methods,* Systems Science, vol. 19, no. 2, 1993, s. 5–21.

[22] BUBNICKI Z., *Podstwy informatycznych systemów zarządzania*, Wydawnictwo Politechniki Wrocławskiej, Wrocław 1993.

[23] BUBNICKI Z., *Application of neural networks to complex operation systems*, [w:] Modelling, Identification and Control (red.: Hamza M.H.), Acta Press, Zurich 1994, s. 264–267.

[24] BUBNICKI Z., *Simulation of an industrial transport system*, Proceedings of 8th European Simulation Symposium, Genua, vol. 1, 1996, s. 305–309.

[25] BUBNICKI Z., *Learning control system for a class of production operations with parametric uncertainties*, [w:] Preprints of IFAC Symposium on Manufacturing, Modelling, Management and Control (red.: Groumpos P.G., Tzes P.), Patras, Grecja 2000, s. 228–233.

[26] BUBNICKI Z., *Uncertain variables in the computer aided analysis of uncertain systems*, [w:] Computer aided systems theory. Lecture Notes in Computer Science (red. Pichler F., Moreno-Diaz R., Kopacek P.), vol. 1798, 2000, s. 528–542.

[27] BUBNICKI Z., WOŁKOWIŃSKI K., *On the allocation problem in a production system with transport of materials*, Systems Science, vol. 24, no. 3, 1998, s. 89–100.

[28] BURKOV V.N., *Raspredelenie riesursov kak zadacza optimalnogo bystrodeistwija*, Avtomatika i Telemiechanika, vol. 27, no. 7, 1966 (w jęz. rosyjskim).

[29] BURKOV V.N., *Optimalnoe upravlenie kompleksami operacii*, Materiały IV Kongresu IFAC, t. 35, Warszawa, 1969, s. 46–57 (w jęz. rosyjskim).

[30] CICHOCKI A., UNBECHAUEN R., *Neural networks for optimization and signal processing*, Wiley, Chichester 1993.

[31] COFFMAN E.G. Jr. (red.), *Teoria szeregowania zadań*, WNT, Warszawa 1980.

[32] COX E., *The fuzzy systems handbook*, Academic Press, London 1994.

[33] CRAIG J.J., *Wprowadzenie do robotyki. Mechanika i sterowanie*, WNT, Warszawa 1993.

[34] CYTOWSKI J., *Algorytmy genetyczne. Podstawy i zastosowania*, Akademicka Oficyna Wydawnicza PLJ, Warszawa 1996.

[35] CZOGAŁA E., PEDRYCZ W., *Elementy i metody teorii zbiorów rozmytych*, PWN, Warszawa 1985.

[36] DANIELS R.L., KOUVELIS P., *Robust scheduling to hedge against processing time uncertainty in single–stage production*, Management Science, vol. 41, no. 2, 1995, s. 363–376.

[37] DUCH W., KORBICZ J., TADEUSIEWICZ R., RUTKOWSKI L. (red.), *Sieci neuronowe*, Akademicka Oficyna wydawnicza Exit, Warszawa 2000.

[38] FILIPOWICZ B., *Modele stochastyczne w badaniach operacyjnych. Analiza i synteza systemów i sieci kolejkowych*, WNT, Warszawa 1996.

[39] FILIPOWICZ B., *Modelowanie i analiza sieci kolejkowych*, Wydawnictwo AGH, Kraków 1997.

[40] FINDEISEN W., SZYMANOWSKI J., WIERZBICKI A., *Teoria i metody obliczeniowe optymalizacji*, PWN, Warszawa 1980.

[41] FISZ M., *Rachunek prawdopodobieństwa i statystyka matematyczna*, PWN, Warszawa 1969.

[42] FRENCH S., *Sequencingn and scheduling: An introduction to the mathematics for the jobshop*, Ellis Horwood Limited, Chichester 1982.

[43] GARREY M.R., JOHNSON D.S., *Computers and intractability: A Guide to the theory of NP–completness*, Freeman, San Francisco, CA 1979.

[44] GRAHAM M.R., LAWLER E.L., LENSTRA J.K., RINOOY KAN A.H.G., *Optimization and approximation in deterministic sequencing and scheduling theory: a survey*, Annals of Discrete Mathematics, vol. 5, 1979, s. 287–326.

[45] HAPKE M., SŁOWIŃSKI R., *Fuzzy priority heuristics for project scheduling*, Fuzzy Sets and Systems, vol. 83, no. 3, 1996, s. 291–299.

[46] HOLLAND J.H., *Adaptation in natural and artificial systems*, Ann Arbor, University of Michigan Press 1975.

[47] IBARRA O.H., KIM C.E., *Heuristic algorithms for scheduling independent tasks on nonidentical processors*, Journal of the ACM, vol. 24, no. 2, 1977, s. 280–289.

[48] JAILLET P., *A priori solution of a travelling salesman problem in which a random subset of customers are visited*, Operations Research, vol. 36, no. 6, 1988, s. 929–936.

[49] JAISVAL N.K., *Priority queues*, Academic Press, New York–London 1968.

[50] JÓZEFCZYK J., *Pewien problem sterowania nadrzędnego dyskretnymi procesami produkcji na przykładzie procesu zgrzewania obudowy pralki automatycznej*, Zeszyty Naukowe Politechniki Śląskiej, seria: Automatyka, z. 95, Gliwice 1988, s. 19-29.

[51] JÓZEFCZYK J., *On the two level control of the flexible manufacturing system with mobile robots*, Proc. of 8th International Conference on Systems Engineering, Coventry, UK 1991, s. 464-471.

[52] JÓZEFCZYK J., *A two-level approach to the motion control and the machine loading problem in the flexible manufacturing system*, Proceedings of 9th International Conference on Systems Engineering, Las Vegas 1993, s. 8-12 (Addendum).

[53] JÓZEFCZYK J., *Modelling and control of the manufacturing system with mobile executors*, Proceedings of International Symposium "Mathematical Models in Automation and Robotics" MMAR '94, Międzyzdroje 1994, s. 115–120.

[54] JÓZEFCZYK J., *Wybrany problem dwupoziomowego sterowania kompleksem operacji produkcyjnych z lokalnie sterowanymi operacjami*, Zeszyty Naukowe Politechniki Śląskiej, seria: Automatyka, z. 114, Gliwice 1994, s. 111–121.

[55] JÓZEFCZYK J., *Two level control algorithm for mobile executors in flexible manufacturing systems*, Proceedings of 10th International Conference on Systems Engineering, Coventry, UK 1994, s. 542–549.

[56] JÓZEFCZYK J., *On the problem of tasks scheduling on moving executors*, Proceedings of 12th International Conference on Systems Science, Technical University of Wrocław, Wrocław 1995, s. 314-319.

[57] JÓZEFCZYK J., *Wybrane problemy szeregowania zadań produkcyjnych z uwzględnieniem ruchu realizatorów*, Zeszyty Naukowe Politechniki Śląskiej, seria Automatyka, z. 117, Gliwice 1996, s. 149–160.

[58] JÓZEFCZYK J., *Szeregowanie zadań w kompleksie operacji z uwzględnieniem ruchu realizatorów*, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 1996.

[59] JÓZEFCZYK J., *On the functional decomposition approach to the problem of tasks scheduling on moving executors*, Proceedings of 11th International Conference on Systems Engineering, Las Vegas, USA 1996, s. 885–890.

[60] JÓZEFCZYK J., *An algorithm for scheduling tasks on moving executors in complex operation systems*, 1st IFAC Workshop on Manufacturing Systems MIM'97, Wiedeń 1997, s. 139–144.

[61] JÓZEFCZYK J., *Simualation of the complex operation system with moving executors*, Proceedings of 12th International Conference on Systems Engineering, Coventry, UK 1997, s. 349–354.

[62] JÓZEFCZYK J., *Scheduling tasks on moving executors – new results*, Proceedings of XIII International Conference on Systems Science, Wrocław 1998, s. 73-83.

[63] JÓZEFCZYK J., *Szeregowanie zadań w kompleksie operacji z uwzględnieniem ruchu realizatorów – badania symulacyjne dla wybranych przypadków*, Zeszyty Naukowe Politechniki Śląskiej, seria Automatyka, z.123, Gliwice 1998, s. 199–210.

[64] JÓZEFCZYK J., *Evaluation of the solution method for scheduling tasks on moving executors to minimise the maximum lateness*, Proceedings of 5th International Symposium on Methods and Models in Automation and Robotics MMAR '98, Międzyzdroje 1998, s. 1053–1058.

[65] JÓZEFCZYK J., *Systemy z reprezentacją wiedzy. Systemy ekspertowe*, [w:] Problemy Automatyki i Informatyki, Ossolineum, Wrocław 1998, s. 71–81.

[66] JÓZEFCZYK J., *Szeregowanie zadań w kompleksie operacji z uwzględnieniem ruchu realizatorów w ujęciu stochastycznym*, [w:] XIII Krajowa Konferencja Automatyki (red. Z. Bubnicki, J. Józefczyk), Oficyna Wydawnicza Politechniki Opolskiej, Opole 1999, s. 109–116.

[67] JÓZEFCZYK J., *Application of knowledge based pattern recognition in the control system of a group of executors*, [w:] 11th International Congress of Cybernetics and Systems. Conference Proceedings (red. R. Valee, J. Rose), World Organisation of Systems and Cybernetics, Uxbridge, Middlesex 1999, s. 330–333.

[68] JÓZEFCZYK J., *Sterowanie jazdą z reprezentacją wiedzy grupy pojazdów autonomicznych w dwupoziomowym systemie sterowania kompleksem operacji*, [w:] XIII Krajowa Konferencja Automatyki (red. Z. Bubnicki, J. Józefczyk), Oficyna Wydawnicza Politechniki Opolskiej, Opole 1999, s. 117–122.

[69] JÓZEFCZYK J. *On the problem of scheduling tasks on moving executors with random processing times*, Systems Science, vol. 25, no. 4 1999, s. 5–13.

[70] JÓZEFCZYK J., *Knowledge based motion control of a group of mobile executors in the two-level complex operation system*, Proceedings of the 2nd World Manufacturing Congress WMC ’99 (red. S. Nahavandi, M. Saadat), University of Durham, UK 1999, s. 167–173.

[71] JÓZEFCZYK J., *Zastosowanie algorytmu genetycznego w problemie szeregowania zadań z ruchomymi realizatorami*, [w:] Inżynieria Wiedzy i Systemy Ekspertowe (red. Z. Bubnicki, A. Grzech), Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2000, s. 29–36.

[72] JÓZEFCZYK J., *Algorithms for decision making in two-level manufacturing operation system*, Proceedings of 14th International Conference on Systems Engineering, Coventry, UK, vol.1, 2000, s. 288–293.

[73] JÓZEFCZYK J., *Scheduling of manufacturing tasks with uncertain processing times*, Proceedings of 14th International Conference on Systems Engineering, Coventry, UK, vol.1, 2000, s. 294–299.

[74] JÓZEFCZYK J., *Decision making and simulation for two- level manufacturing operation system*, Proceedings of 6th International Conference on “Methods and Models in Automation and Robotics” MMAR 2000 (red. S. Domek, R. Kaszyński), Międzyzdroje 2000, s. 877–882.

[75] JÓZEFCZYK J., *Scheduling tasks on moving executors to minimise the maximum lateness*, European Journal of Operational Research, vol. 131, no. 1, 2001, s. 171–187.

[76] JÓZEFCZYK J., KALINOWSKI P*., Mathematical model and exact algorithm for scheduling of tasks on moving executors*, Proceedings of 4th International Symposium “Methods and Models in Automation and Robotics MMAR ‘97”, Międzyzdroje 1997, s. 1037–1042.

[77] JÓZEFCZYK J., ORSKI D., *Algorytm sterowania dwupoziomowym kompleksem operacji produkcyjnych*, Zeszyty Naukowe Politechniki Śląskiej, z. 129, Gliwice 2000, s. 191–200.

[78] JÓZEFCZYK J., SZALA M., *Sterowanie pojazdami autonomicznymi z wykorzystaniem rozpoznawania z reprezentacją wiedzy*, Zeszyty Naukowe Politechniki Śląskiej, z. 129, Gliwice 2000, s. 201–210.

[79] KACPRZYK J., *Zbiory rozmyte w analizie systemowej*, PWN, Warszawa 1986.

[80] KNAPCZYK J. (red.), *Podstawy robotyki. Teoria i elementy manipulatorów i robotów*, WNT, Warszawa 1999.

[81] KORBICZ J., OBUCHOWICZ A., UCIŃSKI D., *Sztuczne sieci neuronowe, podstawy i zastosowania*, Akademicka Oficyna Wydawnicza, Warszawa 1994.

[82] KURZYŃSKI M., *Algorytmy rozpoznawania wieloetapowego oraz ich zastosowania medyczne i techniczne*, Prace Naukowe Instytutu Sterowania i Techniki Systemów Politechniki Wrocławskiej, seria: Monografie, nr 3 1987.

[83] LANGSTON M.A., *Interstage transportation planning in the deterministic flow–shop environment*, Operations Research, vol. 35, no. 4, 1987.

[84] LATOMBE J.C., *Robot motion planning*, Kluwer Academic Publishers, Norwell, MA 1991.

[85] LAWLER E.L., LENSTRA J.K., RINOOY KAN A.H.G., SHMYS D.B., *The traveling salesman problem: a guided tour of combinatorial optimization*, John Wiley and Sons 1985.

[86] LAWLER E.L., *Recent results in the theory of maschine scheduling*, [w:] Mathematical programming,the state of the art (red.: Bachem A., Grotschel M., Korte B.), Springer Verlag, Berlin 1983, s. 202–234.

[87] MICHALEWICZ Z., *Genetic algorithms+Data structures=Evolution programs*, Springer Verlag, Berlin 1994.

[88] MIELCAREK J., *Adaptacyjna obsługa zgłoszeń w systemach operacyjnych czasu rzeczywistego* (rozprawa doktorska), Raport Instytutu Cybernetyki Technicznej Politechniki Wrocławskiej, nr K–554/77, Wrocław 1977.

[89] NOWICKI E., *Czasowo-optymalne sterowanie kompleksem dynamicznych operacji niezależnych* (rozprawa doktorska), Raport Instytutu Cybernetyki Technicznej Politechniki Wrocławskiej, nr K–56/77, Wrocław 1977.

[90] ORSKI D., *Analiza i podejmowanie decyzji w systemie ekspertowym z hybrydową reprezentacją wiedzy* (rozprawa doktorska), Raport Instytutu Sterowania i Techniki Systemów Politechniki Wrocławskiej, seria PRE 2/2000, Wrocław 2000.

[91] OSOWSKI S., *Sieci neuronowe w ujęciu algorytmicznym*, Warszawa, WNT, 1996.

[92] PAUL R.P., *Robot manipulators: mathematics, programming and control*, Cambridge, MIT 1983.

[93] PAWLAK Z., *Rough sets – theoretical aspects of reasoning about data*, Kluwer Academic Publishers, Boston, Dordrecht 1991.

[94] PEDRYCZ W., *Fuzzy control and fuzzy systems*, John Wiley, NY 1993.

[95] PRZYBYŁA J., *Czasowo-optymalne sterowanie kompleksem operacji zależnych*, (rozprawa doktorska), Raport Instytutu Cybernetyki Technicznej Politechniki Wrocławskiej, nr K–196/74, Wrocław 1974.

[96] PUSZ G., *Sterowanie ruchem obiektów w kompleksie operacji* (rozprawa doktorska), Raport Instytutu Sterowania i Techniki Systemów Politechniki Wrocławskiej, seria PRE 3/91, Wrocław 1991.

[97] PUSZ G., *Two-criteria analysis of the flexible manufacturing system*, Systems Science, vol. 18, no. 1–2, 1992, s. 61–70.

[98] RAFF S.J. (red.), *Routing and scheduling of vehicles and crews: the state of the art.*, International Journal Computers and Operation Research, vol. 10, no. 2, 1983, s. 63–211.

[99] RAO C.R., *Modele liniowe statystyki matematycznej*, Warszawa, PWN, 1982.

[100] RASIOWA H., *Wstęp do matematyki współczesnej*, Warszawa, PWN, 1973.

[101] ROSENBLATT .F, *On the convergence of reinforcement procedures in simple perceptrons*, Cornell Aeronautical Laboratoty Report VG-1196-G-4, Buffalo, NY 1960.

[102] RUTKOWSKA D., PLIŃSKI M., RUTKOWSKI L., *Sieci neuronowe, algorytmy genetyczne i systemy rozmyte*, PWN, Warszawa–Łódź 1997.

[103] SAWIK T., *Optymalizacja dyskretna w elastycznych systemach produkcyjnych*, WNT, Warszawa 1992.

[104] SAWIK T., *Planowanie i sterowanie produkcji w elastycznych systemach montażowych*, Warszawa, WNT, 1996.

[105] SKALMIERSKI B., *Mechanika*, PWN, Warszawa, 1977.

[106] SRINIVAS M., PATNAIK M., *Adaptive probabilities of crossover and mutation in genetic algorithms*, IEEE Transactions on Systems Man and Cybernetics, vol. 24, no. 4, 1994, s. 656–666.

[107] STERN H.I., VITNER G., *Scheduling parts in a combined production–transportation workcell*, Journal of the Operational Research Society, vol. 41, no. 7, 1990.

[108] TADEUSIEWICZ R., *Sieci neuronowe–przewodnik problemowy*, Elektrotechnika, t. 10, z. 2, 1991, s. 125–167.

[109] TADEUSIEWICZ R., *Sieci neuronowe*, Akademicka Oficyna Wydawnicza, Warszawa 1993.

[110] TADEUSIEWICZ R., *Wokół problemów uczenia w sieciach neuronowych*, [w:] Problemy Automatyki i Informatyki, Ossolineum, Wrocław 1998, s. 201–224.

[111] TAHA H.I., *Operations research. An introduction*, Prentice Hall International Editions, 1995.

[112] TERANO T., ASAI K., SUGENO M., *Fuzzy systems theory and its applications*, Academic Press, London 1992.

[113] WAGNER H.M., *Badania operacyjne*, PWE, Warszawa 1980.

[114] WĘGLARZ J., *Application of the convex sets theory in a certain problem of time-optimal control of a complex of operations*, Systems Science, vol. 1, no. 1, 1975.

[115] WĘGLARZ J., *Time-optimal control of resource allocation in a complex of operations framework*, IEEE Trans. Systems, Man and Cybernetics, vol. 6, no. 11, 1976.

[116] YAGER J., RYAN M., POWER J., *Using fuzzy logic*, Prentice Hall, London 1994.

[117] ZADEH L.A., *Fuzzy sets*, Information and Control, vol. 8, 1965, s. 338–353.

[118] ZAREMBA M.B., JĘDRZEJEK K., BANASZAK Z.A., *Design of steady state behavior of concurrent repetitive processes*: *an algebraic approach*, IEEE Trans. Systems Man Cybernetics, vol. 28, Part A, no. 2, 1998, s. 199–212.

[119] ZITEK F., *Stracony czas – elementy teorii obsługi masowej*, Warszawa, PWN, 1973.