POLSKA AKADEMIA NAUK
Instytut Badań Systemowych

ANALIZA PRZYPADKU ŚREDNIEGO W OPTYMALIZACJI DYSKRETNEJ

Wielowymiarowe zadanie załadunku oraz zadanie szeregowania prac

Krzysztof Szkatuła

Krzysztof SZKATUŁA - ANALIZA PRZYPADKU ŚREDNIEGO W OPTYMALIZACJI DYSKRETNEJ

ANALIZA PRZYPADKU ŚREDNIEGO W OPTYMALIZACJI DYSKRETNEJ

Polska Akademia Nauk • Instytut Badań Systemowych

Seria: BADANIA SYSTEMOWE
tom 23

Redaktor naukowy:

Prof. dr hab. Jakub Gutenbaum

Warszawa 1999

Krzysztof SZKATUŁA

ANALIZA PRZYPADKU ŚREDNIEGO W OPTYMALIZACJI DYSKRETNEJ

Wielowymiarowe zadanie załadunku oraz zadanie szeregowania prac

Publikację opiniowali do druku:

Prof. dr hab. Juliusz L. Kulikowski
Dr hab. Włodzimierz Ogryczak

ISBN 83-85847-39-1
ISSN 0208-8029

Wprowadzenie

Optymalizacja dyskretna stała się samodzielną dziedziną badawczą od połowy lat pięćdziesiątych dwudziestego wieku. Powstała ona na styku zastosowań praktycznych w dziedzinach takich jak ekonomia, zarządzanie, technika i wiele innych oraz matematyki ze szczególnym uwzględnieniem kombinatoryki, teorii grafów i logiki matematycznej. W pewnym uproszczeniu można powiedzieć, że podstawowym celem optymalizacji dyskretnej jest wybór optymalnego wariantu ze skończonego lub przeliczalnego ich zbioru. Optymalność jest rozumiana jako wyznaczenie maksimum lub minimum pewnej funkcji. Uzyskanie rozwiązania zadania optymalizacji dyskretnej umożliwia podejmowanie trafnych decyzji w odniesieniu do wielu aspektów działalności ludzkiej. Przykładami kryteriów optymalizacyjnych może być maksymalizacja zysków, minimalizacja kosztów lub strat i wiele innych.

Można powiedzieć, że w zaawansowanych zastosowaniach praktycznych pojawiło się wiele ważnych, złożonych i trudnych do rozwiązania problemów o powyższym charakterze. Do ich sformalizowania w najbardziej odpowiedni sposób bardzo przydatny okazał się aparat i metodologia matematyki.

W momencie gdy zagadnienie praktyczne zostało sformalizowane jako optymalizacyjny problem matematyczny, powstaje potrzeba jego efektywnego rozwiązania. Oznacza to konieczność opracowania algorytmów i wykonania ich komputerowej implementacji. Niestety, okazało się, że w przypadku wielu zadań optymalizacji dyskretnej oraz algorytmów opracowanych do ich rozwiązywania pojawia się zasadnicza trudność. Dla nawet niezbyt dużych przykładów tych zadań obliczenia numeryczne wymagają niemożliwego do zaakceptowania nakładu obliczeń, na przykład mierzonego w stuleciach pracy obecnych superkomputerów. Co więcej, nawet drastyczne zwiększenie wydajności komputerów nie jest w stanie zasadniczo poprawić sytuacji. Dla ustalenia uwagi, 100-krotne przyśpieszenie obliczeń zmniejsza nakład obliczeń ze 100 lat do jednego roku, co wciąż jest wielkością abstrakcyjną, niemożliwą do zaakceptowania w praktyce obliczeniowej. Efektem tej sytuacji był rozwój teorii i praktycznego zastosowania algorytmów przybliżonych, których celem jest wyznaczenie przybliżonego rozwiązania zadania o akceptowalnej jakości,

przy "niewielkim" nakładzie obliczeń.

Praktyczna niemożliwość uzyskania rozwiązań optymalnych dla licznych przykładów zadań optymalizacji dyskretnej spowodowała konieczność analizowania nakładu obliczeń wymaganego przez algorytmy, dla zadań o określonej wielkości.

W efekcie powstała dziedzina badawcza zwana złożonością obliczeniową. Jej podstawowym zadaniem jest analizowanie zadań i algorytmów optymalizacji dyskretnej z punktu widzenia oceny nakładu obliczeń niezbędnego do uzyskania rozwiązania optymalnego jako funkcji rozmiaru, wielkości zadania. Zadania optymalizacji dyskretnej zostały umownie podzielone na łatwe i trudne do rozwiązywania.

Należy zwrócić uwagę na fakt, że uzyskane w ten sposób oceny noszą charakter absolutnej gwarancji, to znaczy, że są one prawdziwe dla wszystkich realizacji danych analizowanego zadania. Analiza taka nosi nazwę analizy przypadku najgorszego, gdyż oparta jest ona na najbardziej niekorzystnym zachowaniu się zadań i algorytmów optymalizacji dyskretnej.

Praktyka rozwiązywania zadań optymalizacji dyskretnej wykazała jednak szybko, że uzyskane w ten sposób oceny są bardzo często nadmiernie pesymistyczne. Oceny uzyskane w oparciu o podejście przypadku najgorszego nie charakteryzują w sposób właściwy przeciętnego, średniego zachowania się zadań i algorytmów optymalizacji dyskretnej.

Ta sytuacja spowodowała powstanie dziedziny nazywanej analizą przypadku średniego lub inaczej analizą probabilistyczną zadań i algorytmów optymalizacji dyskretnej. Przeprowadzenie analizy przypadku średniego wymaga zdefniowania losowego modelu rozważanego zadania. Uzyskane w efekcie przeprowadzenia analizy przypadku średniego wyniki odnoszą się tylko do rozważanego losowego modelu zadania optymalizacji dyskretnej. Natomiast ich olbrzymią zaletą jest możliwość uzyskania alternatywnych, w stosunku do złożoności obliczeniowej w przypadku najgorszym, ocen zachowania się zadań i algorytmów optymalizacji dyskretnej.

Ważną i oryginalną właściwością analizy przypadku średniego jest możliwość analizowania innych charakterystyk zadania niż złożoność obliczeniowa. Dobrym przykładem jest asymptotyczna ocena zachowania się wartości rozwiązania optymalnego jako funkcji pewnych parametrów zadania. Wyniki tego typu znacznie poszerzają wiedzę na temat zadań optymalizacji dyskretnej i w efekcie umożliwiają ich rozwiązywanie w znacznie bardziej efektywny sposób.

Zasadniczym celem tej monografii jest wykazanie, na przykładzie wybranych, ważnych zadań optymalizacji dyskretnej, że przy zastosowaniu prostego aparatu rachunku prawdopodobieństwa można uzyskać wartościowe wyniki analizy przypadku średniego.

W celu właściwego osadzenia uzyskanych wyników w teorii i praktyce optymalizacji dyskretnej pierwsze rozdziały monografii poświęcone zostały prezentacji wybranych zadań optymalizacji dyskretnej, metod ich rozwiązywania, złożoności obliczeniowej oraz analizy przypadku średniego. Należy jednak podkreślić, że zamiarem autora była poglądowa, a nie bardzo szczegółowa i wyczerpująca prezentacja tych zagadnień. Szczegółowy plan monografii jest następujący.

W rozdziale 1 zaprezentowano dziedzinę optymalizacji dyskretnej ze szczególnym uwzględnieniem programowania całkowitoliczbowego i liniowego, zadań binarnych, zadań teorii grafów oraz zadań harmonogramowania.

W rozdziale 2 zostały przedstawione znane metody rozwiązywania zadań optymalizacji dyskretnej, takie jak metoda podziału i oszacowań, programowanie dynamiczne, algorytmy zachłanne, metody lokalnej poprawy oraz algorytmy programowania linowego.

W rozdziale 3 rozważono podejście przypadku najgorszego do oceny złożoności obliczeniowej zadań i algorytmów optymalizacji dyskretnej. Zdefiniowano niezbędne elementy oceny rozmiaru wielkości danych zadania i nakładu obliczeń, wprowadzono klasy złożoności obliczeniowej.

Podejście przypadku średniego zastało przedstawione w rozdziale 4. Szczególną uwagę poświęcono aparatowi probablistycznemu niezbędnemu w dalszej części monografii oraz prezentacji wybranych wyników analizy przypadku średniego znanych z literatury.

W rozdziałach 5 oraz 6 zaprezentowano wielowymiarowe zadanie załadunku oraz zadanie szeregowania prac z terminami zakończenia. Na przykładzie tych ważnych zadań optymalizacji dyskretnej dokonano dogłębnej analizy zagadnień będących przedmiotem zainteresowania niniejszej monografii, takich jak metody rozwiązywania, analiza złożoności obliczeniowej oraz analiza przypadku średniego. Przedstawione zostały oryginalne wyniki opisujące asymptotyczne zachowanie się rozwiązań optymalnych jako funkcji parametrów zadań w przypadku średnim. Wykazano również, że proste algorytmy przybliżone mogą być asymptotycznie optymalne w sensie analizy przypadku średniego.

Zamiarem autora było używanie możliwie najprostszej notacji matematycznej dla zachowania maksymalnej przejrzystości wywodu. W monografii stosowane są powszechnie przyjęte oznaczenia matematyczne. Dla przykładu:

- $\{a_1, a_2, \ldots, a_n\}$ oznacza zbiór n - elementowy.
- $[x_1, x_2, \ldots, x_m]$ oznacza wektor o m składowych przyjmujących wartości liczbowe.

- $\lim_{x \to a} f(x)$ lub $\lim_{n \to a} g_n$ oznacza granicę funkcji lub ciągu liczbowego, co jednoznacznie wynika z kontekstu.

- $\{X\}$ oznacza jednoznacznie zdefiniowane zdarzenie, na przykład $x < b$, gdzie x jest zmienną, b pewną stałą.

- $|a|$ oznacza wartość bezwzględną liczby a, natomiast $|A|$ oznacza moc (liczbę elementów) zbioru A.

Kolejne oznaczenia są definiowane w tekście monografii w miarę potrzeb. W przypadku możliwości wystąpienia niejednoznaczności w trakcie przeprowadzania wywodów, odpowiednie pojęcia są przytaczane na bieżąco.

Oryginalna terminologia opisująca pojęcia i obiekty rozważane w monografii w przytłaczającej większości pochodzi z języka angielskiego. W monografii wykorzystywana jest polska terminolagia, która zdaniem autora, z jednej strony właściwie oddaje sens oryginału angielskiego, z drugiej zaś strony jest dobrze osadzona w języku polskim. Poza przypadkami oczywistymi, w momencie pierwszego zastosowania nowego, specjalistycznego pojęcia w nawiasach przytoczono jego angielski odpowiednik.

Niniejsza monografia jest efektem wieloletniego zainteresowania autora tematyką analizy przypadku średniego wybranych zadań optymalizacji dyskretnej. Pracę naukową nad tymi zagadnieniami autor prowadził w Instytucie Badań Systemowych Polskiej Akademii Nauk kolejno w Zakładzie Programowania Matematycznego, Pionie Metod Modelowania Matematycznego i Optymalizacji oraz w Pracowni Metod Obliczeniowych Optymalizacji.

Pragnę serdecznie podziękować wszystkim koleżankom i kolegom z IBS PAN za wieloletnią współpracę. Przez cały okres mojej pracy naukowej szczególne znaczenie miała dla mnie współpraca z doc. dr hab. Markiem Liburą, na którego pomoc i cenne rady mogłem zawsze liczyć.

Swojej rodzinie składam wyrazy wdzięczności za cierpliwość, wyrozumiałość i wsparcie podczas całego okresu mojej pracy naukowej. Szczególnie gorąco pragnę podziękować mojej Żonie i Mamie.

Rozdział 1

Optymalizacja dyskretna

1.1 Wprowadzenie

Przedmiotem zainteresowania dziedziny nazywanej *optymalizacją dyskretną* (*discrete optimization*), patrz Nemhauser i Wolsey [110], jest wyznaczenie wartości największej lub najmniejszej funkcji wielu zmiennych, zwanej *funkcją celu*, przy istnieniu dwóch rodzajów *ograniczeń*:

- ograniczeń równościowych bądź nierównościowych;
- ograniczeń wymuszających przyjęcie przez wszystkie lub część zmiennych wartości całkowitych.

Inna definicja zakłada, patrz Grötschel i Lovász [62], że celem optymalizacji dyskretnej jest wyznaczenie maksimum bądź minimum globalnego funkcji celu na skończonym lub przeliczalnym zbiorze *rozwiązań dopuszczalnych*.

Należy zauważyć, że definicje te nie są sobie tożsame, gdyż możliwe jest konstruowanie bardzo szczególnych kontrprzykładów. Jednak praktycznie wszystkie znane zadania optymalizacji dyskretnej są zgodne z obydwiema powyższymi definicjami.

Ogólnie można stwierdzić, że brak jest jednej, uniwersalnej i wyłącznej definicji dziedziny badawczej zwanej optymalizacją dyskretną, patrz również Papadimitriou i Steiglitz [111] oraz Parker i Riardin [112]. Znacznie większa zgodność panuje przy kwalifikowaniu konkretnych zadań do dziedziny optymalizacji dyskretnej. Wiele klasycznych zadań optymalizacji dyskretnej zostanie przedstawionych w dalszej części tego rozdziału.

Praktycznie zamiennie do nazwy optymalizacja dyskretna mogą być również stosowane w literaturze określenia: *optymalizacja kombinatoryczna lub*

optymalizacja całkowitoliczbowa (*combinatorial optimization, integer optimization*). W dalszej części niniejszej monografii będzie stosowane określenie optymalizacja dyskretna.

Z uwagi na uniwersalność swojego sformułowania optymalizacja dyskretna obejmuje znaczną liczbę zadań ważnych dla bardzo różnych obszarów teorii i praktyki. Z punktu widzenia matematyki optymalizacja dyskretna ma ścisłe powiązania między innymi z kombinatoryką, teorią grafów oraz logiką. Historia optymalizacji dyskretnej jako samodzielnej dziedziny badawczej liczy około 40 lat i w swoich początkach związana była z zagadnieniami ekonomicznymi, takimi jak planowanie i zarządzanie operacjami, efektywnym wykorzystaniem zasobów itp. Następnie pojawiły się zastosowania bardziej techniczne, takie jak zadania szeregowania prac wykonywanych na maszynach, harmonogramowanie procesów produkcji, projektowanie automatycznych procesów wytwarzania i wiele innych.

Obecnie zadania optymalizacji dyskretnej występują w bardzo wielu dziedzinach ludzkiej działalności takich jak na przykład: *alokacja kapitału* (*capital budgeting*), *wybór portfela inwestycji* (*portfolio selection*), projektowanie kampanii rynkowych, planowanie inwestycji i lokalizacja obiektów, określanie okręgów wyborczych, zagadnienia genetyczne, klasyfikacja roślin i zwierząt, projektowanie układów sieci komunikacyjnych, pozycjonowanie satelitów, projektowanie i produkcja układów VLSI oraz płytek drukowanych w elektronice, optymalizacja ładunków i planowanie transportu, harmonogramowanie rozkładów jazdy autobusów, pociągów oraz lotów samolotów, przydział pracowników do pracy (kierowcy autobusów, załogi samolotów, pociągów, itp), konstruowanie niemożliwych do złamania kodów w kryptografii i wiele, wiele innych. Lista zastosowań wydaje się nieograniczona, nawet w dziedzinach takich jak sport, archeologia lub psychologia optymalizacja dyskretna jest stosowana w celu uzyskania odpowiedzi na ważne pytania.

Istnieją różne podejścia do prezentacji dziedziny optymalizacji dyskretnej. Jednym z możliwych podejść jest opis zadań i ich właściwości. Dla zastosowań praktycznych bardzo ważnym zagadnieniem może być budowa i analiza metod rozwiązywania zadań optymalizacji dyskretnej.

W dalszej części tego rozdziału zostanie dokonany przegląd wybranych zadań optymalizacji dyskretnej. Natomiast w rozdziale 2 zaprezentowane zostaną niektóre metody służące do ich rozwiązywania.

Dla praktycznego rozwiązywania zadań optymalizacji dyskretnej szczególnie istotna jest złożoność zadań i algorytmów optymalizacji dyskretnej. W rozdziale 3 zostanie ona omówiona z punktu widzenia tak zwanego przypadku najgorszego, natomiast w rozdziale 4 w oparciu o podejście tak zwanego przypadku średniego.

1.2 Programowanie całkowitoliczbowe

Tak jak w przypadku większości prac poświęconych tematyce optymalizacji dyskretnej, zgodnie z pierwszą z przytoczonych powyżej definicji dziedziny optymalizacji dyskretnej, przyjmujemy, że *funkcja celu* jak również *ograniczenia* są reprezentowane jako funkcje liniowe. Biorąc pod uwagę fakt, że każde ograniczenie równościowe może być przedstawione jako dwa ograniczenia nierównościowe, istnieje możliwość rozważania zadań posiadających wyłącznie ograniczenia nierównościowe. Jest również dość powszechnie przyjętą praktyką rozważanie tylko nieujemnych wartości zmiennych. Przy przyjęciu powyższych założeń możemy sformułować *zadanie liniowego mieszanego programowania całkowitoliczbowego* (*mixed integer programming*) z kryterium maksymalizacji funkcji celu w poniższej postaci:

$$\begin{aligned} & z_{OPT}(n) = \max\left\{\sum_{i=1}^{n} c_i \cdot x_i + \sum_{l=1}^{p} h_l \cdot y_l\right\} \\ & \text{przy ograniczeniach } \sum_{i=1}^{n} a_{ji} \cdot x_i + \sum_{l=1}^{p} g_{jl} \cdot y_l \leq b_j; \quad j = 1, ..., m, \\ & \text{gdzie } \quad x_i, y_l \geq 0,\ x_i \text{ - całkowite},\ i = 1, ..., n,\ l = 1, ..., p. \end{aligned} \tag{1.1}$$

W dalszych rozważaniach przyjmujemy założenie, że współczynniki zadania liniowego mieszanego programowania całkowitoliczbowego (1.1), to znaczy c_i, h_l, a_{ji}, g_{jl} oraz b_j są liczbami wymiernymi, gdzie $i = 1, \ldots, n$, $l = 1, \ldots, p$, $j = 1, \ldots, m$. Każdy zbiór współczynników zadania o ustalonych wartościach liczbowych będziemy nazywać *realizacją danych* zadania. Zauważmy, że takie sformułowanie zadania w możliwie najmniejszym stopniu ogranicza ogólność rozważań. Zadanie z kryterium minimalizacji jest tożsame z zadaniem maksymalizacji tej samej funkcji celu z przeciwnymi znakami, to znaczy, że c_i, h_l są zastąpione przez $-c_i, -h_l$. Każde ograniczenie równościowe

$$\sum_{i=1}^{n} a_{j^*i} \cdot x_i + \sum_{l=1}^{p} g_{j^*l} \cdot y_l = b_{j^*}$$

może zostać zastąpione przez dwa ograniczenia nierównościowe o następującej postaci

$$\begin{aligned} \sum_{i=1}^{n} a_{j'i} \cdot x_i + \sum_{l=1}^{p} g_{j'l} \cdot y_l &\leq b_{j'} \\ \sum_{i=1}^{n} a_{j''i} \cdot x_i + \sum_{l=1}^{p} g_{j''l} \cdot y_l &\leq b_{j''} \end{aligned}$$

gdzie $a_{j'i} = -a_{j''i} = a_{j^*i}$, $g_{j'l} = -g_{j''l} = g_{j^*l}$ oraz $b_{j'} = -b_{j''} = b_{j^*}$. Niech

$$X = \{[x_1, \ldots, x_n] : x_i \geq 0, x_i - \text{całkowite}\} \text{ oraz } Y = \{[y_1, \ldots, y_p] : y_l \geq 0\}.$$

Zbiór wszystkich wektorów $[x, y]$, gdzie $x \in X$, $y \in Y$, spełniających ograniczenia zadania liniowego mieszanego programowania całkowitoliczbowego (1.1):

$$S = \left\{ [x, y] : \sum_{i=1}^{n} a_{ji} \cdot x_i + \sum_{l=1}^{p} g_{jl} \cdot y_l \leq b_j \quad \text{dla wszystkich } j = 1, \ldots, m \right\},$$

będziemy nazywać *zbiorem rozwiązań dopuszczalnych* zadania (1.1). Natomiast każdy wektor o postaci $[x, y]$ spełniający ograniczenia zadania (1.1), to znaczy $[x, y] \in S$, będziemy nazywać *rozwiązaniem dopuszczalnym* zadania (1.1). Zmienne x_i, y_l będziemy nazywać *zmiennymi decyzyjnymi*, ponadto x_i, $i = 1, \ldots, n$ będziemy nazywać *zmiennymi całkowitoliczbowymi*, a y_l, $l = 1, \ldots, p$ *zmiennymi ciągłymi*. Konkretną realizację danych zadania liniowego mieszanego programowania całkowitoliczbowego (1.1) będziemy uważać za dopuszczalną, jeżeli $S \neq \emptyset$. Funkcję

$$z_{OPT}(n) = \max \left\{ \sum_{i=1}^{n} c_i \cdot x_i + \sum_{l=1}^{p} h_l \cdot y_l \right\}$$

będziemy nazywać *funkcją celu* zadania liniowego mieszanego programowania całkowitoliczbowego (1.1). Rozwiązanie dopuszczalne $[x^*, y^*]$, dla którego funkcja celu przyjmuje największą z możliwych wartości, to znaczy

$$\sum_{i=1}^{n} c_i \cdot x_i^* + \sum_{l=1}^{p} h_l \cdot y_l^* \geq \sum_{i=1}^{n} c_i \cdot x_i + \sum_{l=1}^{p} h_l \cdot y_l, \text{ dla każdego } [x, y] \in S$$

będziemy nazywać *rozwiązaniem optymalnym* zadania liniowego mieszanego programowania całkowitoliczbowego (1.1).

Może istnieć taki przypadek zadania liniowego mieszanego programowania całkowitoliczbowego (1.1), który ma rozwiązania dopuszczalne ale nie posiada rozwiązania optymalnego. Będziemy mówić, że realizacja zadania liniowego mieszanego programowania całkowitoliczbowego (1.1) jest *nieograniczona*, jeżeli dla dowolnej stałej rzeczywistej δ istnieje $[x, y] \in S$ taki, że

$$\sum_{i=1}^{n} c_i \cdot x_i + \sum_{l=1}^{p} h_l \cdot y_l > \delta.$$

W przypadku zadań nieograniczonych będziemy pisać $z_{OPT}(n) = \infty$.

W przypadku współczynników zadania będących liczbami wymiernymi, każda dopuszczalna realizacja danych zadania liniowego mieszanego programowania całkowitoliczbowego (1.1) albo ma rozwiązanie optymalne, albo jest nieograniczona.

Jeżeli w realizacji danych zadania wystąpiły współczynniki o wartościach niewymiernych, to istnieje możliwość wystąpienia przypadku, że żadne z rozwiązań dopuszczalnych zadania nie może osiągnąć najmniejszej wartości oszacowania od góry rozwiązania optymalnego. Oznacza to, że zadanie ma rozwiązania dopuszczalne, nie jest również niegraniczone, ale równocześnie nie istnieje rozwiązanie optymalne dla pewnych realizacji danych zadania (1.1), patrz Nemhauser i Wolsey [110].

Aby rozwiązać realizację danych zadania liniowego mieszanego programowania całkowitoliczbowego (1.1) należy więc albo wyznaczyć jej rozwiązanie optymalne albo wykazać niedopuszczalność lub niegraniczoność danej realizacji zadania.

Zadanie liniowego mieszanego programowania całkowitoliczbowego (1.1) zawiera w swoim opisie dwie bardzo ważne klasy zadań: zadanie programowania całkowitoliczbowego i zadanie programowania liniowego, patrz Zorychta i Ogryczak [151].

Zadanie *programowania całkowitoliczbowego* przyjmuje postać:

$$\begin{aligned} & z_{OPT}(n) = \max \sum_{i=1}^{n} c_i \cdot x_i \\ & \text{przy ograniczeniach } \sum_{i=1}^{n} a_{ji} \cdot x_i \leq b_j; \quad j = 1, ..., m, \\ & \text{gdzie } \; x_i \geq 0, \; x_i \text{ - całkowite}, \; i = 1, ..., n. \end{aligned} \tag{1.2}$$

Zadanie programowania całkowitoliczbowego może być interpretowane jako szczególna postać zadania liniowego mieszanego programowania całkowitoliczbowego (1.1) bez zmiennych ciągłych.

Zadanie *programowania linowego* można zapisać w postaci:

$$\begin{aligned} & z_{OPT}(n) = \max \sum_{l=1}^{p} h_l \cdot y_l \\ & \text{przy ograniczeniach } \sum_{l=1}^{p} g_{jl} \cdot y_l \leq b_j; \quad j = 1, ..., m \\ & \text{gdzie } \; y_l \geq 0, \quad l = 1, ..., p. \end{aligned} \tag{1.3}$$

Zadanie programowania liniowego może być rozważane jako szczególna postać zadania liniowego mieszanego programowania całkowitoliczbowego (1.1) bez zmiennych całkowitoliczbowych.

Ogólnie można powiedzieć, że wystąpienie zmiennych całkowitoliczbowych w sformułowaniu zadania związane jest z dokonaniem wyboru spośród

policzalnych i niepodzielnych obiektów, natomiast zmienne ciągłe reprezentują takie zasoby, które są mierzalne, ale nie muszą, bądź nie mogą, być rozpatrywane jako podzielne, skwantowane. Do grupy obiektów policzalnych można na przykład zaliczyć osoby, pracowników, zwierzęta, maszyny, samochody, samoloty, ogólniej rzecz ujmując są to obiekty ze swojej natury przeliczalne, a więc ich liczba jest wyrażana w liczbach całkowitych. Natomiast do grupy obiektów mierzalnych zalicza się zasoby typu waga, rozmiar, objętość, czas i inne, które są mierzone w liczbach wymiernych, a nie koniecznie w liczbach całkowitych.

Na przykład zespół ludzki (instytucja, drużyna sportowa lub jednostka wojskowa) jest w większości przypadków charakteryzowany przez swoją liczebność, a nie wagę lub objętość. Natomiast transport pszenicy lub żużlu jest na ogół oceniany przez swoją wagę lub objętość, a nie, na przykład, przez liczbę ziaren czy grudek. Oczywiście podziały te są do pewnego stopnia umowne i często przy przydziale pracowników do zadań stosuje się "podział" typu 0.7 i 0.3, zaś wagę, rozmiar lub czas można mierzyć z zaokrągleniem do pewnych niezbyt dużych jednostek takich jak na przykład gramy, milimetry bądź sekundy i wtedy wyniki pomiarów będą zawsze liczbami całkowitymi. Tak więc przy modelowaniu zagadnień zawsze należy kierować się zdrowym rozsądkiem i posiadaną wiedzą o "rzeczywistym", istotnym charakterze modelowanych obiektów lub zjawisk.

W ogólnym przypadku należy stwierdzić, że zgodnie z definicją zadanie programowania liniowego nie należy do zadań optymalizacji dyskretnej, gdyż nie występuje w jego definicji warunek całkowitoliczbowości zmiennych, a zbiór rozwiązań dopuszczalnych jest przeliczalny tylko w bardzo szczególnych przypadkach. Natomiast bardzo wiele fundamentalnych zadań optymalizacji dyskretnej może zostać, w mniej lub bardziej naturalny sposób, sprowadzonych do zadań programowania liniowego. Dlatego możliwość efektywnego rozwiązywania zadań programowania liniowego (1.3) jest bardzo ważna dla teorii i praktyki optymalizacji dyskretnej. Zagadnienia te zostaną omówione w rozdziale 2.

Jak już wspomniano, nie istnieje uniwersalna i jednolita definicja optymalizacji dyskretnej, ale istotna część zadań powszechnie zaliczanych do tej dziedziny może zostać zapisana w postaci zadania binarnego, zdefiniowanego na zbiorach skończonych i ich podzbiorach.

Poniżej zostanie przedstawiona jedna z możliwych, ogólnych definicji zadania optymalizacji dyskretnej. Niech

$$Q = \{1, 2, \ldots, q-1, q\},\ c = [c_1, \ldots, c_q],\ z(F) = \sum_{i \in F} c_i,\ F \in \mathcal{F}, \qquad (1.4)$$

gdzie Q jest skończonym zbiorem kolejnych liczb całkowitych (indeksów),

składowe wektora $c = [c_1, \ldots, c_q]$ są pewnymi liczbami wymiernymi, F jest podzbiorem zbioru Q, $F \subseteq Q$, oraz $\mathcal{F}$ jest pewnym zbiorem podzbiorów zbioru Q.

Zadanie *optymalizacji dyskretnej* może zostać zdefiniowane w postaci:

$$z_{OPT}(q) = \max_{F \in \mathcal{F}} z(F) \tag{1.5}$$

1.3 Zadania binarne

W wielu praktycznych zagadnieniach zmienne całkowitoliczbowe są wykorzystywane do przedstawienia zależności logicznych (wyboru alternatywnego: *tak* lub *nie)* i w związku z tym przyjmowane przez nie wartości są ograniczone tylko do dwóch wartości: 0 lub 1. Taki szczególny przypadek zmiennych całkowitoliczbowych nazywamy *zmiennymi binarnymi.*

Ważnym zastosowaniem zadań sformułowanych z wykorzystaniem zmiennych binarnych jest model sytuacji rzeczywistej, gdzie występuje wybór uwarunkowany wystąpieniem lub niewystąpieniem pewnego zjawiska, podjęcie decyzji na *tak* lub na *nie.*

Dla modelowania takiego wyboru można użyć *zmiennej binarnej* x, gdzie

$$x = \begin{cases} 1 & \text{jeżeli zjawisko wystąpi, decyzja na tak} \\ 0 & \text{jeżeli zjawisko nie wystąpi, decyzja na nie.} \end{cases}$$

Zjawisko lub decyzja mogą być dowolnej natury, w zależności od modelowanego obiektu lub zdarzenia.

Zadania programowania całkowitoliczbowego w postaci (1.2), w których warunek x_i - całkowitoliczbowe został zastąpiony warunkiem x_i= 0 lub 1, $i = 1, \ldots, n$, są często nazywane *zadaniami programowania binarnego.* W ogólnym przypadku w stosunku do zadań optymalizacji dyskretnej, do sformułowania których można zastosować tylko zmienne binarne, będziemy stosowali nazwę *zadania binarne.*

W bardzo wielu przypadkach możliwe jest zastąpienie zadań całkowitoliczbowych przez binarne. Wynika to z faktu, że zmienne całkowitoliczbowe y mogą zostać zastąpione przez pewną liczbę zmiennych binarnych. Istnieje jednoznaczny zapis każdej liczby całkowitej y z wykorzystaniem liczb binarnych w następującej postaci:

$$y = \sum_{i=0}^{k} x_i \cdot 2^i \text{ gdzie } x_i = 0 \text{ lub } 1, \; k \leq \log_2 y < k + 1.$$

Zauważmy, że taka transformacja oznacza zwiększenie rozmiaru zadania, ale z drugiej strony może okazać się korzystna w procesie jego rozwiązywania.

W dalszej części tego podrozdziału przedstawimy szereg przykładów zadań binarnych będących klasycznymi zadaniami optymalizacji dyskretnej.

Binarne zadanie załadunku

Binarne zadanie załadunku jest szczególną postacią zadania programowania binarnego lub ogólniej całkowitoliczbowego (1.2) i może być sformułowane w następującej postaci:

$$z_{OPT}(n) = \max\left\{\sum_{i=1}^{n} c_i \cdot x_i : \sum_{i=1}^{n} a_{ji} \cdot x_i \leq b_j(n),\ x_i = 0 \text{ lub } 1,\ j = 1, ..., m\right\}$$

Od ogólnego sformułowania zadania (1.2) zadanie załadunku różni wymóg nieujemności współczynników zadania, $c_i, a_{ji}, b_j \geq 0$, oraz binarności zmiennych decyzyjnych, $x_i = 0$ lub 1. Dla $m = 1$ klasyczne *jednowymiarowe zadanie załadunku* przyjmuje poniższą postać:

$$z_{OPT}(n) = \max\left\{\sum_{i=1}^{n} c_i \cdot x_i : \sum_{i=1}^{n} a_i \cdot x_i \leq b(n),\ x_i = 0 \text{ lub } 1\right\}. \qquad (1.6)$$

Dla $m \geq 2$ otrzymujemy *wielowymiarowe zadania załadunku.* Zadaniom załadunku poświęcono w niniejszej monografii szczególną uwagę, patrz rozdziały 5 i 6. Na ich przykładzie dokonano szczegółowej analizy zagadnień będących obiektem zainteresowania niniejszej monografii.

Zadania przydziału i skojarzenia

Innym dobrze znanym zadaniem optymalizacji dyskretnej jest zadanie przydziału osób do prac. Załóżmy, że mamy n osób i m prac, gdzie $n \geq m$. Każda z prac musi być wykonana przez dokładnie jedną osobę, każda z osób może wykonać co najwyżej jedną z prac. Koszt wykonania przez osobę j pracy i wynosi c_{ij}. Aby sformułować zadanie optymalizacji dyskretnej zwane *zadaniem przydziału* (*assigment problem*) należy wprowadzić zmienną decyzyjną $x_{ij}, i = 1, \ldots, m, j = 1, \ldots, n$, która przyjmuje wartość 1, kiedy osoba j wykonuje pracę i oraz 0 w przypadku przeciwnym. Zadanie przydziału może

zostać sformułowane w poniższej postaci:

$$\begin{aligned} z_{OPT}(n) &= \min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} \cdot x_{ij} \\ \text{przy ograniczeniach } & \sum_{j=1}^{n} x_{ij} = 1, \text{ dla } i = 1, \ldots, m \\ & \sum_{i=1}^{m} x_{ij} \leq 1, \text{ dla } j = 1, \ldots, n \\ \text{gdzie } \quad & x_{ij} = 0 \text{ lub } 1, \; i = 1, \ldots, m, \; j = 1, \ldots, n, \end{aligned}$$

Z uwagi na fakt, że pracę i musi wykonać dokładnie jedna osoba mamy m ograniczeń o postaci $\sum_{j=1}^{n} x_{ij} = 1$ dla wszystkich $i = 1, \ldots, m$. Ponieważ każda z osób może wykonać co najwyżej jedną z prac, mamy dodatkowo n ograniczeń o postaci $\sum_{i=1}^{m} x_{ij} \leq 1$ dla wszystkich $j = 1, \ldots, n$. Funkcja celu zadania przyjmuje postać $\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} \cdot x_{ij}$, co oznacza, że celem zadania optymalizacyjnego jest dokonanie takiego przydziału osób do prac, aby zminimalizować koszty wykonania m prac. Ponadto, spełnienie $x_{ij} = 0$ dla wszystkich $i = 1, \ldots, m$ oznacza, że osobie j, $1 \leq j \leq n$, nie przydzielono żadnej pracy do wykonania.

W zadaniu przydziału mamy zbiór $n + m$ elementów podzielony na rozłączne zbiory osób i prac. Istnieje wiele sytuacji, w których taki podział jest niemożliwy. Przypuśćmy, że mamy $2n$ pracowników oraz n dwuosobowych pomieszczeń. Oceniamy, że wspólna praca w jednym pokoju pracownika i oraz j przynosi korzyść c_{ij}, $i, j = 1, \ldots, 2n$, $i \neq j$, (korzyść ta może być oceniona poprzez potrzebę i zdolność do współpracy, wzajemną sympatię itp.), przy czym $x_{ij} = 1$, jeśli pracownicy ci są przydzieleni do tego samego pomieszczenia oraz $x_{ij} = 0$ w przeciwnym przypadku. Celem *zadania doskonałego skojarzenia* (*perfect matching*) jest dokonanie takiego przydziału pracowników do pomieszczeń, aby łączna korzyść była maksymalna. Odpowiednie zadanie optymalizacji dyskretnej można sfomułować w następującej postaci

$$\begin{aligned} z_{PM}(n) &= \max \sum_{i=1}^{2n-1} \sum_{j=i+1}^{2n} c_{ij} \cdot x_{ij} \\ \text{przy ograniczeniach } & \sum_{k<i} x_{ki} + \sum_{j>i} x_{ij} = 1; \quad i = 1, \ldots, 2n \\ \text{gdzie } \quad & x_{ij} = 0 \text{ lub } 1, i \neq j, \; x_{ii} = 0, \quad i, j = 1, \ldots, 2n \end{aligned} \tag{1.7}$$

Wystąpienie ograniczenia o postaci $\sum_{k<i} x_{ki} + \sum_{j>i} x_{ij} = 1$ oznacza, że wszyscy pracownicy zostaną połączeni w pary i przydzieleni do pomieszczeń.

Natomiast, jeśli naszym celem jest wyodrębnienie takich par pracowników, których wspólna praca może przynieść maksymalną łączną korzyść, ale bez założenia, że każdemu z pracowników przydzielimy partnera, to wtedy

mamy do czynienia z *zadaniem skojarzenia* (*matching*). Wtedy ograniczenia w zadaniu (1.7) przyjmują postać $\sum_{k<i} x_{ki} + \sum_{j>i} x_{ij} \leq 1$ co oznacza, że część pracowników może nie zostać uwzględniona w przydziale partnerów do pomieszczeń. Odpowiednie zadanie optymalizacji dyskretnej przyjmuje poniższą postać

$$\begin{aligned} z_{MA}(n) &= \max \sum_{i=1}^{2n-1} \sum_{j=i+1}^{2n} c_{ij} \cdot x_{ij} \\ \text{przy ograniczeniach } & \sum_{k<i} x_{ki} + \sum_{j>i} x_{ij} \leq 1; \quad i = 1, \ldots, 2n \\ \text{gdzie } & x_{ij} = 0 \text{ lub } 1, i \neq j, \; x_{ii} = 0, \quad i, j = 1, \ldots, 2n \end{aligned} \tag{1.8}$$

Zauważmy, że rozwiązania optymalne obu zadań spełniają poniższą zależność

$$z_{PM}(n) \leq z_{MA}(n).$$

Zależność powyższa wynika wprost z faktu, że każde rozwiązanie dopuszczalne zadania skojarzenia doskonałego (1.7) jest rozwiązaniem dopuszczalnym zadania skojarzenia (1.8), ale nie odwrotnie. Funkcje celu powyższych zadań mają kryteria maksymalizacyjne. Może istnieć rozwiązanie dopuszczalne zadania skojrzenia (1.8) nie będące rozwiązaniem dopuszczalnym zadania doskonałego skojarzenia (1.7), o wartości funkcji celu większej od wartości rozwiązania optymalnego $z_{PM}(n)$ zadania doskonałego skojarzenia (1.7).

Zauważmy, że każde z powyższych zadań binarnych jest zgodne z ogólną definicją zadania optymalizacji dyskretnej (1.5). Na przykład dla zadań załadunku $Q = \{1, \ldots, n\}$ oraz

$$F \in \mathcal{F} \text{ wtedy i tylko wtedy, kiedy } \sum_{i \in F} a_{ji} \leq b_j, \quad j = 1, \ldots, m$$

gdzie $m = 1$ w przypadku klasycznego zadania załadunku lub $m \geq 2$ w przypadku wielowymiarowego zadania załadunku.

Dla zadania przydziału wygodniej jest rozważyć odmienną niż w (1.4) postać zbioru Q:

$$Q = \{ij : i = 1, \ldots, m, \; j = 1, \ldots, n\}, z(F) = \sum_{ij \in F} c_{ij},$$

$$F \in \mathcal{F} \text{ wtedy i tylko wtedy, kiedy}$$

$$|F \cap \{i1, \ldots, in\}| = 1 \text{ dla wszystkich } i = 1, \ldots, m$$

oraz

$$|F \cap \{1j, \ldots, mj\}| \leq 1 \text{ dla wszystkich } j = 1, \ldots, n,$$

gdzie $|A|$ oznacza liczbę elementów (moc) zbioru A.

Natomiast dla zadań skojarzenia

$$Q = \{ij : i = 1, \ldots, n, j = 1, \ldots, n, i \neq j\}$$

$F \in \mathcal{F}$ wtedy i tylko wtedy, kiedy

$|F \cap \{1i, \ldots, (i-1)i, i(i+1), \ldots, in\}| = 1$ dla wszystkich $i = 1, \ldots, n$

w przypadku zadania doskonałego skojarzenia lub

$$|F \cap \{1i, \ldots, (i-1)i, i(i+1), \ldots, in\}| \leq 1$$

w przypadku zadania skojarzenia.

Zadania pokrycia, pakowania i rozbicia zbiorów

Przyjęty w (1.4) sposób zdefiniowania zbioru $\mathcal{F}$ pozwala opisać ważne klasy zadań optymalizacji dyskretnej znane jako problemy: *pokrycia zbioru*, *pakowania zbioru* i *rozbicia zbioru* (*set-covering*, *set-packing* oraz *set-partitioning*). Niech $N = \{1, \ldots, n\}$ będzie zbiorem skończonym, $Q = \{1, \ldots, q\}$ i niech $\{N_j : N_j \subseteq N,\ j \in Q\}$ będzie zbiorem podzbiorów N. Dla przykładu zbiór ten może zawierać wszystkie podzbiory N o mocy k, gdzie $k \leq n$. Będziemy mówić, że

$$F \subseteq Q \text{ jest } \textit{pokryciem } N \text{ jeżeli } \cup_{j \in F} N_j = N.$$

W przypadku przedstawienia zadania pokrycia zbioru jako zadania optymalizacji dyskretnej w sformułowaniu (1.5), otrzymamy

$$\mathcal{F} = \{F : F \text{ jest pokryciem } N\}.$$

Będziemy mówić, że

$$F \subseteq Q \text{ jest } \textit{pakowaniem} \text{ zbioru } N \text{ jeżeli } N_j \cap N_k = \emptyset \text{ dla wszystkich } j, k \in F, j \neq k.$$

W przypadku przedstawienia zadania pakowania zbioru jako zadania optymalizacji dyskretnej w sformułowaniu (1.5), mamy

$$\mathcal{F} = \{F : F \text{ jest pakowaniem } N\}.$$

Natomiast, jeśli $F \subseteq Q$ jest równocześnie pokryciem i pakowaniem zbioru N, to wtedy bedziemy mówili, że F jest *rozbiciem* N. Zadanie optymalizacji dyskretnej w sformułowaniu (1.5) przyjmuje postać:

$$\mathcal{F} = \{F : F \text{ jest rozbiciem } N\}. \tag{1.9}$$

Aby sformułować odpowiednie zadania optymalizacyjne należy zdefiniować wartości współczynników funkcji celu c_j wykorzystane do zapisania funkcji celu w postaci $z(F) = \sum_{j \in F} c_j$ w definicji zadania binarnego (1.4) i (1.5). W przypadku zadania pokrycia, c_j jest kosztem przypisanym N_j i w zadaniu optymalizacyjnym poszukujemy pokrycia o minimalnym koszcie łącznym. W przypadku zadania pakowania, c_j jest wagą lub wartością N_j, zaś celem zadania optymalizacyjnego jest wyznaczenie pakowania zbioru N o maksymalnej wadze lub wartości.

Jeżeli każdemu elementowi i zbioru $N = \{1, \ldots, n\}$ przypisana jest pewna waga x_i , $x_i > 0$, $i = 1, \ldots, n$, to wtedy jednym z możliwych sposobów zdefiniowania c_j może być poniższy wzór:

$$c_j = \sum_{i \in N_j} x_i. \tag{1.10}$$

W przypadku zadania rozbicia odpowiadające mu zadania optymalizacyjne mają nieco odmienną postać niż (1.5). W wielu zastosowaniach praktycznych sprawą o zasadniczym znaczeniu jest, aby zbiór N był rozbity na dokładnie m podzbiorów (bloków), $m \geq 2$. W przyjętej w monografii notacji oznacza to przyjęcie dodatkowego ograniczenia na moc każdego z rozwiązań dopuszczalnych zadania F, w związku z czym definicja (1.9) przyjmie postać

$$\mathcal{F} = \{F : F \text{ jest rozbiciem } N \text{ oraz } |F| = m\}, \tag{1.11}$$

gdzie $|F|$ oznacza moc (liczbę elementów) zbioru F. Również postać funkcji celu zadań optymalizacyjnych związanych z zadaniem rozbicia zbioru jest odmienna niż w ogólnym sformułowaniu (1.5).

Zadanie *rozbicia* (*partioning problem*) ma poniższą postać:

$$z_{OPT}(n, m) = \min_{F \in \mathcal{F}} \left\{ \max_{j \in F} c_j - \min_{k \in F} c_k \right\}. \tag{1.12}$$

Celem tak sformułowanego zadania optymalizacyjnego jest dokonanie rozbicia zbioru $N = \{1, \ldots, n\}$ na m podzbiorów tak, aby wagi lub wartości każdego z nich c_j, patrz (1.10), możliwie najmniej różniły się między sobą. Zastosowanie praktyczne może dotyczyć sytuacji, kiedy mamy grupę m osób, na przykład spadkobierców oraz n obiektów każdy o wartości x_i, $i = 1, \ldots, n$, stanowiących mienie podlegające procedurze spadkowej. Celem zadania (1.12) jest dokonanie takiego przypisania obiektów do osób aby podział mienia pomiędzy spadkobiorców był możliwie najbardziej sprawiedliwy w sensie łącznej wartości obiektów przyznanych każdej osobie.

Inny przykład stanowi zadanie *rozbicia zbioru* (*set partioning problem*):

$$z_{OPT}(n, m) = \min_{F \in \mathcal{F}} \left\{ \max_{j \in F} c_j \right\}. \tag{1.13}$$

Jako potencjalne zastosowanie praktyczne rozważmy następujące zadanie. Mamy zbiór prac $N = \{1, \ldots, n\}$ oraz m identycznych, pracujących równolegle maszyn. Wykonanie każdej z prac na dowolnej z maszyn wymaga czasu x_i, $i = 1, \ldots, n$, gdzie c_j jest opisane przez (1.10). Celem zadania optymalizacyjnego jest dokonanie takiego przydziału prac do maszyn, aby zminimalizować całkowity czas procesu produkcyjnego (*makespan*, czas najdłużej pracującej maszyny).

Zadania (1.12) oraz (1.13) są do siebie bardzo zbliżone. W szczególnym przypadku, kiedy, $m = 2$ są one sobie tożsame w sensie postaci rozwiązań optymalnych.

Innym bardzo ważnym zadaniem o zbliżonym charakterze jest *zadanie pakowania zasobników* (*bin packing problem*). Mamy zbiór n obiektów $N = \{1, \ldots, n\}$, każdy o rozmiarze x_i, $0 \leq x_i \leq c$, $i = 1, \ldots, n$, oraz nieograniczony zbiór zasobników, każdy o pojemności c, gdzie c jest stałą. Zadanie optymalizacyjne polega na dokonaniu takiego zapakowania obiektów do zasobników, aby łączna liczba wykorzystanych zasobników była minimalna. Zbiór $\mathcal{F}$ - rozwiązań dopuszczalnych zadania pakowania zasobników można zapisać w poniższej modyfikacji zadania rozbicia zbioru (1.9), gdzie c_j jest opisane przez (1.10):

$$\mathcal{F} = \{F : F \text{ jest rozbiciem } N \text{ oraz } c_j \leq c \text{ dla wszystkich } j \in F\}.$$

Zadanie optymalizacyjne można sformułować w poniższej postaci, odmiennej od ogólnego sformułowania (1.5):

$$z_{OPT}(n) = \min_{F \in \mathcal{F}} |F| \,. \tag{1.14}$$

Zadanie pakowania zasobników posiada wiele ważnych zastosowań praktycznych. Materiały takie jak tkaniny, przewody elektryczne, rury i wiele innych są na ogół dostarczane w standardowych opakowaniach o pewnym rozmiarze. W procesach produkcyjnych są często potrzebne odcinki wyżej wymienionych materiałów o zadanej długości. Niech c oznacza rozmiar, na przykład długość materiału, a x_i potrzebne odcinki materiału. Rozwiązanie zadania pakowania zasobników (1.14) pozwala optymalnie wykorzystywać materiał, minimalizując straty. Jeżeli pojęcie *materiał* w powyższych rozważaniach zastąpimy pojęciem *zasoby*, to lista potencjalnych zastosowań zadania może się znacznie zwiększyć. Na przykład niech zasobem będzie waga. Niech c oznacza maksymalne obciążenie każdej z jednakowych ciężarówek, zaś x_i wagę każdego z n ładunków do przewiezienia. Rozwiązując zadanie (1.14) minimalizujemy liczbę ciężarówek niezbędnych do przewiezienia wszystkich ładunków. Niech zasobem będzie czas oraz niech c oznacza czas trwania przerwy na reklamy

w radiu lub telewizji, x_i czas trwania każdej z reklam. Rozwiązanie zadania (1.14) pozwala w tym przypadku optymalnie wykorzystać przerwy na reklamę przy planowaniu programu.

Zadania pakowania, pokrycia i rozbicia mogą być z łatwością przedstawione w postaci zadań programowania binarnego. Przyjmując dla wszystkich $i \in N$, N_j, $j \in Q$, $F \subseteq Q$:

$$a_{ij} = \begin{cases} 1 & \text{jeśli } i \in N_j \\ 0 & \text{jeśli } i \notin N_j \end{cases} \qquad y_j = \begin{cases} 1 & \text{jeśli } j \in F \\ 0 & \text{jeśli } j \notin F \end{cases},$$

F będzie dla zbioru N:

$$\text{pokryciem, jeżeli} \quad : \quad \sum_{j=1}^{q} a_{ij} \cdot y_j \geq 1 \text{ dla wszystkich } i = 1, \ldots, n;$$

$$\text{pakowaniem, jeżeli} \quad : \quad \sum_{j=1}^{q} a_{ij} \cdot y_j \leq 1 \text{ dla wszystkich } i = 1, \ldots, n;$$

$$\text{rozbiciem, jeżeli} \quad : \quad \sum_{j=1}^{q} a_{ij} \cdot y_j = 1 \text{ dla wszystkich } i = 1, \ldots, n.$$

Na przykład zadanie pakowania zbioru jest szczególnym przypadkiem zadania programowania binarnego w postaci:

$$z_{OPT}(n) = \max \left\{ \sum_{j=1}^{q} c_j \cdot x_j : \sum_{j=1}^{q} a_{ij} \cdot y_j \leq 1,\ y_j = 0 \text{ lub } 1,\ i = 1, ..., n \right\},$$

gdzie dodatkowo $a_{ij} = 0$ lub 1, $i = 1, \ldots, n$, $j = 1, \ldots, q$. Zauważmy także, że zadanie przydziału z n pracami oraz n osobami jest zadaniem rozbicia zbioru, w którym $N = \{1, \ldots, n, n+1, \ldots 2n\}$ oraz N_j, $|N_j| = 2$, dla $j = 1, \ldots, n^2$, są dwuelementowymi podzbiorami N, każdy N_j składa się z jednej pracy i jednej osoby.

Wiele praktycznych zagadnień może zostać przedstawionych w postaci zadań pokrycia zbioru. Jako typowy przykład może służyć *zadanie lokalizacyjne*. Załóżmy, że dany jest zbiór $N = \{1, \ldots, n\}$ potencjalnych lokalizacji stacji straży pożarnej. Koszt zlokalizowania stacji w miejscu j wynosi c_j. Dany jest również zbiór $M = \{1, \ldots, m\}$ zamieszkałych obszarów (ulic, osiedli lub miejscowości), które mają podlegać ochronie przeciwpożarowej. Podzbiór obszarów, które mogą być chronione z lokalizacji j oznaczamy jako M_j. Dla ustalenia uwagi M_j może zostać zdefiniowane jako zbiór obszarów, do których można dojechać z lokalizacji j w ciągu 10 minut. Wtedy problem zlokalizowania stacji straży pożarnej w najbardziej ekonomiczny sposób, ale

przy spełnieniu warunku, że straż pożarna może dojechać do miejsca pożaru w co najwyżej 10 minut, jest zadaniem pokrycia zbioru. Istnieje wiele innych zastosowań o podobnym charakterze, jak na przykład przydział klientów do tras samochodów rozwożących dostawy do ich miejsca zamieszkania, załóg samolotów do lotów lub pracowników do zmian w zakładach pracy itp.

Powyższe modele pokazują, jak można wykorzystać liniowe ograniczenia z wykorzystaniem zmiennych binarnych w celu przedstawienia pewnych współzależności pomiędzy zdarzeniami. Na przykład ograniczenie

$$\sum_{j=1}^{q} a_{ij} \cdot y_j \leq 1, \text{ gdzie } y_j = 0 \text{ lub } 1$$

w zadaniu pakowania zbioru, pozwala modelować taką sytuację, że co najwyżej jedno ze zdarzeń y_j może zaistnieć. Analogicznie, w przypadku zadań pokrycia lub rozbicia zbioru ograniczenia stanowią, że musi zaistnieć odpowiednio co najmniej jedno ze zdarzeń lub dokładnie jedno zdarzenie. W rzeczywistości znacznie bardziej złożone współzależności mogą być modelowane z zastosowaniem zmiennych binarnych. Poniżej zostaną przedstawione wybrane możliwości modelowania sytuacji rzeczywistych z wykorzystaniem współzależności zmiennych binarnych.

Rozważmy sytuację, gdy dane są dwie zmienne binarne: x_1, x_2 opisujące wystąpienie ($x_j = 1$, $j = 1, 2$) lub niewystąpienie ($x_j = 0$, $j = 1, 2$) pewnych zdarzeń. Równość $x_2 - x_1 = 0$ oznacza, że oba zdarzenia muszą zaistnieć równocześnie albo żadne z nich nie może zaistnieć. Podobnie, nierówność $x_2 - x_1 \leq 0$ oznacza, że zdarzenie 2 może zaistnieć tylko wtedy, kiedy zaistnieje zdarzenie 1. Ogólnie, rozważmy zmienną ciągłą y, która może przyjąć dowolną wartość z zakresu $0 \leq y \leq u$. Załóżmy, że przyjęcie przez y wartości większej od zera jest uwarunkowane przez zmienną binarną x. Taka relacja może być opisana przez liniową nierówność $y - ux \leq 0$, która w przypadku $x = 0$ wymusza $y = 0$, zaś dla $x = 1$ dopuszcza, aby y przyjmowało wartości z pełnego zakresu zmienności $0 \leq y \leq u$. Poniżej zostaną zaprezentowane dwa modele wykorzystujące zależności tego rodzaju.

Zadania lokalizacyjne

Powyżej zadanie lokalizacyjne zostało przedstawione w uproszczonej postaci jako realizacja modelu zadania pokrycia zbioru. W ogólnej postaci *zadanie lokalizacyjne* (*facility location problem*) przyjmuje postać opisaną poniżej.

Dany jest zbiór potencjalnych lokalizacji usług $N = \{1, \ldots, n\}$ oraz zbiór klientów $I = \{1, \ldots, m\}$. Umieszczenie punktu świadczenia usług w lokalizacji j, dla $j \in N$, kosztuje c_j. Ogólna postać zadania lokalizacyjnego tym

różni się od jego uproszczonej wersji, że każdy klient i, $i \in I$, posiada określone zapotrzebowanie na dobra lub usługi oferowane w lokalizacji j. Łączny koszt zaspokojenia potrzeb klienta i w lokalizacji j wynosi h_{ij}. Zadanie optymalizacyjne polega na wyznaczeniu takiego podzbioru możliwych lokalizacji oraz przypisaniu do nich klientów, aby łączny koszt świadczenia usług był minimalny. W przypadku nieograniczonego zadania lokalizacyjnego nie istnieje ograniczenie na liczbę klientów, którzy mogą być obsłużeni w każdej z lokalizacji.

Wprowadzamy zmienną binarną x_j, $j \in N$, przyjmującą wartość $x_j = 1$, jeżeli usługa jest umieszczona w lokalizacji j oraz $x_j = 0$ w przeciwnym przypadku. Dodatkowo wprowadzamy zmienną ciągłą y_{ij}, która opisuje stopień zaspokojenia potrzeb klienta i w lokalizacji j. Warunek, że potrzeby każdego z klientów muszą być w pełni zaspokojone, określa ograniczenie:

$$\sum_{j \in N} y_{ij} = 1 \text{ dla każdego } i \in I. \tag{1.15}$$

Ponadto, ponieważ klient i może zostać obsłużony z lokalizacji j wtedy i tylko wtedy, kiedy jest tam umieszczona usługa, mamy ograniczenia o postaci:

$$y_{ij} - x_j \leq 0 \text{ dla wszystkich } i \in I \text{ oraz } j \in N. \tag{1.16}$$

Dlatego *nieograniczone zadanie lokalizacyjne* (*uncapacitated facility location problem*) jest zadaniem liniowego mieszanego programowania całkowitoliczbowego (1.1) z funkcją celu:

$$\min \left\{ \sum_{j \in N} c_j x_j + \sum_{i \in I} \sum_{j \in N} h_{ij} y_{ij} \right\}$$

przy spełnieniu ograniczeń (1.15), (1.16), gdzie $x_j = 0$ lub 1, $y_{ij} \geq 0$, $j \in N$, $i \in I$.

Założenie, że lokalizacja j może obsłużyć nieograniczoną liczbę klientów może okazać się niezbyt realistyczne. Wobec tego przyjmijmy, że usługa umieszczona w lokalizacji j ma ograniczenie u_j oraz, że klient i ma zapotrzebowanie b_i. Teraz y_{ij} będzie oznaczać ilość dóbr lub usług świadczonych z lokalizacji j do klienta i. Niech h_{ij} oznacza koszt dostarczenia jednej sztuki dobra lub usługi z lokalizacji j do klienta i. Aby sformułować *ograniczone zadanie lokalizacyjne* (*capacitated facility location problem*) w postaci zadania liniowego mieszanego programowania całkowitoliczbowego (1.1) ograniczenie (1.15) należy zastąpić przez:

$$\sum_{j \in N} y_{ij} = b_i \text{ dla kazdego } i \in I.$$

Natomiast ograniczenie (1.16) należy zastąpić przez:

$$\sum_{i \in I} y_{ij} - u_j x_j \leq 0 \text{ dla każdego } j \in N.$$

1.4 Zadania teorii grafów

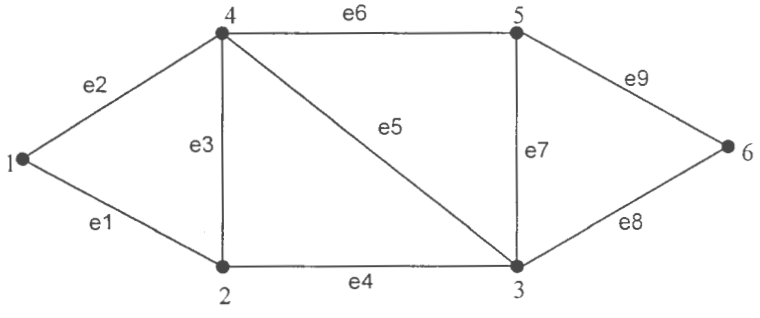
Zadania optymalizacyjne na grafach stanowią bardzo istotną grupę zadań optymalizacji dyskretnej i mają zastosowanie w modelowaniu wielu ważnych zagadnień praktycznych. Teoria grafów jest bardzo obszerną i bogatą dziedziną badawczą, której przegląd znajduje się między innymi w książce J.L. Kulikowskiego [91]. Celem tego podrozdziału jest zarysowanie jej specyfiki w odniesieniu do innych rozważanych zadań optymalizacji dyskretnej.

Wyróżniamy dwa zasadnicze rodzaje grafów: grafy nieskierowane oraz grafy skierowane.

Definicja 1 *Graf nieskierowany G jest zadany przez zbiór wierzchołków $V = \{1, \ldots, n\}$ oraz zbiór krawędzi $E, E \subseteq V \times V$. Istnienie krawędzi $e \in E$, $e = (i, j)$, $i, j \in V$ oznacza, że łączy ona wierzchołki: i oraz j.*

Krawędź w grafie nieskierowanym jest przypisana nieuporządkowanej parze wierzchołków, co oznacza, że krawędź (i, j) jest tożsama z krawędzią (j, i).

Graf nieskierowany jest wykorzystywany do modelowania struktur połączeń logicznych, na przykład dróg pomiędzy miejscowościami, schematów logicznych i wielu innych. Na rysunku 1.1 przedstawiony jest graf nieskierowany z sześcioma wierzchołkami oraz dziewięcioma krawędziami.

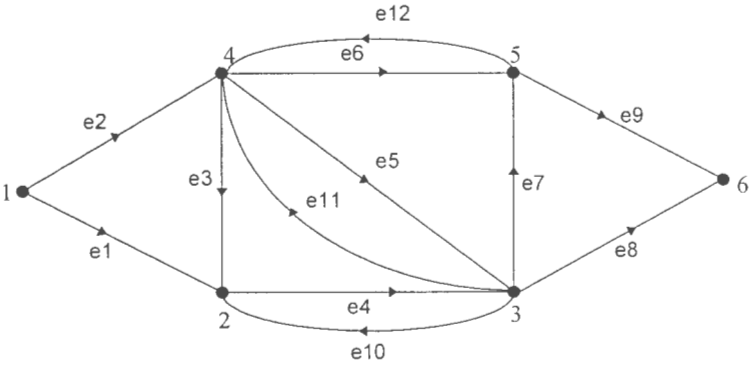


Rysunek 1.1: Graf nieskierowany

Definicja 2 *Graf skierowany G jest zadany przez zbiór wierzchołków $V = \{1, \ldots, n\}$ oraz zbiór łuków A, $A \subseteq V \times V$. Łuk $e = (i, j)$ jest skierowany z wierzchołka i do wierzchołka j (łączy i z j).*

W definicji łuku w grafie skierowanym G porządek tworzących go wierzchołków jest istotny. W szczególności może się okazać, że w grafie G istnieje łuk $e = (i, j)$, a nie istnieje w nim łuk $e' = (j, i)$.

Grafy skierowane są wykorzystywane do modelownia zjawisk, w których istotny jest kierunek połączenia. Są to na przykład drogi lub ulice jednokierunkowe, przepływ ropy naftowej lub gazu w rurociągach, przeplyw ładunków elektrycznych w przewodach i wiele innych. Na rysunku 1.2 przedstawiono graf skierowany z sześcioma wierzchołkami oraz dwunastoma łukami.



Rysunek 1.2: Graf skierowany

Poniżej przedstawiono definicje wybranych, ważnych rodzajów grafów.

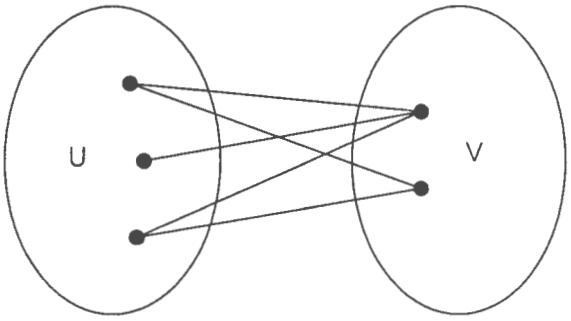
- *Grafem ważonym* nazywamy graf, w którym każdemu łukowi (krawędzi) $e \in A$ jest przyporządkowana *waga* $c(e)$. Jest to wartość liczbowa reprezentująca na przykład długość drogi, czas przejazdu lub jego koszt, przepustowość lub inną cechę ilościową wynikającą z istoty modelowanego zjawiska.

- Jeśli dla dowolnych dwóch wierzchołków danego grafu i oraz j istnieje krawędź lub łuk (i, j) to taki graf nazywamy *pełnym*.

- Natomiast graf, w którym zbiór łuków lub krawędzi jest niewielki w porównu do zbioru $V \times V$ nazywamy *grafem rzadkim*.

- Graf nazywamy *dwudzielnym* (*bipartite graph*), jeśli jego zbiór wierzchołków może zostać podzielony na dwa rozłączne podzbiory U oraz V ($U \cap V = \emptyset$) przy czym nie istnieje krawędź (łuk) $e = (i, j) \in E$ taka, że $i \in U$ oraz $j \in U$ lub $j \in V$ oraz $i \in V$. Każda z krawędzi grafu dwudzielnego łączy wierzchołek z U i wierzchołek z V.

Na rysunku 1.3 przedstawiono poglądowo graf dwudzielny nieskierowany



Rysunek 1.3: Graf dwudzielny

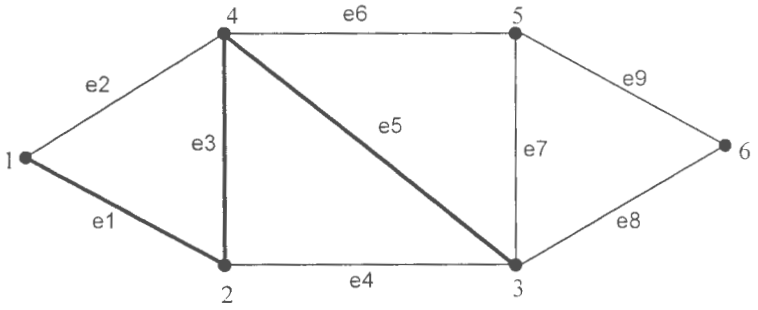
Na rysunku 1.3 widać, że przy graficznej reprezentacji grafu dwudzielnego nieskierowanego na płaszczyźnie niektóre z jego krawędzi mogą się przecinać. Jednym z ważnych zastosowań grafów dwudzielnych (lub szerzej wielodzielnych) jest projektowanie schematów połączeń logicznych, na przykład w elektronice przy projektowaniu płytek drukowanych. Wystąpienie dużej ilości przecięć krawędzi może powodować negatywne konsekwencje praktyczne, takie jak duża liczba warstw napylanych na płytkach. Dlatego bardzo ważne jest wyznaczenie reprezentacji modelowanego obiektu jako grafu dwudzielnego na płaszczyźnie o jak najmniejszej liczbie przecięć krawędzi. Bardziej szczegółowo zagadnienie wyznaczania liczby przecięć krawędzi w grafie dwudzielnym nieskierowanym na płaszczyźnie zostało omówione w pracy May i Szkatuła [106].

Drogi i drzewa

- *Drogą* z wierzchołka i_1 do wierzchołka i_k w grafie G nazywamy ciąg krawędzi (łuków) $(i_1, i_2), (i_2, i_3), \ldots, (i_{k-2}, i_{k-1}), (i_{k-1}, i_k)$. W przypadku grafu skierowanego droga jest *skierowana* z i_1 do i_k.

- *Cyklem* nazywamy drogę zamkniętą, czyli taką, że $i_1 = i_k$. Graf nie zawierający cykli nazywamy *acyklicznym*.
- *Podgrafem* grafu G nazywamy graf, którego wierzchołki i łuki należą do grafu G.

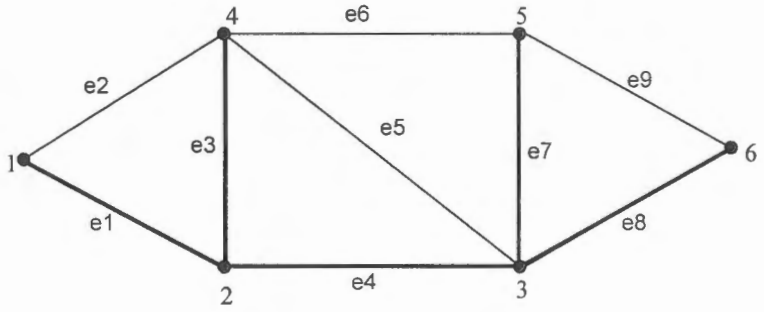
Na rysunku 1.4 w grafie nieskierowanym o sześciu wierzchołkach i dziewięciu krawędziach linią pogrubioną zaznaczono podgraf o czterech wierzchołkach i trzech krawędziach.



Rysunek 1.4: Podgraf w grafie nieskierowanym

- Nieskierowany graf G nazywamy *spójnym*, jeśli istnieje w nim droga między każdą parą wierzchołków i oraz j.
- Graf skierowany G nazywamy (*słabo*) *spójnym* wtedy, kiedy graf nieskierowany, uzyskany z G poprzez usunięcie orientacji łuków jest spójny.
- Graf skierowany G nazywamy *silnie spójnym* wtedy, kiedy dowolną uporządkowaną parę jego wierzchołków (i, j) można połączyć drogą skierowaną z i do j.
- Spójny nieskierowany graf acykliczny nazywamy *drzewem* (*niskierowanym*).
- *Drzewem rozpinającym* w spójnym nieskierowanym grafie G, nazywamy podgraf, który jest drzewem i zawiera wszystkie wierzchołki G.
- Zbiór drzew nazywamy *lasem*.

Na rysunku 1.5 w grafie nieskierowanym o sześciu wierzchołkach i dziewięciu krawędziach linią pogrubioną zaznaczono drzewo rozpinające.



Rysunek 1.5: Drzewo rozpinające w grafie nieskierowanym

Istotna część zadań optymalizacyjnych na grafach związana jest z grafami ważonymi, to znaczy kiedy łukom (lub krawędziom) grafu $e \in A$ przypisane są wartości liczbowe - wagi $c(e)$, często rozumiane jako odległości pomiędzy wierzchołkami. Dobrym przykładem zadania optymalizacyjnego na grafach jest problem wyznaczenia *najkrótszego drzewa rozpinającego* (*minimum spanning tree*). Długością każdego drzewa rozpinającego T w spójnym nieskierowanym grafie ważonym G jest suma wag jego krawędzi:

$$z(T) = \sum_{e_i \in T} c(e_i).$$

Niech $\mathcal{T}$ oznacza zbiór wszystkich drzew rozpinających w grafie G. Zadanie wyznaczenia najkrótszego drzewa rozpinającego w grafie ważonym G polega na wyznaczeniu drzewa rozpinającego T^* o najmniejszej długości

$$z(T^*) = \min_{T \in \mathcal{T}} z(T).$$

Długością drogi od wierzchołka i_1 do wierzchołka i_k jest suma wag tworzących ją krawędzi (i_1, i_2), $(i_2, i_3), \ldots, (i_{k-2}, i_{k-1})$, (i_{k-1}, i_k). Zagadnienia związane z wyznaczaniem najkrótszych dróg w grafach ważonych są podstawowymi i najczęściej spotykanymi zagadnieniami w analizie sieci transportowych i komunikacyjnych. Jest to bardzo obszerna dziedzina teorii grafów i optymalizacji dyskretnej. Na przykład, możemy poszukiwać najkrótszej (najbardziej ekonomicznej lub najszybszej) drogi z jednego zadanego wierzchołka i do drugiego zadanego wierzchołka j lub z wierzchołka i do wszystkich pozostałych wierzchołków grafu, lub między wszystkimi parami wierzchołków. W pewnych sytuacjach istotne jest, aby droga z wierzchołka i do wierzchołka j przechodziła przez zadane wierzchołki $l_1, l_2, \ldots, l_k$. W pewnych przypadkach należy wyznaczyć nie tylko najkrótszą, ale także drugą, trzecią i ewentualnie kolejne, najkrótsze drogi. Dla niektórych zastosowań istotna jest tylko wartość najkrótszej drogi.

Spośród wielu możliwych wariantów zadań wyznaczenia najkrótszych dróg najbardziej uniwersalne wydają się dwa:

- Wyznaczenie najkrótszej drogi oraz jej długości z jednego zadanego wierzchołka i do drugiego zadanego wierzchołka j.
- Wyznaczenie najkrótszych dróg oraz ich długości dla każdej pary wierzchołków w grafie ważonym.

W oparciu o dwa powyższe zadania można sformułować i rozwiązać wiele zadań pokrewnych.

Wiele zadań teorii grafów może być przedstawionych w postaci zadań binarnych. Poniżej przedstawiamy dwa z takich zadań: zadanie przepływu w sieci oraz zadanie komiwojażera.

Zadanie przepływu w sieci

Rozważmy sieć (graf skierowany), która jest reprezentowana przez zbiór wierzchołków V (miejsc związanych ze świadczeniem, korzystaniem i przekazywaniem dóbr lub usług) oraz przez zbiór łuków A. Istnienie łuku $e = (i, j), e \in A$ oznacza, że istnieje bezpośrednie połączenie transportowe skierowane z wierzchołka i do wierzchołka j oraz związany z tym łukiem przepływ dóbr lub usług. Do każdego wierzchołka i przypisana jest waga b_i zwana zapotrzebowaniem. W zależności od wartości b_i wierzchołek i jest konsumentem ($b_i > 0$), dostawcą ($b_i < 0$) lub punktem przekazu ($b_i = 0$) dóbr lub usług. Przyjmujemy założenie, że łączne zapotrzebowanie wynosi 0, to znaczy, że $\sum_{i \in V} b_i = 0$. Ponadto, dla każdego łuku (i, j) zostały zdefiniowane: przepustowość przepływu u_{ij} oraz koszt przepływu h_{ij} jednostki dóbr lub usług.

Niech zmienna ciągła y_{ij} oznacza wielkość przepływu przez łuk (i, j). Przepływ w sieci będziemy nazywać dopuszczalnym wtedy i tylko wtedy, kiedy spełnione są poniższe ograniczenia

$$0 \leq y_{ij} \leq u_{ij} \text{ dla wszystkich } (i, j) \in A \tag{1.17}$$

$$\sum_{j \in V} y_{ij} - \sum_{j \in V} y_{ji} = b_i \text{ dla każdego } i \in V. \tag{1.18}$$

Ograniczenia (1.18) są nazywane ograniczeniami *zachowania przepływu*. Zadanie optymalizacyjne:

$$\min \left\{ \sum_{(ij) \in A} h_{ij} y_{ij} : \text{ gdzie } \quad y_{ij}, (i, j) \in A \quad \text{spełniają} \quad (1.17) \text{ i } (1.18) \right\}$$

jest nazywane *zadaniem przepływu w sieci* (*network flow problem*).

Zadanie przepływu o ustalonym koszcie (*fixed-charge network flow problem*) może zostać uzyskane, jeżeli istnieje dodatni przepływ na łuku (i,j), poprzez wprowadzenie dodatkowego, ustalonego kosztu c_{ij}. Wprowadzamy zmienną binarną x_{ij}, aby wskazać, że łuk (i,j) jest $(x_{ij} = 1)$ lub nie jest $(x_{ij} = 0)$ wykorzystywany. Spełnienie warunku $y_{ij} = 0$ jeśli $x_{ij} = 0$, dla każdego łuku $(i,j) \in A$, jest zagwarantowane przez:

$$y_{ij} - u_{ij}x_{ij} \leq 0 \quad \text{dla wszystkich} \quad (i,j) \in A. \tag{1.19}$$

Stąd uzyskujemy sformułowanie zadania przepływu o ustalonym koszcie w postaci zadania liniowego mieszanego programowania całkowitoliczbowego:

$$\min \left\{ \sum_{(i,j)\in A} (c_{ij}x_{ij} + h_{ij}y_{ij}) : \text{spełnione są (1.18) i (1.19)} \right\}$$

$$\text{gdzie } x_{ij} = 0 \text{ lub } 1,\ y_{ij} \geq 0 \text{ dla wszystkich } (i,j) \in A.$$

Zadanie przepływu o ustalonym koszcie jest modelem bardzo użytecznym w przypadku opisu zjawisk związanych z przepływem surowców lub materiałów w sieciach. Można tu wymienić sieci zaopatrujące w wodę, gaz i paliwa płynne, jak również systemy ogrzewania oraz sieci dróg.

Zadanie komiwojażera

Rozważany jest graf ważony G, zadany przez $V = \{1, \ldots, n\}$ - zbiór wierzchołków oraz A - zbiór krawędzi lub łuków. Dla każdej krawędzi lub łuku $e = (i,j)$ mamy wagę c_{ij}. W przypadku grafu nieskierowanego mamy $c_{ij} = c_{ji}$, natomiast w grafie skierowanym łuk (j,i) może nie istnieć lub $c_{ij} \neq c_{ji}$. Interpretacja tego modelu może być taka, że V jest zbiorem miast, A jest zbiorem uporządkowanych par miast (i,j), dla których istnieje możliwość bezpośredniego przejazdu z miasta i do miasta j, natomiast c_{ij} jest odległością lub czasem przejazdu.

Zadanie optymalizacyjne, znane jako *zadanie komiwojażera* (*traveling salesman problem*) polega na wyznaczeniu drogi rozpoczynającej się w zadanym mieście, dla ustalenia uwagi w wierzchołku 1, która:

- zawiera każdy wierzchołek (odwiedza każde miasto) dokładnie raz oraz kończy się w wierzchołku 1.
- łączna długość drogi lub czasu przejazdu jest minimalna.

Zauważmy, że tak postawione zadanie jest tożsame z wyznaczeniem w grafie G cyklu z początkiem i końcem w wierzchołku 1, w którym każdy z wierzchołków sieci odwiedzony jest dokładnie raz, zaś łączna jego długość jest minimalna. Cykl taki nazywany jest *cyklem Hamiltona.*

Aby sformułować zadanie komiwojażera w postaci zadania programowania binarnego należy wprowadzić zmienną binarną:

$$x_{ij} = 0 \text{ lub } 1 \text{ dla wszystkich } (i,j) \in A, \tag{1.20}$$

$x_{ij} = 1$, jeśli miasto j jest odwiedzane bezpośrednio po mieście i oraz $x_{ij} = 0$ w przeciwnym przypadku.

Wymóg, że miasto jest odwiedzone dokładnie raz, to znaczy, że istnieje dokładnie po jednej krawędzi lub łuku wchodzącym i wychodzącym z każdego z wierzchołków w poszukiwanej drodze, jest zagwarantowany przez dwa poniższe ograniczenia:

$$\sum_{\{i:(i,j)\in A\}} x_{ij} = 1 \text{ dla każdego } j \in V \tag{1.21}$$

oraz

$$\sum_{\{j:(i,j)\in A\}} x_{ij} = 1 \text{ dla każdego } i \in V \tag{1.22}$$

Ograniczenia (1.20)-(1.22) nie są wystarczające dla zdefiniowania drogi odwiedzającej wszystkie miasta, gdyż mogą być one spełnione przez podzbiór wierzchołków grafu U, $U \subset V$, który nie tworzy cyklu a zawiera wszystkie wierzchołki grafu. Aby wyeliminować taką sytuację należy wymusić istnienie łuku łączącego U oraz $V \backslash U$. Można to uzyskać wprowadzając jedno z dwóch możliwych postaci ograniczeń dla każdego podzbioru wierzchołków grafu, $U, U \subset V$, $2 \leq |U| \leq |V| - 2$:

$$\sum_{\{(i,j)\in A:\, i\in U, j\in V\backslash U\}} x_{ij} \geq 1 \text{ dla każdego } U : \; 2 \leq |U| \leq |V| - 2 \tag{1.23}$$

lub

$$\sum_{\{(i,j)\in A:\, i\in U, j\in V\backslash U\}} x_{ij} \leq |U| - 1 \text{ dla każdego } U : \; 2 \leq |U| \leq |V| - 2 \tag{1.24}$$

Zadanie komiwojażera może więc być przedstawione w dwóch sformułowaniach w postaci zadań programowania binarnego:

$$z_{OPT}(n) = \min \left\{ \sum_{(i,j)\in A} c_{ij} x_{ij} : x \text{ spełnia (1.20)-(1.23)} \right\}$$

lub

$$z_{OPT}(n) = \min \left\{ \sum_{(i,j)\in A} c_{ij}x_{ij} : x \text{ spełnia (1.20)-(1.22), (1.24)} \right\}$$

Zauważmy, że niezależnie od tego, czy użyjemy ograniczeń postaci (1.23) czy (1.24), to liczba tych ograniczeń może być bardzo duża (nawet porównywalna z $2^{|V|}$), co może w praktyce obliczeniowej bardzo utrudnić efektywne rozwiązywanie zadania komiwojażera zapisanego w postaci zadania programowania binarnego.

1.5 Zadania harmonogramowania

Kolejną ważną grupę zadań optymalizacji dyskretnej stanowią *zadania harmonogramowania.* Podstawowym celem tej klasy zadań optymalizacji dyskretnej jest modelowanie i optymalizacja złożonych procesów produkcyjnych, sieci komputerowych i temu podobnych. Przyjmuje się założenie, że procesy będące przedmiotem modelowania są opisywane poprzez zadania (prace) do wykonania i relacje pomiędzy nimi oraz maszyny (procesory komputerów) na których prace te są wykonywane. W zależności od potrzeb stosowane są również dodatkowe zasoby (wraz z ich specyficznymi funkcjami operacyjnymi) oraz parametry opisujące wszystkie wymienione powyżej składniki z większą dokładnością. Podstawowym celem zadania optymalizacyjnego jest wyznaczenie optymalnego (lub sub-optymalnego) harmonogramu wykonania zadań z uwzględnieniem zadanego kryterium optymalizacyjnego. Celem tego podrozdziału jest zwięzłe zaprezentowanie podstawowych pojęć teorii harmonogramowania. Bardzo szczegółowy i dogłębny opis tej niezmiernie ważnej dziedziny jest zawarty w monografii Błażewicz i inni [13].

Podstawowe definicje

Zadania harmonogramowania, zwane też zadaniami *szeregowania prac* (*scheduling problems*) są definiowane poprzez trzy zbiory: $\mathcal{T} = \{T_1, T_2, \ldots T_n\}$ - *zbiór zadań*, $\mathcal{M} = \{M_1, M_2, \ldots M_m\}$ - *zbiór maszyn* (procesorów) oraz $\mathcal{R} = \{R_1, R_2, \ldots, R_s\}$ - *zbiór dodatkowych zasobów. Harmonogramowanie* (*szeregowanie*) *prac* polega na przypisaniu maszyn ze zbioru $\mathcal{M}$ oraz (ewentualnie) zasobów ze zbioru $\mathcal{R}$ do zadań ze zbioru $\mathcal{T}$ tak, aby przy spełnieniu zadanych ograniczeń wykonać zadania oraz osiągnąć najlepszą wartość kryterium optymalizacyjnego. W klasycznej teorii harmonogramowania przyjęto dwa założenia.

1. Każde zadanie może być wykonywane w tej samej jednostce czasu przez co najwyżej jedną maszynę.

2. Każda z maszyn może wykonywać na raz co najwyżej jedno zadanie.

Maszyny mogą być charakteryzowane jako:

- *Uniwersalne* (*parallel*), to znaczy wykonujące wszystkie funkcje.
- *Wyspecjalizowane* (*dedicated*) do wykonywania pewnych prac.

Wśród maszyn uniwersalnych można wyodrębnic trzy typy w zależności od ich prędkości. Jeśli wszystkie maszyny ze zbioru $\mathcal{M}$ mają jednakową prędkość wykonywania prac, to są nazywane *identycznymi* (*identical*), jeśli mają one różne prędkości, ale są one stałe dla każdej z maszyn i nie zależą od rodzaju zadań ze zbioru $\mathcal{T}$, to nazywamy je *jednolitymi* (*uniform*). Natomiast, jeśli czas pracy maszyny zależy od wykonywanej pracy ze zbioru $\mathcal{T}$ to nazywamy ją *niezwiązaną* (*unrelated*). Znane są trzy modele wykonywania grup zadań dla maszyn wyspecjalizowanych: *flow shop*, *open shop* oraz *job shop*.

Zadania $T_j \in \mathcal{T}$ są charakteryzowane poprzez następujące parametry:

1. *Wektor czasów wykonania* (*processing times*) $\mathbf{p}_j = [p_{1j}, p_{2j}, \ldots, p_{mj}]$, gdzie p_{ij} jest czasem niezbędnym do wykonania zadania T_j na maszynie M_i, $j = 1, \ldots, n$, $i = 1, \ldots, m$.

2. *Czas przybycia* lub inaczej *czas gotowości* (*arrival time*, *ready time*) r_j, to znaczy czas, w którym zadanie T_j, $j = 1, \ldots, n$, jest gotowe do wykonania.

3. *Termin zakończenia* (*deadline*) d_j, przed upływem którego zadanie T_j, $j = 1, \ldots, n$, powinno być zakończone.

4. *Wagę* (*priorytet*) w_j, współczynnik, który wyraża względną ważność zadania T_j, $j = 1, \ldots, n$.

Wykonanie pewnych prac może być wstrzymywane i podejmowane później bez dodatkowych kosztów. Ponadto zadania $\mathcal{T}$ mogą być podporządkowane zasadom *precedencji* (*precedence constraints*); $T_i \prec T_j$ oznacza, że wykonywanie zadania T_j nie może być rozpoczęte przed zakończeniem wykonywania zadania T_i, $i, j = 1, \ldots, n$.

Powyżej przytoczono tylko najważniejsze ze stosowanych charakterystyk maszyn i zadań. W przypadku modelowania bardziej złożonych procesów wytwarzania mogą one być w miarę potrzeb rozbudowywane.

Zadania optymalizacyjne

Dla każdego zadania T_j, $j = 1, 2, \ldots n$, oraz zadanego harmonogramu wykonania prac $\mathcal{T}$ na maszynach $\mathcal{M}$ można zdefiniowac poniższe wielkości:

- C_j, *czas zakończenia* (*completion time*) zadania T_j;
- $F_j = C_j - r_j$, *czas przepływu* (*flow time*);
- $L_j = C_j - d_j$, *opóźnienie (lateness)*;
- $D_j = \max\{C_j - d_j, 0\}$, miarę opóźnienienia wykonania zadania T_j (*po terminie, tardiness*);
- $E_j = \max\{d_j - C_j, 0\}$, miarę przyśpieszenia wykonania zadania T_j (*przed terminem, earliness*);
- $U_j = 1$: jeśli $C_j > d_j$; $U_j = 0$: jeśli , $C_j \leq d_j$, wskaźnik, że praca zostanie wykonana po terminie (*tardy task indicator*).

Aby ocenić harmonogramy i wybrać najlepszy z nich, należy zdefiniować *kryteria optymalizacyjne (optimality criteria)*, zwane też *miarą wydajności* (*performance measure*) konkretnego harmonogramu. Poniżej zostanie zaprezentowanych szereg z możliwych do zastosowania kryteriów optymalizacyjnych.

- $C_{\max} = \max\limits_{1 \leq j \leq n} C_j$, *długość harmonogramu (makespan)*;
- $\overline{F} = \frac{1}{n} \sum_{j=1}^{n} F_j$, *średni czas przepływu (mean flow time)*;
- $\overline{F}_w = \frac{1}{n} \sum_{j=1}^{n} w_j F_j / \sum_{j=1}^{n} w_j$, *średni ważony czas przepływu (mean weighted flow time)*;
- $L_{\max} = \max\limits_{1 \leq j \leq n} L_j$, *maksymalne opóźnienie (maximum lateness)*;
- $\overline{D} = \frac{1}{n} \sum_{j=1}^{n} D_j$, *średni czas wykonania zadania po terminie (mean tardiness)*;
- $\overline{D}_w = \frac{1}{n} \sum_{j=1}^{n} w_j D_j / \sum_{j=1}^{n} w_j$, *średni ważony czas wykonania zadania po terminie (mean weighted tardiness)*;
- $\overline{E} = \frac{1}{n} \sum_{j=1}^{n} E_j$, *średni czas wykonania zadania przed terminem (mean earliness)*;

- $\overline{E}_w = \frac{1}{n}\sum_{j=1}^{n} w_j E_j / \sum_{j=1}^{n} w_j$, *średni ważony czas wykonania zadania przed terminem (mean weighted earliness)*;
- $U = \sum_{j=1}^{n} U_j$, *liczba zadań wykonanych po terminie (number of tardy tasks)*;
- $U_w = \sum_{j=1}^{n} w_j U_j$, *ważona liczba zadań wykonanych po terminie (weighted number of tardy tasks).*

Zauważmy, że kryterium optymalizacyjne $C_{\max} = \max\limits_{1\leq j\leq n} C_j$ długości harmonogramu zostało rozważone w podrozdziale 1.3 jako szczególny przypadek zadania rozbicia zbioru, patrz (1.13).

Olbrzymia różnorodność zadań harmonogramowania spowodowała konieczność ich usystematyzowania i klasyfikacji. W pracach Błażewicz i inni [14] oraz Graham i inni [61] zaproponowano taką klasyfikację. Składa się ona z trzech pól i przyjmuje następującą postać:

$$\alpha|\beta|\gamma$$

Pierwsze pole $\alpha = \alpha_1, \alpha_2$ opisuje liczbę i rodzaj zastosowanych maszyn. Drugie pole $\beta = \beta_1, \beta_2, \ldots \beta_8$ opisuje rodzaje zadań, ich współzależności oraz zasoby. W przypadku przyjęcia przez któryś z parametrów $\alpha_1, \alpha_2, \beta_1, \beta_2, \ldots \beta_8$ wartości $\emptyset$ jest on pomijany w opisie zadania. Trzecie pole γ oznacza rodzaj przyjętego kryterium optymalizacyjnego. W rozdziale 6 rozważono szczegółowo *zadanie szeregowania prac z terminami zakończenia* sklasyfikowane jako $1||U_w$ zgodnie z powyższą klasyfikacją. Na jego przykładzie dokonano analizy zagadnień będących obiektem zainteresowania monografii.

1.6 Podsumowanie

Celem tego rozdziału było krótkie przedstawienie dziedziny badawczej znanej jako optymalizacja dyskretna. Optymalizacja dyskretna powstała na styku zastosowań praktycznych, skąd pochodzą problemy do rozwiązania, oraz działów matematyki, które dostarczyły niezbędnego aparatu, aby sformalizować problemy w postaci matematycznych zadań optymalizacyjnych ze skończonymi bądź przeliczalnymi zbiorami rozwiązań dopuszczalnych.

W tym rozdziale monografii dokonano prezentacji wybranych grup zadań optymalizacji dyskretnej. Dokonując wyboru zadań do prezentacji autor kierował się poniższymi przesłankami.

- Do prezentacji należy wybrać szczególnie ważne, klasyczne zadania optymalizacji dyskretnej.

- Wybrane zadania powinny mieć zróżnicowany charakter tak, aby zostały zaprezentowane zarówno różnorodne techniki formułowania zadań, jak również obszary potencjalnych zastosowań.
- Wiele ważnych zadań optymalizacji dyskretnej może zostać sformułowanych w postaci zadania liniowego mieszanego programowania całkowitoliczbowego lub w postaci zadań binarnych.
- Prezentacja zadań powinna mieć charakter poglądowy i nie powinna być zbyt drobiazgowa i techniczna.

Kierując sie powyższymi założeniami zaprezentowanych zostało kilka charakterystycznych grup zadań optymalizacji dyskretnej, takich jak:

- Programowanie całkowitoliczbowe i liniowe, w tym zadania programowania mieszanego, gdzie na część zmiennych został nałożony wymóg całkowitoliczbowości.
- Zadania binarne, które stanowią szczególnie ważną i charakterystyczną grupę zadań optymalizacji dyskretnej. Wymóg binarności zmiennych jest na ogół spowodowany koniecznością modelowania sytuacji wymagających podejmowania decyzji na tak lub na nie. Wśród zadań binarnych wyróżniono zadania załadunku, przydziału, skojarzenia; zadania pokrycia, pakowania i rozbicia zbiorów; zadania pakowania zasobników oraz zadania lokalizacyjne.
- Zadania teorii grafów. Dokonano prezentacji podstawowych pojęć z teorii grafów takich jak graf skierowany i nieskierowany, graf ważony, drogi i drzewa w grafach i innych. W postaci zadania programowania liniowego mieszanego całkowitoliczbowego przedstawiono zadanie przepływu w sieci. Jako zadanie programowania binarnego sformułowane zostało zadanie komiwojażera.
- Zadania harmonogramowania, które są szczególnie przydatne przy modelowaniu i optymalizacji procesów produkcji z wykorzystaniem maszyn. Przedstawiono klasyfikację tych problemów jako zadań optymalizacyjnych.

Należy podkreślić, że zamiarem autora nie była drobiazgowa prezentacja zadań optymalizacji dyskretnej. Chodziło raczej o zarysowanie specyfiki optymalizacji dyskretnej jako dziedziny badawczej oraz zaprezentowanie najbardziej charakterystycznych jej zadań. W rozdziałach 5 oraz 6 dokonano szczegółowej analizy dwóch wybranych ważnych zadań optymalizacji dyskretnej: wielowymiarowego zadania załadunku oraz zadania szeregowania prac z terminami zakończenia.

Uwagi końcowe

W monografii rozważono podejście przypadku średniego do analizy zadań oraz algorytmów optymalizacji dyskretnej. Celem monografii było wykazanie, na przykładzie binarnego wielowymiarowego zadania załadunku, zadania szeregowania prac z terminami zakończenia oraz wyników znanych z literatury, że podejście przypadku średniego jest bardzo użytecznym narzędziem analizy zadań i algorytmów optymalizacji dyskretnej.

W monografii dokonano przeglądu dziedziny optymalizacji dyskretnej z zaprezentowaniem najbardziej charakterystycznych zadań, takich jak: programowanie całkowitoliczbwe i liniowe, programowanie binarne, zadania pokrycia, pakowania i rozbicia zbiorów, wybrane zadania teorii grafów oraz zadania harmonogramowania.

Następnie zaprezentowano popularne i powszechnie stosowane techniki rozwiązywania zadań optymalizacji dyskretnej, takie jak: metoda pełnego przeglądu, metoda podziału i oszacowań, programowanie dynamiczne, algorytmy zachlanne, metody programowania liniowego i inne.

Zdefinowane zostały sposoby oceny dokładności pracy algorytmów optymalizacji dyskretnej. Do oceny złożoności obliczeniowej zadań i algorytmów optymalizacji dyskretnej wprowadzono metodologię przypadku najgorszego. Dokonano podziału zadań optymalizacji na umowne kategorie zadań łatwych (należących do klasy $\mathcal{P}$), trudnych (należących do klasy zadań $\mathcal{NP}$-trudnych) oraz szczególnie trudnych (zadań silnie $\mathcal{NP}$-trudnych). Z pewnym uproszczeniem można powiedzieć, że o łatwości rozwiązywania wybranych zadań optymalizacji dyskretnej decyduje istnienie dla nich algorytmów dokładnych o wielomianowej złożoności obliczeniowej.

Jako dopełnienie oraz poszerzenie możliwości poznawczych podejścia przypadku najgorszego zaprezentowano podejście przypadku średniego. Zdefiniowano niezbędne podstawy teoretyczne analizy przypadku średniego oraz dokonano prezentacji wybranych wyników znanych z literatury.

Ogólne rozważania dotyczące zadań optymalizacji dyskretnej, metod ich rozwiązywania, podejścia analizy przypadku najgorszego i średniego do oceny zadań i algorytmów optymalizacji dyskretnej szczegółowo omówiono na przy-

kładzie wielowymiarowego binarnego zadania załadunku, z odrębnym rozpatrzeniem przypadku jednowymiarowego oraz zadania szeregowania prac z terminami zakończenia.

Dla rozważanych modeli losowych zadań, które uzyskano przyjmując założenie, że współczynniki funkcji celu i lewych stron ograniczeń są realizacjami zmiennych losowych o rozkładzie równomiernym w przedziale $(0, 1]$, uzyskano szereg ciekawych wyników analizy przypadku średniego. Najważniejszym z nich było wykazanie, że wartości rozwiązań optymalnych dla całych losowych klas zadań dążą do swoich wartości oczekiwanych - deterministycznych funkcji: rozmiaru zadania n (liczby zmiennych decyzyjnych), liczby ograniczeń m (w przypadku binarnego wielowymiarowego zadania załadunku), oraz wartości prawych stron ograniczeń. Z przeprowadzonych rozważań wynika istotny wpływ wartości i wzajemnych uwarunkowań wektorów prawych stron ograniczeń na asymptotyczny wzrost wartości rozwiązań optymalnych jako funkcji rozmiaru zadania n.

W przypadku binarnego wielowymiarowego zadania załadunku można zauważyć, że liczba ograniczeń m ma bardzo duży wpływ na asymptotyczny wzrost wartości rozwiązań optymalnych jako funkcji rozmiaru zadania n, szczególnie w przypadku funkcyjnej zależności wartości prawych stron ograniczeń od m oraz małych wartości prawych stron ograniczeń $b_j(n)$, $j = 1, \ldots, m$. Dla dużych wartości $b_j(n)$, $j = 1, \ldots, m$, zależność od m ulega znacznemu osłabieniu, a przy $b_j(n) \approx n/2$ praktycznie zanika.

Powszechnie stosowaną w praktyce obliczeniowej metodą służącą do oceny algorytmów przybliżonych jest testowanie ich na zadaniach generowanych losowo. Uzyskane wyniki są następnie poddawane analizie statystycznej. Łatwo jest zauważyć, że powszechnie stosowane generatory zadań losowych są praktycznie tożsame z rozważanymi w monografii losowymi modelami zadań. Wyniki analizy przypadku średniego są więc w wielu przypadkach potwierdzeniem i teoretycznym uzasadnieniem wyników eksperymentalnych.

Co więcej, w uzasadnionych przypadkach wyniki analizy przypadku średniego mogą wyeliminować konieczność przeprowadzania eksperymentu obliczeniowego testującego algorytmy dla zadań optymalizacji dyskretnej. W takim przypadku w oczywisty sposób jest oszczędzany czas badaczy oraz zmniejszane wykorzystanie zasobów komputerowych.

W monografii wykazano, że bardzo proste algorytmy heurystyczne, o liniowej złożoności obliczeniowej, nie mające nawet gwarancji uzyskania rozwiązań dopuszczalnych zadań, są asymptotycznie optymalne w średnim przypadku. Wyniki tego typu są w oczywisty sposób odmienne od wyników analizy przypadku najgorszego i dlatego stanowią ich wartościowe uzupełnienie.

Pogląd, że analiza przypadku średniego może zastąpić analizę przypadku

najgorszego jest w odczuciu autora błędny. Zarówno analiza przypadku najgorszego, jak również analiza przypadku średniego mają swoją specyfikę oraz uzyskują wartościowe wyniki i oceny. Dopiero zapoznanie się z wynikami analiz różnego rodzaju pozwala wyrobić sobie możliwie najbardziej obiektywny pogląd na różne aspekty analizowanego zadania lub algorytmu optymalizacji dyskretnej.

Należy również podkreślić, że wyniki analizy przypadku średniego są prawdziwe tylko dla rozważanych losowych klas zadań. Należy więc zachować szczególną ostrożność przy próbach uogólniania uzyskanych wyników na inne klasy zadań, gdyż może to prowadzić do fałszywych i nieuzasadnionych wniosków.

Istotnym wyzwaniem badawczym pozostaje nadal przeprowadzenie analizy przypadku średniego dla szczególnie trudnych zadań optymalizacji dyskretnej. Dobrym przykładem jest zadanie programowania całkowitoliczbowego, patrz podrozdział 1.2 oraz wzór (1.2). W przypadku zadania programowania całkowitoliczbowego postać funkcji Lagrange'a oraz zadania dualnego nie sprzyja zastosowaniu technik i oszacowań wykorzystanych w podrozdziałach 5.2 oraz 6.2.

Innym ważnym celem przyszłych prac badawczych wydaje się rozważenie bardziej realistycznych modeli zadań, co w szczególności może wymagać zastosowania złożonych rozkładów prawdopodobieństwa zmiennych losowych opisujących charakterystyki analizowanych zadań. Również w tym przypadku oczekiwane jest uzyskanie analitycznych wyników o podobnym charakterze jak wyniki zawarte w podrozdziałach 5.5 oraz 6.3 niniejszej monografii.

Literatura

[1] R. Aboudi, K. Jörnsten. Tabu search for general zero-one integer programs using the pivot and complement heuristic. *ORSA Journal on Computing*, 6:82–93, 1994.

[2] A.V. Aho, J.E. Hopcroft, J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA, 1974.

[3] G. d' Atri. Probabilistic analysis of the knapsack problem. Working Paper 7, Groupe de Récherche 22, Centre National de la Récherche Scientifique, Paris, 1978.

[4] G. Ausiello, A. Marchetti-Spaccamela, M. Protasi. Probabilistic analysis of the solution of the knapsack problem. W: *Proceedings of the Tenth IFIP Conference*, ss. 557–565, New York, 1982. Springer.

[5] I. Averbakh. Probabilistic properties of the dual structure of the multidimensional knapsack problem and fast statistically effcient algorithms. *Mathematical Programming*, 65:311–330, 1994.

[6] E. Balas. An additive algorithm for solving linear programs with zero-one variables. *Operations Research*, 13:517–546, 1965.

[7] E. Balas, C.H. Martin. Pivot and complement - a heuristic for 0-1 programming. *Management Science*, 26:86–96, 1980.

[8] E. Balas, E. Zemel. An algorithm for large zero-one knapsack problems. *Operations Research*, 28:1130–1154, 1980.

[9] R. Battiti, G. Tecchiolli. Local search with memory: benchmarking RTS. *OR Spektrum*, 17:67–86, 1995.

[10] J. Beardwood, J. Halton, J.M. Hammersley. The shortest path through many points. *Proc. Cambridge Philos. Soc.*, 55:299–327, 1959.

[11] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

[12] M. Bertocchi, L. Słomiński, J. Sobczyńska. Probabilistic and deterministic local search for solving the binary multiknapsack problem. *Optimization*, 33:155–166, 1995.

[13] J. Błażewicz, K.H. Ecker, E. Pesch, G. Schmidt, J. Węglarz. *Scheduling Computer and Manufacturing Processes*. Springer Verlag, Berlin, Heidelberg, 1996.

[14] J. Błażewicz, J.K. Lenstra, A.H.G. Rinnooy Kan. Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics*, 5:11–24, 1983.

[15] K.H. Borgwardt. The average number of steps required by simplex-method is polynomial. *Zeitschrift für Operations Research*, 26:157–177, 1982.

[16] I.N. Bronsztejn, K.A. Siemiendiajew. *Matematyka, poradnik encyklopedyczny*. Państwowe Wydawnictwo Naukowe, Warszawa, 1973.

[17] A.V. Cabot. An enumeration algorithm for knapsack problems. *Operations Research*, 18:306–311, 1970.

[18] P.M. Camerini, F. Maffioli, C. Vercellis. Multi-constrained matroidal knapsack problems. *Mathematical Programming*, 45:211–231, 1989.

[19] V. Cerny. A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. Preprint, Institute of Physics and Biophysics, Comenius University, Bratislava, 1982.

[20] L.G. Chaczian. Wielomiananowy algorytm programowania liniowego. *Dokl. Akad. Nauk SSSR, Nova Seria*, 244(5):1093–1096, 1979. (w języku rosyjskim).

[21] I. Charon, O. Hudry. The noising method: a new method for combinatorial optimization. *Operations Research Letters*, 14:133–137, 1993.

[22] P.C. Chu, J.E. Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4(1):63–86, 1998.

[23] E.G. Coffman Jr, M.R. Garey, D.S. Johnson. Approximation algorithm for bin-packing - an updated survey. W: G. Ausiello, M. Lucertini, P. Serafini, editors, *Algorithm Design for Computer System Design*, ss. 49–106, New York, 1984. Springer.

[24] E.G. Coffman Jr, G.S. Lueker. *Probabilistic Analysis of Packing and Partitioning Algorithms.* Wiley-Interscience, New York, Chichester, Brisbane, Toronto, Singapore, 1991.

[25] E.G. Coffman Jr, G.S. Lueker, A.H.G. Rinnooy Kan. Asymptotic methods in the probablistic analysis of sequencing and packing heuristics. *Management Science*, 34:266–290, 1988.

[26] S.A. Cook. The complexity of theorem-proving procedures. W: *Proc. 3rd Annu. ACM Symp. On Theory of Computing*, ss. 151–158, New York, 1971. ACM Press.

[27] T.H. Cormen, C.E. Leiserson, R.L. Rivest. *Wprowadzenie do algorytmów.* Wydawnictwa Naukowo Techniczne, Warszawa, 1997.

[28] Y. Crama, J.B. Mazzola. On the strength of relaxations of multidimensional knapsack problems. *INFOR*, 32:219–225, 1994.

[29] F. Dammeyer, S. Voss. Dynamic tabu list management using reverse elimination method. *Annals of Operations Research*, 41:31–46, 1993.

[30] G.B. Dantzig. Discrete variable problems. *Operations Research*, 5:266–277, 1957.

[31] G.B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, Princeton, NJ, 1963.

[32] A. Drexl. A simulated annealing approach to the multiconstraint zero-one knapsack problem. *Computing*, 40:1–8, 1988.

[33] K. Dudziński, K. Szkatuła. A note on sequencing jobs with deadlines problem. *European Journal of Operational Research*, 59:333–336, 1992.

[34] G. Dueck. New optimization heuristics. *Journal of Computational Physics*, 104:86–92, 1993.

[35] G. Dueck, T. Schuer. Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90:161–175, 1990.

[36] A.E. Eiben, E.H.L. Aarts, K.H. Van Hee. Global convergence of genetic algorithms: A markov chain analysis. *Lecture Notes in Computer Science*, 496:4–9, 1991.

[37] H. Everett. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11:399–417, 1963.

[38] W. Feller. *Wstęp do rachunku prawdopodobieństwa, tom II.* Państwowe Wydawnictwo Naukowe, Warszawa, 1981.

[39] J.F. Fontanari. A statistical analysis of the knapsack problem. *Journal of Physics A - Mathematical and General*, 28:4751–4759, 1995.

[40] G.E. Fox, G.D. Scudder. A heuristic with tie breaking for certain 0-1 integer programming models. *Naval Research Logistics Quarterly*, 32:613–623, 1985.

[41] A. Freville, G. Plateau. Heuristics and reduction methods for multiple constraints 0-1 linear programming problems. *European Journal of Operational Research*, 24:206–215, 1986.

[42] A. Freville, G. Plateau. Hard 0-1 multiknapsack test problems for size reduction methods. *Investigacion Operativa*, 1:251–270, 1990.

[43] A. Freville, G. Plateau. An efficient preprocessing procedure for the multidimensional 0-1 knapsack problem. *Discrete Applied Mathematics*, 49:189–212, 1994.

[44] A. Freville, G. Plateau. The 0-1 bidimensional knapsack problem: toward an efficient high-level primitive tool. *Journal of Heuristics*, 2:147–167, 1997.

[45] A.M. Frieze, M.R.B Clarke. Approximation algorithms for the m-dimensional 0-1 knapsack problem: Worst case and probabilistic analysis. *European Journal of Operational Research*, 15:100–109, 1984.

[46] M.R. Garey, R.L. Graham, D.S. Johnson, A. Yao. Resource constrained scheduling as generalized bin packing. *J. Combin. Theory A*, (21):257–298, 1976.

[47] M.R. Garey, D.S. Johnson. Strong NP-completeness results: motivation, examples and implications. *Journal of the Association of Computer Machinery*, 25:499–508, 1978.

[48] M.R. Garey, D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

[49] R.S. Garfinkel, G.L. Nemhauser. *Programowanie całkowitoliczbowe.* Państwowe Wydwnictwo Naukowe, Warszawa, 1978.

[50] S.L. Gass. *Programowanie liniowe.* PWN, Warszawa, 1976.

[51] B. Gavish, H. Pirkul. *Allocation of Databases and Processors in a Distributed Computing System.* North-Holland, Amsterdam, 1982.

[52] B. Gavish, H. Pirkul. Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality. *Mathematical Programming*, 31:78–105, 1985.

[53] S. Van de Geer, L. Stougie. On rates of convergence and asymptotic normality in the multiknapsack problem. *Mathematical Programming*, 51:349–358, 1991.

[54] P.C. Gilmore, R.E. Gomory. The theory and computation of knapsack functions. *Operations Research*, 14:1045–1075, 1966.

[55] F. Glover. Heuristic for integer programming using surrogate constraints. *Decision Sciences*, 8:156–160, 1977.

[56] F. Glover. Tabu search: a tutorial. *Interfaces*, 20(4):74–94, 1991.

[57] F. Glover. Optimization by ghost image processes in neural networks. *Computers and Operations Research*, 21:801–822, 1994.

[58] F. Glover, G.A. Kochenberger. Critical event tabu search for multidimensional knapsack problems. W: I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics: Theory and Applications*, ss. 407–427. Kluwer Academic Publishers, 1996.

[59] F. Glover, M. Laguna. *Tabu Search.* Kluwer, Boston, 1997.

[60] A.V. Goldberg, A. Marchetti-Spaccamela. On finding the exact solutions of a 0-1 knapsack problem. W: *Proceedings of the 16th ACM Symposium on Theory of Computing*, ss. 359–368, New York, 1984. Association for Computing Machinery.

[61] M.R. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling theory: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[62] M. Grötschel, L. Lovász. Combinatorial optimization. W: R. Graham, M. Grötschel, L. Lovász, editors, *Handbook of Combinatorics*, ss. 1541–1597. Elsevier Science B.V., 1995.

[63] S. Hanafi, A. Freville. An efficient tabu search approach for the 0-1 multidimensional knapsack problem. *European Journal of Operational Research*, to appear.

[64] S. Hanafi, A. Freville, A.El. Abedellaoui. Comparison of heuristics for the 0-1 multidimensional knapsack problem. W: I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics: Theory and Applications*, ss. 449–465. Kluwer Academic Publishers, 1996.

[65] D. Hausman, R. Kannan, B. Korte. Exponential lover bounds on a class of knapsack algorithms. *Mathematics of Operations Research*, 6:225–232, 1981.

[66] F.S. Hiller. Efficient heuristic procedures for integer linear programming with an interior. *Operations Research*, 17:600–637, 1969.

[67] D.S. Hochbaum. A nonlinear knapsack problem. *Operations Research Letters*, 17:103–110, 1995.

[68] A. Hoff, A. Løkketagen, I. Mittet. Genetic algorithms for 0/1 multidimensional knapsack problems. Working paper, Molde College, Britvein 2, Molde, Norway, 1996.

[69] J.H. Holland. Adaptation in natural and artificial systems. Technical report, The University of Michigan Press, Ann Arbor, 1975.

[70] D.S. Johnson. *Near-optimal allocation algorithms.* PhD thesis, MIT, Cambridge, MA, 1973.

[71] D.S. Johnson. The NP completeness column: an ongoing guide. *J. Algorithms*, 5:284–299, 1984.

[72] D.S. Johnson. The NP-completeness column: an ongoing guide. *J. Algorithms*, 9:426–444, 1988.

[73] D.S. Johnson. Local optimization and the travelling salesman problem. W: *Proc. 17th Coll. On Automata, Languages and Programming*, ss. 446–461, Heidelberg, 1990. Springer.

[74] D.S. Johnson. The NP-completeness column: an ongoing guide. *J. Algorithms*, 13:502–524, 1992.

[75] D.S. Johnson, C.R. Aragon, L.A. McGeoch, C. Schevon. Optimization by simulated annealing: an experimental evaluation; part I, graph partitioning. *Operational Research*, 37:865–892, 1989.

[76] D.S. Johnson, C.R. Aragon, L.A. McGeoch, C. Schevon. Optimization by simulated annealing; an experimental evaluation, part II, graph coloring and number partitioning. *Operations Research*, 39:378–406, 1991.

[77] R. Kannan, B. Korte. Approximative combinatorial algorithms. W: *Mathematical Programming*, ss. 195–248, Amsterdam, New York, 1984. North Holland.

[78] N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4:375–395, 1984.

[79] R.M. Karp. Reducibility among combinatorial problems. W: R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, ss. 85–103. Plenum Press, New York, 1972.

[80] R.M. Karp. The probablilistic analysis of some combinatorial search algorithms. W: J.F. Traub, editor, *Algorithms and Complexity: New Directions and Recent Results*, ss. 1–19. Academic Press, 1976.

[81] R.M. Karp. Probabilistic analysis of partitioning algorithms for the travelling salesman problem in the plane. *Mathematics of Operations Research*, 2:209–224, 1977.

[82] R.M. Karp. A patching algorithms for the nonsymmetric travelling salesman problem. *SIAM Journal on Computing*, 8:561–573, 1979.

[83] R.M. Karp, J.K. Lenstra, C.J.H. McDiarmid, A.H.G. Rinnooy Kan. Probablistic analysis of combinatorial algorithms. W: M. OhEigeartaigh, J.K. Lenstra, A.G.H. Rinnooy Kan, editors, *Combinatorial Optimization: Annotated Bibliographies*, ss. 52–88. Wiley-Interscience, New York, Chichester, 1985.

[84] S. Khuri, T. Bäck, J. Heitkötter. The zero/one multiple knapsack problem and genetic algorithms. W: *Proceedings of the 1994 ACM Symposium on Applied Computing (SAC'94)*, ss. 188–193. ACM Press, 1994.

[85] S. Kirkpatrick, C.D. Gelatt, M. Vecchi. Optimization by simmulated annealing. *Science*, 220:621–680, 1983.

[86] V. Klee, G.J. Minty. How good is the simplex algorithm. W: O. Shisba, editor, *Inequalities III*, ss. 159–175. Academic Press, 1972.

[87] G.A. Kochenberger, B.A. McCarl, F.P. Wyman. A heuristic for general integer programming. *Decision Sciences*, 5:36–44, 1974.

[88] B. Korte, D. Hausmann. An analysis for the greedy algorithm for independence systems. *Ann. Discrete Math.*, 2:65–74, 1978.

[89] B. Korte, L. Lovász, R. Schrader. *Greedoids.* Springer, Heidelberg, 1991.

[90] J.B. Kruskal. On the shortest spanning subtree of a graph and the travelling salesman problem. *Proc. Amer. Math. Soc.*, 2:48–50, 1956.

[91] J.L. Kulikowski. *Zarys Teorii Grafów.* Państwowe Wydawnictwo Naukowe, Warszawa, 1986.

[92] E.L. Lawler, J.M. Moore. A functional equation and its application to resource alocation and sequencing problems. *Management Science*, 16:77–84, 1969.

[93] J.S. Lee, M. Guignard. An approximate algorithm for multidimensional zero-one knapsack problems - a parametric approach. *Management Science*, 34:402–410, 1988.

[94] T.-E. Lee, G.-T. Oh. The asymptotic value-to-capacity ratio for the multi-class stochastic knapsack problem. *European Journal of Operational Research*, 103:584–594, 1997.

[95] L.A. Levin. Average case complete problems. *SIAM J. Comput.*, 15:285–286, 1986.

[96] M. Loève. *Probability Theory I.* Springer Verlag, New York, Heidelberg, Berlin, 1977.

[97] A. Løkketagen, K. Jörnsten, S. Storøy. Tabu search within a pivot and complement framework. *International Transactions of Operations Research*, 1:305–316, 1994.

[98] A. Løkketangen, F. Glover. Probabilistic move selection in tabu search for zero-one mixed integer programming problems. W: I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics*, ss. 467–487. Kluwer Academic Publishers, 1996.

[99] A. Løkketangen, F. Glover. Solving zero-one mixed integer programming problems using tabu search. *European Journal of Operational Research*, to appear.

[100] G.S. Lorie, L. Savage. Three problems in capital rationing. *Journal of Business*, 28:229–239, 1955.

[101] R. Loulou, E. Michaelides. New greedy-like heuristics for the multidimensional 0-1 knapsack problem. *Operations Research*, 27:1101–1114, 1979.

[102] G.S. Lueker. On the average difference between the solution to linear and integer knapsack problems. W: *Applied Probability-Computer Science, the Interface, Vol 1*, ss. 489–504. Birkhauser, Basel, 1982.

[103] M.J. Magazine, O. Oguz. A heuristic algorithm for the multidimensional zero-one knapsack problem. *European Journal of Operational Research*, 16:319–326, 1984.

[104] J.W. Mamer, K.E. Schilling. On the growth of random knapsacks. *Discrete Applied Mathematics*, 28:223–230, 1990.

[105] S. Martello, P. Toth. *Knapsack Problems: Algorithms and Computer Implementations.* Wiley & Sons, 1990.

[106] M. May, K. Szkatuła. On the bipartite crossing number. *Control and Cybernetics*, 17:85–98, 1988.

[107] M. Meanti, A.H.G. Rinnooy Kan, L. Stougie, C. Vercellis. A probabilistic analysis of the multiknapsack value function. *Mathematical Programming*, 46:237–247, 1990.

[108] M. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller. Equation of state calculations by fast computing machines. *J. Chemical Physics*, 21:1087–1092, 1953.

[109] G.L. Nemhauser, Z. Ullmann. Discrete dynamic programming and capital allocation. *Management Science*, 15:494–505, 1969.

[110] G.L. Nemhauser, L.A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley & Sons Inc., New York, 1988.

[111] C.H. Papadimitriou, K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Englewood Cliffs, 1982.

[112] R.G. Parker, R.L. Rardin. *Discrete Optimization.* Academic Press, Boston, 1988.

[113] H. Pirkul. A heuristic solution procedure for the multiconstraint zero-one knapsack problem. *Naval Research Logistics*, 34:161–172, 1987.

[114] C.N. Potts, L.N. Van Wassenhove. Algorithms for scheduling a single machine to minimize the weighted number of late jobs. *Management Science*, 34:843–858, 1988.

[115] A.H.G. Rinnooy Kan. Probabilistic analysis of algorithms. *Annals of Discrete Mathematics*, 31:365–384, 1987.

[116] A.H.G. Rinnooy Kan, L. Stougie. Probabilistic analysis of algorithms. W: J.K. Lenstra, H. Tijms, T. Volgenant, editors, *Twenty Five Years of Operations Research in the Netherlands*, ss. 104–121. Math. Centrum Wish. Inform, Amsterdam, 1989.

[117] A.H.G. Rinnoy Kan, L. Stougie, C. Vercellis. A class of generalized greedy algorithms for the multi-knapsack problem. *Discrete Applied Mathematics*, 42:279–290, 1993.

[118] G. Rudolph, J. Sprave. A cellular genetic algorithm with self-adjusting acceptance threshold. W: *Proceedings of the First IEE/IEEE Internatinal Confernce on Genetic Algorithms in Engineering Systems: Innovations and Applications*, ss. 365–372, London, 1995. IEE.

[119] G. Rudolph, J. Sprave. Significance of locality and selection pressure in the grand deluge evolutionary algorithm. W: H.M. Voigt, W. Ebeling, I. Rechenberg, H.P. Schwefel, editors, *Parallel Problem Solving from Nature IV. Proceedings of the International Conference on Evolutionary Computation*, ss. 686–694. Springer, Lecture Notes in Computer Science, 1996.

[120] S.K. Sahni. Algorithms for scheduling independent jobs. *Journal of ACM*, 23:116–127, 1976.

[121] K.E. Schilling. The growth of m-constraint random knapsacks. *European Journal of Operational Research*, 46:109–112, 1990.

[122] K.E. Schilling. Random knapsacks with many constraints. *Discrete Applied Mathematics*, 48:163–174, 1994.

[123] S. Senju, Y. Toyoda. An approach to linear programming with 0-1 variables. *Management Science*, 15:196–207, 1968.

[124] W. Shih. A branch and bound method for the multiconstraint zero-one knapsack problem. *Journal of the Operational Research Society*, 30:369–378, 1979.

[125] S. Smale. On the average number of steps of the simplex method of lineaar programming. *Mathematical Programming*, 27:241–262, 1983.

[126] A.L. Soyster, B. Lev, W. Slivka. Zero-one programming with many variables and few constraints. *European Journal of Operational Research*, 2:195–201, 1978.

[127] J.M. Steele. Probabilistic and worst case analyses of classical problems of combinatorial optimization in Euclidean space. *Marhematics of Operations Research*, 15(4):749–770, 1990.

[128] J.M. Steele. *Probability Theory and Combinatorial Optimization.* Society for Industrial and Applied Mathematics, Philadephia, 1996.

[129] H.I. Stern, Z. Avivi. The selection and scheduling of textile orders with due dates. *European Journal of Operational Research*, 44:11–16, 1990.

[130] A. Straszak, M. Libura, J. Sikorski, D. Wagner. Computer-assisted constrained approval voting. *Group Decision and Negotiation*, 2:375–385, 1993.

[131] M.M. Sysło, N. Deo, J.S. Kowalik. *Discrete Optimization Algorithms.* Prentice-Hall Inc., Englewood Cliffs, 1983.

[132] K. Szkatuła. Probabilistic analysis of the sequencing jobs with deadlines problem and the threshold algorithm. W: G.Menga and V.Kempe, editors, *Proceedings of the Workshop on Informatics in Industrial Automation*, ss. 113–124, Berlin, 1989.

[133] K. Szkatuła. Analiza probabilistyczna wielowymiarowych, ograniczonych całkowitoliczbowych zadań załadunku. *Techniczeskaja Kibernetika*, 4:236–241, 1994. (w języku rosyjskim).

[134] K. Szkatuła. On the asymptotical growth of multi-constraint entirely random knapsacks. W: *Systems Analysis and Decision Support in Economics and Technology*, ss. 299–304, Warszawa, 1994.

[135] K. Szkatuła. On the growth of entirely random multi-constraint 0-1 knapsacks. W: *BOS 93, Trzecia Konferencja Badań Operacyjnych I Systemowych*, ss. 205–304, Warszawa, 1994.

[136] K. Szkatuła. On the growth of multi-constraint random knapsacks with various right-hand sides of the constraints. *European Journal of Operational Reserch*, 73:199–204, 1994.

[137] K. Szkatuła. Analiza średniego przypadku m-wymiarowych zadań załadunku. W: *Wspomaganie Decyzji, Systemy Eksperckie*, ss. 195–200, Warszawa, 1995.

[138] K. Szkatuła. The growth of multi-constraint random knapsacks with large right-hand sides of the constraints. *Operations Research Letters*, 21:25–30, 1997.

[139] K. Szkatuła. The growth of multi-constraint random knapsacks with mixed right-hand-sides of the constraints. Instytut Badań Systemowych PAN, Warszawa, 1997.

[140] K. Szkatuła. Random sequencing jobs with deadlines problem: growth of the optimal solutions values. *European Journal of Operational Research*, 109:160–169, 1998.

[141] K. Szkatuła, M. Libura. Probabilistic analysis of simple algorithms for binary knapsack problem. *Control and Cybernetics*, 12:147–157, 1983.

[142] K. Szkatuła, M. Libura. On probabilistic properties of greedy like algorithms for the binary knapsack problem. W: *Stochastics in Combinatorial Optimization*, Singapore, 1987. World Scientific Publishing.

[143] J. Thiel, S. Voss. Some experiences on solving multiconstraint zero-one knapsack problems with genetic algorithms. *INFOR*, 32:226–242, 1994.

[144] Y. Toyoda. A simplified algorithm for obtaining approximate solutions to zero-one programming problems. *Management Science*, 21:1417–1427, 1975.

[145] A. Volgenant, J.A. Zoon. An improved heuristic for multidimensional 0-1 knapsack problems. *Journal of the Operational Research Society*, 41:963–970, 1990.

[146] S. Walukiewicz. *Programowanie dyskretne.* Państwowe Wydawnictwo Naukowe, Warszawa, 1986.

[147] B. Weide. *Statistical methods in algorithm design and analysis.* PhD thesis, Carnegie-Mellon University, Pittsbutgh, PA, 1978. CMU-CS 78-142.

[148] H.M. Weingartner. *Mathematical Programming and the Analysis of Capital Budgeting Problems.* Markham Publishing, Chicago, 1967.

[149] H.M. Weingartner, D.N. Ness. Methods for the solution of the multidimensional 0/1 knapsack problem. *Operations Research*, 15:83–103, 1967.

[150] S.H. Zanakis. Heuristic 0-1 linear programming: an experimental comparison of three methods. *Management Science*, 24:91–104, 1977.

[151] K. Zorychta, W. Ogryczak. *Programowanie liniowe i całkowitoliczbowe: metoda podziału i ograniczeń.* Wydawnictwa Naukowo Techniczne, Warszawa, 1981.

Analiza przypadku średniego w optymalizacji dyskretnej

Wielowymiarowe zadanie załadun oraz zadanie szeregowania prac

Krzysztof Szkatuła

W monografii omawiane są wybrane zadania optymalizacji dyskretnej i metody ich rozwiązywania oraz problematyka analizy złożoności obliczeniowej zadań i algorytmów.

Głównym celem książki jest wykazanie, na przykładzie wielowymiarowego zadania załadunku i zadania szeregowania prac, że przy zastosowaniu niezbyt zaawansowanego aparatu rachunku prawdopodobieństwa można uzyskać wartościowe wyniki analizy przypadku średniego w optymalizacji dyskretnej.

Wyniki zaprezentowane w monografii potwierdzają, że analiza przypadku średniego jest bardzo efektywnym narzędziem wspomagającym i uzupełniającym powszechnie stosowane do analizy zadań i algorytmów optymalizacji dyskretnej podejście przypadku najgorszego.

ISBN 83-85847-39-1
ISSN 0208-8029

W celu uzyskania bliższych informacji i zakupu dodatkowych egzemplarzy prosimy o kontakt z Instytutem Badań Systemowych PAN
ul. Newelska 6, 01-447 Warszawa
tel. 37-35-78 w. 241 e-mail: kotuszew@ibspan.waw.pl