STUDIA INFORMATICA 2011
Volume 32 Number 4A (100)

Jacek WIDUCH
Politechnika Śląska, Instytut Informatyki

ALGORYTMY OPTYMALIZACJI TRAS PRZEJAZDU POJAZDÓW

Streszczenie. Wyznaczanie tras przejazdu jest jednym z badanych problemów transportowych. W niniejszej pracy przedstawiono problem wyznaczania połączeń w sieciach komunikacyjnych. Problem ten jest przykładem zadania optymalizacji wielokryterialnej, którego rozwiązaniem jest zbiór rozwiązań niezdominowanych. Przedstawione zostały 4 algorytmy, umożliwiające wyznaczenie połączeń należących do zbioru rozwiązań niezdominowanych. Dodatkowo omówiono inne problemy związane z wyznaczaniem tras przejazdu, które były badane przez pracowników Instytutu Informatyki.

Słowa kluczowe: problem transportowy, problem NP–zupełny, optymalizacja wielokryterialna, zbiór rozwiązań niezdominowanych, metoda skalaryzacji, dwukryterialny problem wyznaczania najkrótszej ścieżki, drzewo ścieżek

OPTIMIZATION ALGORITHMS FOR VEHICLE ROUTING PROBLEM

Summary. The routing problem is one of the studied transportation problems. In the paper the communication networks routing problem is presented. This problem is an example of mulitcriteria optimization problem which the solution is the set of non–dominated solutions. Four algorithms for determination routes belong to the set of non–dominated solutions are shown. Additionally, other routing problems that have been researched by staff of the Institute of Informatics are discussed.

Keywords: transportation problem, NP–complete problem, multicriteria optimization, set of non–dominated solutions, scalarization method, bicriterion shortest path problem, paths tree

1. Wstęp

Problem wyznaczania tras przejazdu jest jednym z badanych problemów transportowych. Są rozważane m.in. wyznaczenie trasy przejazdu o minimalnej długości, minimalnym koszcie

lub czasie przejazdu itp. Podczas wyznaczania tras przejazdu sieć transportowa jest opisywana za pomocą skierowanego grafu $G = (V, E)$ z funkcją wagową $w : E \rightarrow R$, a problem wyznaczenia trasy sprowadza się do wyznaczenia w grafie G ścieżki o minimalnej wadze. W niektórych zastosowaniach uwzględnienie tylko jednej wagi jest niewystarczające, gdyż wyznaczone w ten sposób rozwiązanie nie uwzględnia wszystkich istotnych cech badanego problemu. Stąd wprowadza się k ($k > 1$) funkcji wagowych, które są jednocześnie uwzględniane podczas wyznaczania ścieżki w grafie. Jeżeli $k = 2$, wówczas mówi się o dwukryterialnym problemie wyznaczania najkrótszej ścieżki. Znanych jest wiele algorytmów rozwiązywania tego problemu, które można podzielić na: algorytmy korygowania etykiet [3, 6, 10, 37], algorytmy nadawania etykiet [19, 27, 42], algorytmy dwufazowe [30], algorytmy działające na bazie wyznaczenia K najkrótszych ścieżek [4] oraz pozostałe [14, 24, 25, 26, 29, 33]. We wszystkich wymienionych algorytmach zakłada się stałe wagi łuków grafu.

W niniejszej pracy, jako przykładowy dwukryterialny problem wyznaczania najkrótszej ścieżki w grafie, przedstawiono problem wyznaczania połączeń w sieciach komunikacyjnych. Rozważany problem polega na wyznaczeniu połączeń komunikacyjnych dla zadanej pary przystanków i ustalonej godziny rozpoczęcia podróży. Podczas wyznaczania połączeń należy dążyć do jednoczesnej minimalizacji czasu i kosztu przejazdu. Badany problem jest przykładem zadania optymalizacji wielokryterialnej, którego rozwiązaniem w ogólnym przypadku jest zbiór rozwiązań niezdominowanych.

Niniejsza praca stanowi podsumowanie prac badawczych, jakie zostały wykonane w latach 2001–2011, związanych z rozwiązaniem problemu wyznaczania połączeń w sieciach komunikacyjnych. W wyniku przeprowadzonych badań zostały opracowane algorytmy rozwiązywania wymienionego problemu, które przedstawiono w pracach [46, 47, 49, 50, 51] oraz skrótowo opisano w niniejszej pracy w rozdziale 3. W rozdziale tym opisano algorytmy korygowania etykiet z usuwaniem rozwiązań częściowych (punkt 3.3) oraz z pamiętaniem rozwiązań częściowych (punkt 3.4). W punkcie 3.5 przedstawiono algorytm wyznaczania połączeń na podstawie wyznaczenia K–tej najkrótszej ścieżki, a w punkcie 3.6 algorytm wyznaczania połączeń z użyciem metody skalaryzacji.

Problem wyznaczania połączeń w sieciach komunikacyjnych nie jest jedynym problemem związanym z wyznaczaniem tras przejazdu, który był badany przez pracowników Instytutu Informatyki. W rozdziale 3 zostały opisane podobne problemy, które podlegały badaniom. Zostały opisane: problem doboru trasy transportu przesyłki realizowanej przez firmę kurierską, problem dostawy z oknami czasowymi (ang. *Vehicle Routing Problem with Time Windows*) oraz problem dostawy, w którym nie są uwzględniane okna czasowe (ang. *Vehicle Routing Problem*).

2. Optymalizacja wielokryterialna

Celem optymalizacji jest wybór spośród wielu możliwych rozwiązań jednego najlepszego. Wybór rozwiązania odbywa się na podstawie funkcji kryterialnej, która w wyniku przeprowadzenia procesu optymalizacji przyjmuje wartość optymalną. Dzięki zastosowaniu tylko jednej funkcji kryterialnej metody wyznaczania rozwiązania nie są złożone, jednak wyznaczone w ten sposób rozwiązanie może nie uwzględniać wszystkich istotnych cech przedmiotu będącego celem badań.

W zadaniach optymalizacji wielokryterialnej danych jest k ($k > 1$) funkcji kryterialnych f_i ($i = 1, \ldots, k$), na podstawie których jest dokonywana ocena i wybór rozwiązań [5, 32, 38, 43, 44]. Wyróżnia się funkcje kryterialne minimalizowane lub maksymalizowane. W pierwszym przypadku dąży się do wyznaczenia rozwiązania, dla którego funkcja przyjmuje wartość minimalną, a w drugim wartość maksymalną. W większości przypadków nie istnieje jedno rozwiązanie, dla którego wszystkie funkcje kryterialne przyjmują wartości optymalne, stąd rozwiązaniem zadania optymalizacji wielokryterialnej jest zbiór rozwiązań, zwany zbiorem rozwiązań niezdominowanych (rozwiązań Pareto optymalnych [31]).

Niech będzie danych k minimalizowanych funkcji kryterialnych $f_1, ..., f_k$. Rozwiązanie X dominuje rozwiązanie Y, co jest oznaczane przez $X \succ Y$, jeżeli spełniona jest zależność (1). Zbiór rozwiązań niezdominowanych S_{ND} jest zbiorem takich rozwiązań, dla których nie istnieje rozwiązanie, które je dominuje, tj.: $S_{ND} = \{X: \neg\exists Y,\ Y \succ X\}$.

$$\underset{i\in\{1,\ldots,k\}}{\forall} f_i(Y) \le f_i(X) \wedge \underset{j\in\{1,\ldots,k\}}{\exists} f_j(Y) < f_j(X) \tag{1}$$

Jeżeli metody wyznaczania wszystkich rozwiązań niezdominowanych są bardzo złożone, wówczas wyznacza się jedynie niektóre z nich. Jedną z metod umożliwiających wyznaczenie rozwiązań niezdominowanych jest skalaryzacja, w której poszczególnym funkcjom kryterialnym są przyporządkowywane wagi i jest wprowadzana zastępcza funkcja kryterialna F_Z [108]. Niech będzie danych k funkcji kryterialnych $f_1, ..., f_k$, gdzie funkcji f_i ($i = 1, \ldots, k$) jest przyporządkowana waga λ_i. Zastępcza funkcja kryterialna F_Z jest określona zależnością (2), przy czym najczęściej wagi λ_i spełniają zależność (3).

$$F_Z = \sum_{i=1}^{k} \lambda_i f_i \tag{2}$$

$$\sum_{i=1}^{k} \lambda_i = 1 \tag{3}$$

W metodzie skalaryzacji dąży się do wyznaczenia rozwiązania, dla którego funkcja F_Z przyjmuje wartość minimalną. Przez wprowadzenie zastępczej funkcji F_Z jest dokonywana redukcja zadania optymalizacji wielokryterialnej do zadania optymalizacji jednokryterialnej

i istnieje możliwość zastosowania metod rozwiązywania, które są mniej złożone niż w przypadku zadania optymalizacji wielokryterialnej. Przez odpowiedni dobór wartości wag λ_i jest określany wpływ danej funkcji kryterialnej na wyznaczone rozwiązanie, jednak w ogólnym przypadku nie zostaną wyznaczone wszystkie rozwiązania ze zbioru rozwiązań niezdominowanych, a jedynie te, dla których wartość zastępczej funkcji kryterialnej przyjmuje wartość minimalną [106, 108, 109]. Przeprowadzając wielokrotnie proces optymalizacji dla różnych wartości wag λ_i, istnieje możliwość uzyskania kolejnych rozwiązań ze zbioru rozwiązań niezdominowanych.

3. Problem wyznaczania połączeń w sieciach komunikacyjnych

3.1. Sformułowanie problemu

Sieć komunikacyjna zawiera n przystanków $v_1, \ldots, v_n$ i jest podzielona na strefy, co wpływa na sposób wyznaczania kosztu przejazdu. W sieci kursują pojazdy M linii komunikacyjnych o numerach $1, \ldots, M$. Dla każdej linii jest zdefiniowana trasa przejazdu. Przejazd między przystankami odbywa się w określonym kierunku, tzn. jeżeli pojazd danej linii kursuje z przystanku v_i do przystanku v_j, nie oznacza to, że kursuje on w przeciwnym kierunku. Pojazd danej linii może kursować w obydwu kierunkach między przystankami v_i oraz v_j, a trasy przejazdu mogą się różnić. Przystanki należące do trasy danej linii są różne, z wyjątkiem przystanków początkowego i końcowego, które mogą być identyczne.

Pojazd komunikacyjny danej linii kursuje z przystanku początkowego v_0 do przystanku końcowego v_k linii z określoną częstotliwością. Niech trasa przejazdu pojazdu linii i–tej ma postać: $\langle v_0, v_1, \ldots, v_{k-1}, v_k\rangle$. Pojazd i–tej linii wyjeżdża z przystanku v_0 o godzinie T_0, przejeżdża przez przystanki $v_1, \ldots, v_{k-1}$ wyjeżdżając z nich odpowiednio w godzinach $T_0+\delta_0$, $T_0+\delta_1, \ldots, T_0+\delta_{k-2}$ i dojeżdża do przystanku v_k o godzinie $T_0+\delta_{k-1}$. Następny kurs rozpoczyna się o godzinie T_1 ($T_1 = T_0+\beta_0$), tak więc z przystanków $v_1, \ldots, v_{k-1}$ pojazd wyjeżdża w godzinach $T_1+\delta_0$, $T_1+\delta_1, \ldots, T_1+\delta_{k-2}$ dojeżdżając do przystanku v_k o godzinie $T_1+\delta_{k-1}$. Przyjmuje się idealny model sieci, w którym pojazdy kursują zgodnie z rozkładem jazdy oraz pomija się czas wsiadania i wysiadania pasażerów na przystanku. Pojazd i–tej linii komunikacyjnej wykonuje p kursów wyjeżdżając z przystanku początkowego v_0 w godzinach $T_0, \ldots, T_{p-1}$ ($T_0 < \ldots < T_{p-1}$), gdzie $T_j = T_{j-1} + \beta_{j-1}$ ($j = 1, \ldots, p-1$). Rozkład jazdy i–tej linii komunikacyjnej definiuje wartości $T_0, \beta_0, \ldots, \beta_{p-2}, \delta_0, \ldots, \delta_{k-1}$, przy czym $0 < \beta_0 < \ldots < \beta_{p-2}$; $0 < \delta_0 < \ldots < \delta_{k-1}$.

Dla zdefiniowanej sieci komunikacyjnej, rozkładu jazdy i tras przejazdu pojazdów linii komunikacyjnej, dane są przystanki początkowy v_s i końcowy v_e podróży, oraz godzina

rozpoczęcia podróży T_s na przystanku v_s. Należy wyznaczyć trasę przejazdu z przystanku v_s do przystanku v_e minimalizując czas i koszt przejazdu, przy założeniu że wyjazd z przystanku początkowego v_s nie może nastąpić wcześniej niż w godzinie rozpoczęcia podróży T_s. Wyznaczona trasa przejazdu zawiera:

- sekwencję przystanków, przez które następuje przejazd,
- informację o liniach komunikacyjnych, pojazdami których następuje przejazd pomiędzy poszczególnymi przystankami,
- godziny wyjazdów z poszczególnych przystanków należących do trasy przejazdu,
- przystanki ewentualnych przesiadek.

Czas przejazdu jest równy sumie czasów przejazdu pomiędzy kolejnymi przystankami należącymi do trasy, ewentualnego czasu oczekiwania na przystanku v_s oraz ewentualnych czasów oczekiwania na połączenie w przypadku przesiadki.

Koszt przejazdu jest wyznaczany następująco. Koszt przejazdu bez przesiadki w granicach jednej strefy wynosi c_1 ($c_1 > 0$) jednostek, w granicach dwóch stref c_2 ($c_2 > c_1$) jednostek, a w granicach trzech lub więcej stref c_3 ($c_3 > c_2$) jednostek. W przypadku przesiadek łączny koszt przejazdu z przystanku v_s do przystanku v_e jest równy sumie kosztów przejazdu pomiędzy przystankami, na których następuje przesiadka. Niektóre linie komunikacyjne są oznaczone jako pospieszne. Przejazd pojazdem takiej linii odbywa się szybciej, co jest uwzględnione w rozkładzie jazdy, natomiast koszt przejazdu jest dwa razy większy od kosztu przejazdu pojazdem linii zwykłej.

3.2. Analiza problemu

Do opisu sieci komunikacyjnej zostanie użyty skierowany graf $G=(V, E)$ z wagami [8, 20, 23, 34, 53]. Graf G zawiera n wierzchołków, które reprezentują przystanki pojazdów komunikacyjnych[1], a łuki grafu reprezentują z kolei trasy przejazdu pojazdów komunikacyjnych. Łuk $e_i = (v_j, v_k)$ ($e_i \in E$; $v_j, v_k \in V$) odpowiada jednej linii komunikacyjnej, której pojazdy kursują z przystanku v_j do przystanku v_k. Ponieważ między daną parą przystanków mogą kursować pojazdy więcej niż jednej linii komunikacyjnej, więc graf może zawierać łuki równoległe.

Każdy łuk $e_i = (v_j, v_k)$ ($e_i \in E$; $v_j, v_k \in V$) ma 3 wagi: $t(e_i)$, $c(e_i)$ i $l(e_i)$. Waga $t(e_i)$ przyjmuje wartości dodatnie i jest równa: $t(e_i) = T_k - T_j$, gdzie T_k i T_j są godzinami przyjazdu odpowiednio do v_k i v_j. Waga $t(e_i)$ jest więc równa sumie: czasu przejazdu z v_j do v_k oraz czasu oczekiwania na wyjazd w v_j. Waga $c(e_i)$ przyjmuje wartości nieujemne i jest równa $c(e_i) = c_k - c_j$, gdzie c_k jest kosztem przejazdu z v_s do v_k, a c_j jest kosztem przejazdu z v_s do v_j.

Ostatnia waga, tj. $l(e_i)$ opisuje numer linii komunikacyjnej, pojazdem której odbywa się przejazd z v_j do v_k. Jak wykazano w pracach [47, 49], $t(e_i)$ i $c(e_i)$ są wagami o zmiennych wartościach.

Pojedyncze rozwiązanie, tj. połączenie komunikacyjne pomiędzy parą wierzchołków v_s do v_e, jest reprezentowane przez:

- ścieżkę $p_{se} = \langle v_0, (v_0, v_1), v_1, \ldots, v_{m-1}, (v_{m-1}, v_m), v_m \rangle$ w grafie G opisującym sieć komunikacyjną, gdzie $v_0 = v_s$ i $v_m = v_e$,
- godziny odjazdów $T_0, \ldots, T_{m-1}$ z kolejnych wierzchołków należących do ścieżki, gdzie T_i jest godziną odjazdu z v_i ($i = 0, \ldots, m-1$; $v_i \in p_{se}$), a $v_0 = v_s$ i $v_m = v_e$.

Rozwiązaniem częściowym z kolei jest połączenie komunikacyjne z wierzchołka v_s do dowolnego wierzchołka v_i, gdzie $v_i \neq v_e$.

Długość ścieżki p_{se}, oznaczana przez $len(p_{se})$, jest równa liczbie łuków należących do ścieżki. Ścieżka p_{se} ma dwie wagi $T(p_{se})$ i $C(p_{se})$ określone zależnością (4), równe sumie wag t i c łuków należących do ścieżki. Wagi te reprezentują czas i koszt przejazdu z v_s do v_e. Zgodnie z założeniami przedstawionymi w punkcie 3.1 należy wyznaczyć ścieżkę p_{se} w grafie G, dla której wagi $T(p_{se})$ i $C(p_{se})$ przyjmują wartość minimalną.

$$T(p_{se}) = \sum_{i=1}^{m} t(v_{i-1}, v_i), \; C(p_{se}) = \sum_{i=1}^{m} c(v_{i-1}, v_i) \tag{4}$$

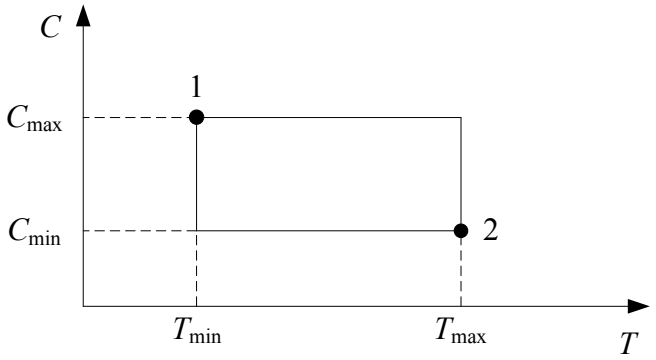
Problem wyznaczania połączeń komunikacyjnych jest przykładem zadania optymalizacji wielokryterialnej. Wagi T i C ścieżek reprezentujących połączenia komunikacyjne są funkcjami kryterialnymi minimalizowanymi. Rozwiązanie problemu wyznaczania połączeń komunikacyjnych sprowadza się do rozwiązania dwukryterialnego problemu wyznaczania najkrótszej ścieżki w grafie (ang. *Bicriterion Shortest Path*) o zmiennych wagach. Rozwiązaniem problemu jest zbiór ścieżek, reprezentujących połączenia komunikacyjne, stanowiących zbiór rozwiązań niezdominowanych. Dwukryterialny problem wyznaczania najkrótszej ścieżki w grafie o stałych wagach, poprzez transformację do 0–1 problemu plecakowego, należy do grupy problemów NP–zupełnych [18, 19]. W przypadku pesymistycznym liczba rozwiązań rośnie wykładniczo wraz ze wzrostem liczby wierzchołków grafu [37].

W pracy [49] dokonano dokładnej analizy własności ścieżek należących do zbioru rozwiązań niezdominowanych oraz zamieszczono przykładowe ścieżki. Wykazano, że zbiór rozwiązań niezdominowanych może zawierać wiele ścieżek o takich samych wagach T i C. Ścieżki te różnią się wierzchołkami i łukami tworzącymi ścieżkę lub godzinami odjazdów.

Podczas wyznaczania rozwiązań nie ma konieczności przeszukiwania całego obszaru przestrzeni rozwiązań. W pracy [49] wykazano, że połączenia o minimalnym czasie i koszcie

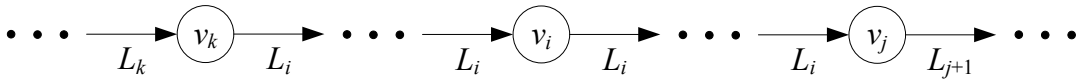
[1] W dalszej części pracy używając zwrotu wierzchołek, w odniesieniu do grafu reprezentującego sieć komunikacyjną, będziemy mieć na myśli przystanek w sieci komunikacyjnej, który on reprezentuje.

przejazdu określają obszar przestrzeni rozwiązań, w którym znajdują się potencjalne rozwiązania niezdominowane. Rozwiązanie 1 (rys. 1), będące połączeniem o minimalnym czasie przejazdu $T_{\min}$, określa maksymalny koszt przejazdu $C_{\max}$ rozwiązania należącego do zbioru rozwiązań niezdominowanych, a połączenie o minimalnym koszcie przejazdu $C_{\min}$ (rozwiązanie 2 na rys. 1) określa maksymalny czas przejazdu $T_{\max}$ takiego rozwiązania. Na podstawie wartości $T_{\max}$ i godziny rozpoczęcia podróży T_s można określić najpóźniejszą godzinę przybycia do v_e w rozwiązaniu niezdominowanym.

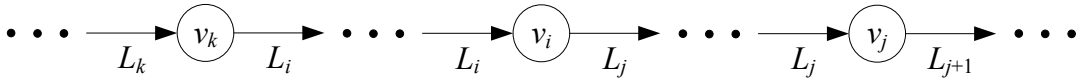


Rys. 1. Obszar rozwiązań niezdominowanych
Fig. 1. A space of non–dominated solutions

a)



b)



Rys. 2. Fragment ścieżki reprezentującej połączenie: a) bez przesiadki w wierzchołku v_i, b) z przesiadką w wierzchołku v_i
Fig. 2. A part of path representing a route: a) without a change at v_i, b) with a change at v_i

Szczególną własność ma koszt przejazdu, co dokładnie przeanalizowano w pracy [49]. Niech będzie dana ścieżka p_{se}, która powstaje przez złączenie ścieżek p_{si} i p_{ie}, odpowiednio o kosztach przejazdu $C(p_{si})$ i $C(p_{ie})$. Koszt przejazdu $C(p_{se})$ utworzonej ścieżki zależy od tego, czy w wierzchołku v_i jest dokonywana przesiadka (rys. 2b), czy też przejazd odbywa się bez przesiadki w tym wierzchołku (rys. 2a). Jeżeli przejazd odbywa się z przesiadką, to koszt przejazdu jest równy: $C(p_{se}) = C(p_{si}) + C(p_{ie})$. W przypadku przejazdu bez przesiadki wynosi: $C(p_{se}) = C(p_{si}) + C(p_{ie}) - \Delta c$, gdzie Δc jest przyrostem kosztu, o jaki jest powiększany koszt przejazdu $C(p_{se})$ ze względu na to, że w wierzchołku v_i nie jest dokonywana przesiadka. Wartość Δc jest równa: $\Delta c = C(p_{ki}) + C(p_{ij}) - C(p_{kj})$, gdzie $C(p_{ki})$, $C(p_{ij})$, $C(p_{kj})$ są kosztami

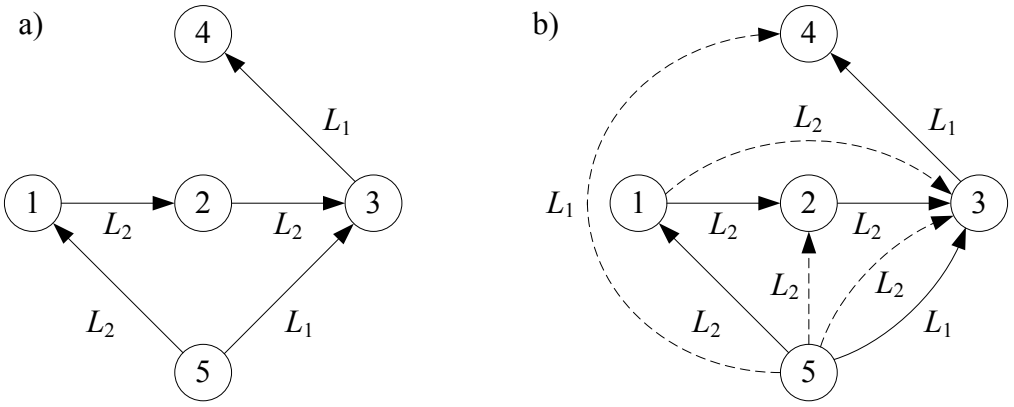
przejazdu linią L_i. Wartości przyrostu kosztu Δc zostały przedstawione w tabeli 1[2] i są one wyznaczane na podstawie liczby stref, które zostały przekroczone przy dojeździe linią L_i do wierzchołka v_i z wierzchołka v_k oraz z wierzchołka v_i do wierzchołka v_j. Wierzchołki v_k i v_j są wierzchołkami, w których jest dokonywana przesiadka.

Tabela 1

Wartości przyrostu kosztu Δc uwzględnianego przy wyznaczaniu kosztu dojazdu z wierzchołka v_k do wierzchołka v_j

		Liczba przekroczonych stref przy dojeździe z v_i do v_j		
		0	1	2 lub więcej
Liczba przekroczonych stref przy dojeździe z v_k do v_i	0	c_1	c_1	c_1
	1	c_1	$2c_2 - c_3$	c_2
	2 lub więcej	c_1	c_2	c_3

Do wyznaczenia połączeń o minimalnym czasie i koszcie przejazdu może zostać użyty np. algorytm Dijkstry [8, 15, 20, 23, 34, 39]. W tym celu należy jednak zmodyfikować reprezentację sieci komunikacyjnej, gdyż, jak wykazano w pracach [46, 49], nie jest możliwe wyznaczenie w grafie G połączenia o minimalnym koszcie przejazdu z użyciem algorytmu Dijkstry. Dla każdej linii komunikacyjnej w grafie G należy umieścić dodatkowe łuki dla każdej pary kolejnych wierzchołków należących do trasy danej linii, tworząc w ten sposób graf $G_d=(V, E_d)$, $E \subseteq E_d$. Niech trasa linii komunikacyjnej L_i ma postać: $\langle v_0, v_1, \ldots, v_k \rangle$. W grafie G zostaną umieszczone łuki (v_i, v_j), gdzie: $i = 0, \ldots, k–2$, $j = i+2, \ldots, k$. Dzięki dodaniu dodatkowych łuków istnieje możliwość rozpatrzenia dojazdu linią L_i z wierzchołka v_i do wszystkich wierzchołków v_j będących następnikami wierzchołka v_i w trasie linii L_i.



Rys. 3. Reprezentacja sieci komunikacyjnej: a) z użyciem grafu G, b) z użyciem grafu G_d
Fig. 3. A representation of the communication network: a) using the G graph, b) using the G_d graph

Na rys. 3 przedstawiono reprezentację przykładowej sieci komunikacyjnej za pomocą grafów G i G_d. Przy każdym łuku $e_i = (v_j, v_k)$ zaznaczono linię komunikacyjną, pojazdem której odbywa się przejazd z v_j do v_k. W sieci kursują pojazdy linii komunikacyjnych L_1 i L_2.

[2] Przedstawione wartości dotyczą linii zwykłej, w przypadku linii pospiesznej ulegają one podwojeniu.

Trasa przejazdu linii L_1 składa się z sekwencji wierzchołków: ⟨*5*, 3, 4⟩, a linii L_2 zawiera wierzchołki: ⟨5, 1, 2, 3⟩. W celu utworzenia grafu G_d (rys. 3b) dla linii L_1 należy umieścić w grafie G (rys. 3a) łuk (5, 4), a dla linii L_2 łuki: (5, 2), (5, 3), (1, 3). Na rysunku łuki te zaznaczono w sposób przerywany.

3.3. Algorytm korygowania etykiet z usuwaniem rozwiązań częściowych

Algorytm, w którym podczas wyznaczania rozwiązań nie są pamiętane rozwiązania częściowe, przedstawiono w pracy [46]. Podczas wyznaczania rozwiązań pamiętane jest tylko jedno rozwiązanie P, które aktualnie jest poddawane analizie. Rozwiązanie P zawiera rekordy postaci (v_k, t_{sk}, c_{sk}, L_k, T_{k-1}), indeksowane od 0 (tj. $k = 0, 1, \ldots$), gdzie:

- v_k – wierzchołek znajdujący się na k–tej pozycji w ścieżce,
- t_{sk}, c_{sk} – czas i koszt dojazdu z wierzchołka początkowego v_s do wierzchołka v_k,
- L_k – numer linii komunikacyjnej, pojazdem której następuje przejazd z v_{k-1} do v_k,
- T_{k-1} – godzinę wyjazdu z wierzchołka v_{k-1} w kierunku wierzchołka v_k.

Dodawanie kolejnych rekordów do rozwiązania P, co odpowiada rozszerzeniu ścieżki o kolejny wierzchołek, odbywa się przez odwiedzanie wierzchołków grafu G, reprezentującego sieć komunikacyjną, za pomocą przeszukiwania grafu w głąb połączonego z metodą powrotów [1, 2, 8, 20, 23, 34]. Do rozwiązania są dodawane nowe rekordy aż do momentu uzyskania rozwiązania końcowego, tj. dodania rekordu zawierającego wierzchołek v_e. Rozwiązanie częściowe P reprezentuje więc połączenie komunikacyjne z wierzchołka v_s do wierzchołka v_k, będącego ostatnim wierzchołkiem w rozwiązaniu P. Ponieważ graf zawiera łuki równoległe, więc przy wyborze wierzchołka v_{k+1}, o który zostanie rozszerzone rozwiązanie, są analizowane wszystkie łuki wychodzące z wierzchołka v_k.

W przypadku kiedy nie ma możliwości wyznaczenia wierzchołka v_{k+1}, o który zostanie rozszerzone rozwiązanie P, jest dokonywany powrót, polegający na usunięciu z rozwiązania ostatniego rekordu, tj. rekordu zawierającego wierzchołek v_k. Dokonanie powrotu z wierzchołka v_s oznacza zakończenie wyznaczania rozwiązań. Przez usunięcie wierzchołka v_k z rozwiązania P jest usuwane rozwiązanie częściowe, będące połączeniem z wierzchołka v_s do wierzchołka v_k, które nie zostało zapamiętane. Ponadto, po wyznaczeniu rozwiązania końcowego nie ma możliwości określenia, czy wynikowy zbiór rozwiązań będzie zawierał to rozwiązanie, stąd algorytm ten jest zaliczany do grupy algorytmów korygowania etykiet z usuwaniem rozwiązań częściowych.

Dokonując w opisany sposób rozszerzenia rozwiązania P o kolejne wierzchołki, zostaną wyznaczone wszystkie istniejące ścieżki z wierzchołka v_s do wierzchołka v_e, a oznacza to przeszukanie całej przestrzeni rozwiązań, co nie jest konieczne. W trakcie rozszerzania rozwiązania P o kolejne wierzchołki można bowiem stwierdzić, że uzyskane rozwiązanie

końcowe będzie rozwiązaniem zdominowanym. Można wykazać [49], że rozwiązanie P powinno być analizowane wyłącznie w sytuacji, kiedy są spełnione zależności (5) i (6), gdzie $T^e_{\min}[v_k]$ i $C^e_{\min}[v_k]$ są odpowiednio minimalnym czasem i kosztem dojazdu z wierzchołka v_k do v_e, a $\Delta c_{\max}$ jest maksymalnym przyrostem kosztu określonym tabelą 1 (jeżeli v_k jest wierzchołkiem końcowym v_e, wówczas $\Delta c_{\max} = 0$). W przeciwnym przypadku można dokonać powrotu, ponieważ rozwiązanie końcowe uzyskane z rozwiązania P będzie rozwiązaniem zdominowanym.

$$t_{sk} + T^e_{\min}[v_k] \le T_{\max} \tag{5}$$

$$c_{sk} + C^e_{\min}[v_k] - \Delta c_{\max} \le C_{\max} \tag{6}$$

Z rozwiązaniem P jest skojarzona historia H zawierająca informację o wszystkich analizowanych łukach wychodzących z każdego wierzchołka należącego do rozwiązania. Pamiętanie historii H zapewnia zakończenie operacji wyszukiwania rozwiązań oraz zapobiega wielokrotnemu wyznaczaniu tego samego rozwiązania. Podczas wyboru wierzchołka v_{i+1}, o który zostanie rozszerzone rozwiązanie P, łuk wychodzący z wierzchołka v_i jest analizowany wyłącznie wtedy, gdy nie był on analizowany dla rozwiązania P.

```
    Wejście: G = (V, E) - graf skierowany reprezentujący sieć komunikacyjną
             v_s        - wierzchołek reprezentujący przystanek początkowy
             v_e        - wierzchołek reprezentujący przystanek końcowy
             T_s        - godzina rozpoczęcia podróży na przystanku początkowym
    Wyjście: R          - zbiór rozwiązań, tj. ścieżek reprezentujących
                          połączenia komunikacyjne
 1: procedure DFSB(G, v_s, v_e, T_s, R)
 2: begin
 3:   wyznacz wartości T_max i C_max;
 4:   for all v ∈ V do
 5:     wyznacz wartości T^e_min[v] i C^e_min[v] dla wierzchołka v;
 6:   end for
 7:   P := (v_s, 0, 0, 0, T_s);      // wstawienie wierzchołka v_s do rozwiązania
 8:   k := 0; H := ∅; R := ∅; koniec := false;
 9:   while not koniec do
10:     k := k+1; v_k := wierzchołek, o który zostanie rozszerzone rozwiązanie P;
11:     if ∃v_k then
12:       P := P ⊕ v_k; aktualizacja historii H;
13:       if P jest rozwiązaniem akceptowalnym then
14:         if v_k = v_e then          // znaleziono rozwiązanie końcowe
15:           dodanie rozwiązania P do zbioru rozwiązań R;
16:           dokonanie powrotu, tj. usunięcie z P wierzchołka v_k; k := k-1;
17:         end if
18:       else   // rozwiązanie końcowe uzyskane z rozw. P będzie zdominowane
19:         dokonanie powrotu, tj. usunięcie z P wierzchołka v_k; k := k-1;
20:       end if
21:     else     // nie istnieje v_k, o który zostanie rozszerzone rozw. P
22:       k := k-1;
23:       if v_k = v_s then
24:         koniec := true;       // zakończenie wyznaczania rozwiązań
25:       else
26:         dokonanie powrotu, tj. usunięcie z P wierzchołka v_k; k := k-1;
27:       end if
28:     end if
29:   end while
30:   return R;
31: end
```

Treść algorytmu została przedstawiona w postaci procedury *DFSB*. W części inicjalizacyjnej algorytmu, na podstawie połączeń o minimalnym czasie i koszcie przejazdu, są wyznaczane wartości T_{max} i C_{max} (wiersz 3), a dla każdego wierzchołka v są wyznaczane $T^e_{min}[v]$ i $C^e_{min}[v]$ (wiersz 5), które można wyznaczyć, podobnie jak wartości T_{max} i C_{max}, np. za pomocą algorytmu Dijkstry [49]. Rozwiązanie P jest inicjowane rekordem zawierającym wierzchołek v_s (wiersz 7), ponadto jest zerowany indeks k określający ostatni rekord w rozwiązaniu P, a zbiór rozwiązań R i historia H są inicjowane jako puste (wiersz 8).

Rozwiązania są wyznaczane w iteracji **while** (wiersze 9–29), której zakończenie jest sterowane wartością znacznika *koniec*. W każdej iteracji następuje próba wyznaczenia wierzchołka v_k, o który zostanie rozszerzone rozwiązanie P (wiersz 10). Jeżeli taki wierzchołek istnieje, to rozwiązanie P jest rozszerzane o ten wierzchołek oraz jest aktualizowana historia H (wiersz 12). Rozszerzenie rozwiązania P jest określone operacją $P \oplus v_k$, co należy rozumieć przez dodanie na końcu rozwiązania P rekordu zawierającego wierzchołek v_k. Jeżeli uzyskane w ten sposób rozwiązanie P jest rozwiązaniem akceptowalnym (ocena jest dokonywana na podstawie (5) i (6)), a ponadto jest rozwiązaniem końcowym, to jest ono dodawane do zbioru rozwiązań R (wiersz 15) i jest dokonywany powrót (wiersz 16). Powrót jest dokonywany także w przypadku, kiedy utworzone rozwiązanie P nie jest rozwiązaniem akceptowalnym (wiersz 19).

Ostatnim przypadkiem, w którym jest dokonywany powrót, jest sytuacja, kiedy nie ma możliwości wyznaczenia wierzchołka v_k, o który zostanie rozszerzone rozwiązanie P. W tym przypadku powrót (wiersz 26) jest dokonywany wyłącznie w sytuacji, kiedy rozwiązanie P zawiera więcej niż jeden rekord, tj. rozwiązanie reprezentuje ścieżkę zawierającą więcej niż jeden wierzchołek. Jeżeli rozwiązanie P zawiera tylko jeden rekord, oznacza to zakończenie wyznaczania rozwiązań przez nadanie znacznikowi *koniec* wartości **true** (wiersz 24). Jest to sytuacja, w której jest dokonywany powrót z wierzchołka początkowego v_s.

3.4. Algorytm korygowania etykiet z pamiętaniem rozwiązań częściowych

Algorytm wyznaczania połączeń komunikacyjnych, w którym są pamiętane rozwiązania częściowe, został przedstawiony w pracy [47], a opracowany na podstawie algorytmów przedstawionych w pracach [3, 37]. W pracach tych zostały przedstawione algorytmy rozwiązywania dwukryterialnego problemu wyznaczania najkrótszej ścieżki w grafie o stałych nieujemnych wagach. Bezpośrednie zastosowanie wymienionych algorytmów w omawianym problemie nie jest możliwe, gdyż w grafie o zmiennych wagach obowiązują inne zależności niż w grafie o stałych wagach, co zostało dokładnie opisane w pracy [49]. Ponadto w wymienionych algorytmach nie są wyznaczane same ścieżki, tylko ich wagi.

Dla każdego wierzchołka v_j grafu jest pamiętana lista rozwiązań $D[v_j]$, gdzie każde z nich jest rozwiązaniem częściowym opisującym ścieżkę z wierzchołka v_s do wierzchołka v_j[3]. Pojedyncze rozwiązanie jest rekordem postaci $(t_{sj}, c_{sj}, v_i, L_j, T_i, P_j)$, gdzie:

- t_{sj}, c_{sj} – czas i koszt dojazdu z wierzchołka początkowego v_s do wierzchołka v_j,
- v_i – wierzchołek, który poprzedza w trasie wierzchołek v_j,
- L_j – numer linii komunikacyjnej, pojazdem której następuje przejazd z v_i do v_j,
- T_i – godzina wyjazdu z wierzchołka v_i w kierunku wierzchołka v_j,
- P_j – wskaźnik do rozwiązania dla wierzchołka v_i poprzedzającego w ścieżce v_j.

Na podstawie wskaźnika P_j istnieje możliwość odtworzenia trasy dojazdu z wierzchołka początkowego v_s do v_j. Po wyznaczeniu rozwiązania dla wierzchołka v_e nie ma możliwości określenia, czy wynikowy zbiór rozwiązań będzie zawierał to rozwiązanie. Ponieważ dla każdego wierzchołka jest pamiętana lista rozwiązań częściowych, a nie tylko jedno aktualnie analizowane rozwiązanie częściowe, stąd algorytm ten należy do grupy algorytmów korygowania etykiet z pamiętaniem rozwiązań częściowych.

Wyznaczanie rozwiązań odbywa się przez odwiedzanie wierzchołków grafu G za pomocą przeszukiwania grafu wszerz [8, 20, 23, 34]. Z każdego rozwiązania znajdującego się w liście $D[v_j]$ jest podejmowana próba utworzenia nowych rozwiązań dla wszystkich wierzchołków v_k sąsiadujących z v_j, co sprawdza się do rozszerzenia ścieżki z wierzchołka v_s do v_j o wierzchołek v_k i wymaga analizy wszystkich łuków wychodzących z wierzchołka v_j. Nowo utworzone rozwiązanie reprezentuje więc ścieżkę z wierzchołka v_s do v_k, gdzie v_j jest wierzchołkiem poprzedzającym w ścieżce wierzchołek v_k. Niech będzie dane rozwiązanie (7) dla wierzchołka v_j. Rozszerzenie tego rozwiązania o wierzchołek v_k, do którego prowadzi łuk $e_i = (v_j, v_k)$ o wagach: $t(e_i)$, $c(e_i)$ i $l(e_i)$, będzie określone operacją (8), gdzie P_k' zawiera wskaźnik do rozwiązania (7).

$$(t_{sj}', c_{sj}', v_i', L_j', T_i', P_j') \tag{7}$$

$$(t_{sj}', c_{sj}', v_i', L_j', T_i', P_j') \oplus (v_j, v_k) = (t_{sj}' + t(e_i), c_{sj}' + t(e_i), v_j, l(e_i), T_j', P_k') \tag{8}$$

W algorytmach przedstawionych w pracach [3, 37] dla każdego wierzchołka v_j pamiętany jest zbiór rozwiązań niezdominowanych, gdyż w przypadku stałych nieujemnych wag nie ma możliwości uzyskania rozwiązania niezdominowanego dla wierzchołka v_j przez rozszerzenie rozwiązania zdominowanego dla wierzchołka v_i. W przypadku wyznaczania połączeń komunikacyjnych, gdzie wagi są zmienne, pamiętanie dla każdego wierzchołka zbioru rozwiązań niezdominowanych jest niewystarczające i nie zapewnia wyznaczenia wszystkich rozwiązań niezdominowanych dla wierzchołka końcowego v_e. W pracach [47, 49] wykazano, że przez rozszerzenie zdominowanego rozwiązania dla wierzchołka v_j można uzyskać niezdominowane rozwiązanie dla wierzchołka v_i.

[3] Dla wierzchołka v_e rozwiązania te są rozwiązaniami końcowymi opisującymi szukane połączenia.

```
    Wejście: G = (V, E) - graf skierowany reprezentujący sieć komunikacyjną
             v_s        - wierzchołek reprezentujący przystanek początkowy
             v_e        - wierzchołek reprezentujący przystanek końcowy
             T_s        - godzina rozpoczęcia podróży na przystanku początkowym
    Wyjście: R          - zbiór rozwiązań, tj. ścieżek reprezentujących
                          połączenia komunikacyjne
 1: procedure BFS(G, v_s, v_e, T_s, R)
 2: begin
 3:   wyznacz wartości T_max i C_max;
 4:   for all v ∈ V do
 5:     wyznacz wartości T^e_min[v] i C^e_min[v] dla wierzchołka v;   D[v] := ∅;
 6:   end for
 7:   D[v_s] := (0, T_s, 0, 0, 0, nil);    Q ⇐ v_s;
 8:   while Q ≠ ∅ do                  // dopóki kolejka nie jest pusta
 9:     v_j ⇐ Q;     // pobranie i usuniecie z kolejki pierwszego wierzchołka
10:     for all łuki (v_j, v_k) ∈ E wychodzące z wierzchołka v_j do
11:       for all nieanalizowane rozwiązanie d_j ∈ D[v_j] do
12:         d_k := d_j ⊕ (v_j, v_k);   // utworzenie rozwiązania dla wierzchołka v_k
13:         if rozwiązanie d_k jest rozwiązaniem akceptowalnym then
14:           D[v_k] ⇐ d_k;   Q ⇐ v_k;
15:         end if
16:       end for
17:     end for
18:   end while
19:   R := utworzone pełne ścieżki na podstawie listy rozwiązań D[v_e];
20:   return R;
21: end
```

Treść algorytmu wyznaczania połączeń komunikacyjnych, w którym są pamiętane rozwiązania częściowe, została przedstawiona w postaci procedury *BFS*. W części inicjalizacyjnej są wyznaczane wartości $T_{\max}$ i $C_{\max}$ (wiersz 3), a dla każdego wierzchołka v są wyznaczane wartości $T^e_{\min}[v]$ i $C^e_{\min}[v]$, a lista rozwiązań $D[v]$ jest inicjowana jako pusta (wiersz 5). Wierzchołek v_s jest wstawiany do kolejki Q i jest tworzone dla niego rozwiązanie startowe, od którego rozpocznie się wyznaczanie wszystkich rozwiązań (wiersz 7).

Wyznaczanie rozwiązań odbywa się w iteracji **while** (wiersze 8–18), która wykonywana jest do chwili, w której kolejka Q nie będzie zawierała żadnego wierzchołka. W każdej iteracji jest pobierany i usuwany z kolejki Q pierwszy wierzchołek v_j (wiersz 9). Następnie są analizowane wszystkie łuki (v_j, v_k) wychodzące z tego wierzchołka (wiersze 10–17) oraz rozwiązania d_j znajdujące się w liście rozwiązań $D[v_j]$, które nie były jeszcze analizowane (wiersze 11–16)[4]. Dla każdego rozwiązania d_j, zgodnie z operatorem określonym zależnością (8), następuje próba utworzenia nowego rozwiązania d_k dla wierzchołka v_k (wiersz 12).

Jeżeli utworzone rozwiązanie d_k jest rozwiązaniem akceptowalnym, tj. są spełnione zależności (5) i (6), to jest dodawane do listy rozwiązań $D[v_k]$ dla wierzchołka v_k, a wierzchołek ten jest wstawiany do kolejki Q (wiersz 14)[5]. Jeżeli wierzchołek v_k jest wierzchołkiem

[4] Rozwiązaniem dla wierzchołka v_j, które nie było jeszcze analizowane, jest rozwiązanie, dla którego nie była podejmowana próba utworzenia nowych rozwiązań dla wierzchołków sąsiadujących.

[5] Jeżeli w chwili wstawiania wierzchołka v_k do kolejki Q kolejka zawiera ten wierzchołek, to nie jest on ponownie wstawiany.

końcowym v_e, to po dodaniu rozwiązania do listy $D[v_k]$ są z niej usuwane rozwiązania zdominowane.

Rozwiązania wyznaczone w iteracji **while** (wiersze 8–18) nie zawierają pełnych ścieżek, stąd po ich wyznaczeniu na podstawie listy rozwiązań znajdujących się w liście $D[v_e]$ są wyznaczane pełne ścieżki reprezentujące trasy przejazdu (wiersz 19). Wyznaczone ścieżki są umieszczane w zbiorze R, który jest zwracany jako wynik wykonania algorytmu.

3.5. Algorytm *K*-tej najkrótszej ścieżki

Połączenia komunikacyjne są wyznaczane z użyciem algorytmu wyznaczania K–tej najkrótszej ścieżki w algorytmie *MKSP*, przedstawionym w pracy [50]. Problem wyznaczania K ($K > 1$) najkrótszych ścieżek z wierzchołka v_s do v_e polega na wyznaczeniu zbioru ścieżek $P_K = \{p_1, ..., p_K\}$ spełniającego następujące warunki:

- $\forall i \in \{1, ..., K–1\}\ W(p_i) \le W(p_{i+1})$,
- $\forall p \in P \setminus P_K\ \ W(p_K) \le W(p)$,
- $\forall i \in \{1, ..., K–1\}$ ścieżka p_i jest wyznaczana przed wyznaczeniem ścieżki p_{i+1},

gdzie W jest funkcją wagową.

Wyznaczenie K najkrótszych ścieżek z v_s do v_e jest podobne do konstruowania drzewa ścieżek, którego korzeniem jest v_s, a liśćmi są wierzchołki v_e. Można wykazać, że dla danych dwóch ścieżek z v_s do v_e, określonych zależnościami (9) i (10) zachodzi zależność (11).

$$p_i = <v_0^{\,i} = v_s, (v_0^{\,i}, v_1^{\,i}), v_1^{\,i}, (v_1^{\,i}, v_2^{\,i}), ..., (v_{n-1}^{\,i}, v_n^{\,i}), v_n^{\,i} = v_e> \in P_K \tag{9}$$

$$p_j = <v_0^{\,j} = v_s, (v_0^{\,j}, v_1^{\,j}), v_1^{\,j}, (v_1^{\,j}, v_2^{\,j}), ..., (v_{m-1}^{\,j}, v_m^{\,j}), v_m^{\,j} = v_e> \in P_K \tag{10}$$

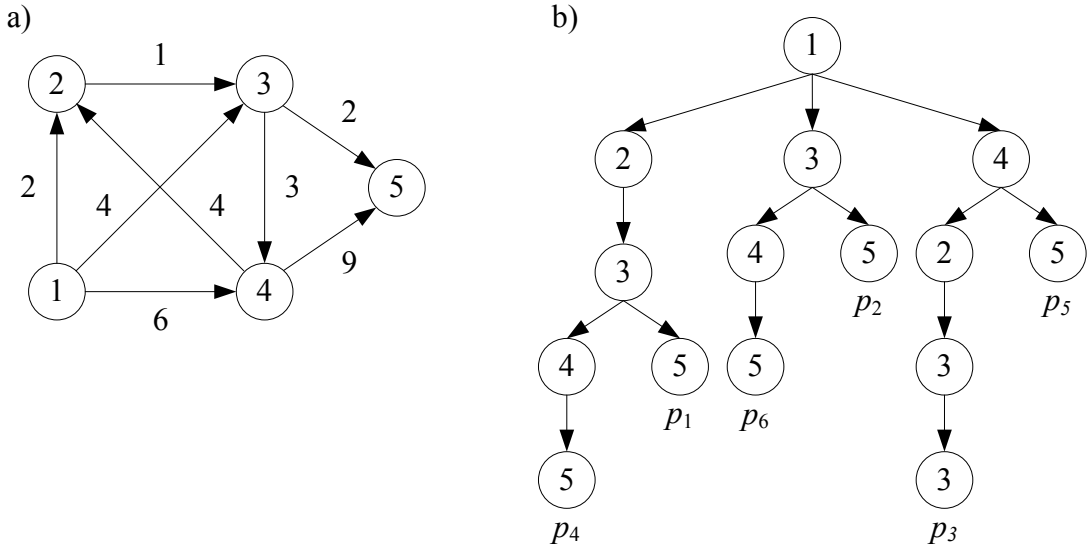
$$\exists\ 0 \le y < min\{n, m\}\ \forall\ x = \{0, ..., y\}\ v_x^{\,i} = v_x^{\,j}. \tag{11}$$

Wierzchołki $v_y^{\,i}$ oraz $v_y^{\,j}$ są ostatnimi wierzchołkami we wspólnej podścieżce ścieżek określonych zależnościami (9) i (10). Taki wierzchołek nosi nazwę wierzchołka odchylającego (ang. *deviation node*) i jest oznaczany przez $d(p_i)$. Przez $d_1(p_i)$ zostanie oznaczony wierzchołek będący następnikiem w ścieżce wierzchołka $d(p_i)$.

Na rys. 4a został przedstawiony przykładowy skierowany graf ważony, a na rys. 4b drzewo wszystkich ścieżek z wierzchołka 1 do wierzchołka 5, o następujących wagach: $W(p_1) = 5$, $W(p_2) = 6$, $W(p_3) = 13$, $W(p_4) = 15$, $W(p_5) = 15$, $W(p_6) = 16$. Przykładowo, dla ścieżek p_1 i p_4 wierzchołkiem odchylającym jest wierzchołek $d(p_1) = d(p_4) = 3$. Wyznaczając np. $K = 3$ najkrótszych ścieżek zostanie uzyskany zbiór $P_K = \{p_1, p_2, p_3\}$.

Aby skonstruować drzewo ścieżek konieczne jest wyznaczenie dowolnej ścieżki p_i z v_s do v_e, na podstawie której są tworzone kolejne ścieżki. Przed wyznaczaniem nowych ścieżek są usuwane z grafu wszystkie wierzchołki poprzedzające wierzchołek $d(p_i)$ w ścieżce p_i, a także wszystkie łuki $(d(p_i), v_i)$ należące do wcześniej analizowanych ścieżek, tj., p_1, …, p_{i-1}.

Operacja ta zabezpiecza przed wielokrotnym wyznaczeniem tych samych ścieżek i zapewnia skończoność algorytmu. Niech ścieżka p_i ma postać określoną zależnością (9). Z grafu zostaną usunięte kolejno łuki $(d(p_i), v_j^i)$, (v_j^i, v_{j+1}^i), ..., (v_{n-1}^i, v_n^i), a po usunięciu każdego z nich zostanie wyznaczona ścieżka z wierzchołków $d(p_i)$, v_j^i, ..., v_{n-1}^i do wierzchołka v_e. Z każdego wierzchołka jest wyznaczana tylko jedna ścieżka, a po jej wyznaczeniu wierzchołek, z którego była wyznaczana, jest usuwany z grafu. W taki sposób na podstawie ścieżki p_i powstaje $n–j+1$ ścieżek. Po ich wyznaczeniu w grafie są umieszczane wszystkie wierzchołki i łuki, jakie zostały usunięte podczas analizy ścieżki p_i, a na podstawie nowo utworzonych ścieżek są tworzone kolejne ścieżki. Opisana metoda konstruowania drzewa ścieżek została przedstawiona w pracy [28], a jej analiza wraz z przykładami w pracy [50].



Rys. 4. Przykładowe grafy: a) skierowany ważony, b) drzewo ścieżek z wierzchołka 1 do wierzchołka 5

Fig. 4. A sample graphs: a) directed weighted, b) a paths tree from node 1 do node 5

Można wykazać, że ścieżki p_i oraz p_j rozpoczynają się od tej samej podścieżki [50], gdzie p_j jest ścieżką utworzoną na podstawie ścieżki p_i. Przed utworzeniem nowych ścieżek jest konieczne usunięcie niektórych wierzchołków i łuków, dlatego podczas tworzenia nowych ścieżek na podstawie ścieżki p_j zachodzi konieczność usunięcia z grafu tych samych wierzchołków i łuków, jakie były usunięte podczas tworzenia ścieżek na podstawie ścieżki p_i. Stąd wadą przedstawionej metody konstruowania drzewa ścieżek jest wielokrotne usuwanie z grafu tych samych wierzchołków i łuków.

Aby uniknąć wielokrotnego usuwania tych samych wierzchołków i łuków, należy zmienić kolejność wyznaczania ścieżek, tj. z każdego analizowanego wierzchołka ścieżki p_i nie wyznaczać tylko jednej ścieżki, ale jeśli to jest możliwe, to wyznaczyć kolejne ścieżki. Niech z analizowanego wierzchołka $d(p_i)$ istnieje możliwość utworzenia m nowych ścieżek. W takim przypadku zostanie utworzonych m nowych ścieżek, a wierzchołki poprzedzające

w ścieżce p_i wierzchołek $d(p_i)$ i wszystkie łuki $(d(p_i), v_i)$ należące do ścieżek $p_1, \ldots, p_{i-1}$ zostaną usunięte z grafu tylko jeden raz. W opisanej wcześniej metodzie wierzchołki i łuki te zostałyby usunięte z grafu m razy. Ponadto, każda nowo utworzona ścieżka p_j nie musi być analizowana od wierzchołka odchylającego $d(p_j)$, tylko od kolejnego wierzchołka, tj. $d_1(p_j)$. Wszystkie ścieżki, jakie można uzyskać analizując wierzchołek $d(p_j)$, zostały już wyznaczone w trakcie analizy ścieżki p_i, na podstawie której została utworzona ścieżka p_j.

```
    Wejście: G = (V, E) - graf skierowany reprezentujący sieć komunikacyjną
             v_s        - wierzchołek reprezentujący przystanek początkowy
             v_e        - wierzchołek reprezentujący przystanek końcowy
             T_s        - godzina rozpoczęcia podróży na przystanku początkowym
    Wyjście: R          - zbiór rozwiązań, tj. ścieżek reprezentujących
                          połączenia komunikacyjne
 1: procedure MKSP(G, v_s, v_e, T_s, R)
 2: begin
 3:   p := ścieżka z wierzchołka v_s do v_e o minimalnym czasie przejazdu;
 4:   wyznacz wartości T_max i C_max;
 5:   d_1(p) := v_s;   R := {p};  k := 1;
 6:   while ∃p_k ∈ R do
 7:     p_k := ścieżka <v_0^k=v_s, ..., v_j^k=v_e> o indeksie k należąca do zbioru R;
 8:     usuń z grafu G wierzchołki znajdujące się w ścieżce sub_pk(v_s, d(p_k));
 9:     for all v_i^k ∈ sub_pk(d_1(p_k), v_{j-1}^k) do
10:       usuń z grafu G łuk (v_i^k, v_{i+1}^k);
11:       repeat
12:         próba wyznaczenia z v_i^k do v_e ścieżki q = <v_0^q = v_i^k, v_1^q, ..., v_e>
            o minimalnym czasie;
13:         if ∃q then
14:           p := sub_pk(v_s, v_i^k) ⊕ q;              // nowa ścieżka z v_s do v_e
15:           if p jest rozwiązaniem akceptowalnym then
16:             d_1(p) := v_1^q;  R := R ∪ {p};         // wstaw p do zbioru rozwiązań
17:           end if
18:           usuń z grafu G łuk (v_0^q = v_i^k, v_1^q);
19:         end if
20:       until (not ∃q) cor (T(p) > T_max);
21:       usuń z grafu wierzchołek v_i^k;
22:     end for
23:     umieść w grafie usunięte łuki i wierzchołki;
24:     k := k + 1;    // następna ścieżka do przeanalizowania ze zbioru R
25:   end while
26:   usuń rozwiązania zdominowane ze zbioru R;
27:   return R;
28: end
```

Treść algorytmu wyznaczania połączeń komunikacyjnych z użyciem algorytmu wyznaczania K–tej najkrótszej ścieżki i konstruowania drzewa ścieżek została przedstawiona w postaci procedury *MKSP*. W części inicjalizacyjnej jest wyznaczana z wierzchołka v_s do v_e ścieżka p o minimalnym czasie przejazdu (wiersz 3) oraz wartości $T_{\max}$ i $C_{\max}$ (wiersz 4). Ścieżka p o minimalnym czasie jest ścieżką startową, na podstawie której zostaną wyznaczone pozostałe ścieżki, zostaje ona umieszczona w zbiorze rozwiązań R, a wierzchołkiem $d_1(p)$, od którego rozpocznie się jej analiza, jest wierzchołek v_s (wiersz 5). Ścieżki są numerowane w zbiorze R za pomocą indeksu k.

Ścieżki są wyznaczane w iteracji **while** (wiersze 6–25), która jest wykonywana dopóki w zbiorze R istnieje nieanalizowana ścieżka p_k. Do wyznaczania ścieżek użyta jest druga

z opisanych metod konstruowania drzewa ścieżek. W każdej iteracji pobierana jest ze zbioru rozwiązań R ścieżka p_k, na podstawie której będą tworzone nowe ścieżki (wiersz 7). Usuwane są także z grafu G wszystkie wierzchołki poprzedzające wierzchołek $d_1(p_j)$ w ścieżce p_k (wiersz 8), tj. wierzchołki stanowiące podścieżkę z wierzchołka v_s do $d(p_k)$ ścieżki p_k. Do określenia podścieżki użyto zapisu $sub_{pk}(v_s, d(p_k))$, co określa podścieżkę ścieżki p_k zawierającą sekwencję wierzchołków i łuków od wierzchołka v_s do wierzchołka $d(p_k)$.

Dla ścieżki p_k są analizowane w iteracji **for** wszystkie wierzchołki należące do podścieżki $sub_{pk}(d_1(p_k), v_{j-1}^k)$ (wiersze 9–22). Dla każdego analizowanego wierzchołka v_i^k jest usuwany z grafu G łuk wychodzący z niego, należący do ścieżki p_k (wiersz 10), oraz jest podejmowana próba wyznaczenia nowych ścieżek z tego wierzchołka do wierzchołka v_e (wiersze 11–20). Każdorazowo jest wyznaczana z wierzchołka v_i^k do v_e ścieżka q o minimalnym czasie (wiersz 12). Jeżeli istnieje możliwość wyznaczenia takiej ścieżki, to jest tworzone nowe rozwiązanie, tj. ścieżka p z wierzchołka v_s do v_e, powstająca ze złączenia ścieżek: podścieżki $sub_{pk}(v_s, v_i^k)$ ścieżki p_k i ścieżki q (wiersz 14). Jeżeli ścieżka p jest rozwiązaniem akceptowalnym, tj. są spełnione dla niej zależności $T(p) \le T_{\max}$, $C(p_i) + \Delta c \le C_{\max}$, to jest wstawiana do zbioru rozwiązań R, a wierzchołkiem $d_1(p)$ jest wierzchołek v_1^q będący następnikiem wierzchołka v_i^k w ścieżce (wiersz 16). Następnie jest usuwany z grafu łuk $(v_0^q = v_i^k, v_1^q)$ należący do ścieżki q (wiersz 18) i zostanie podjęta próba wyznaczenia kolejnej ścieżki z wierzchołka v_i^k.

Wyznaczanie ścieżek z wierzchołka v_i^k jest przerywane, jeżeli nie ma możliwości wyznaczenia kolejnej ścieżki lub czas przejazdu uzyskanej ścieżki p przekracza wartość $T_{\max}$ (wiersz 20). W takim przypadku jest usuwany z grafu wierzchołek v_i^k (wiersz 21) i jest podejmowana próba wyznaczenia nowych ścieżek z następnego wierzchołka należącego do ścieżki p_k. Po przeanalizowaniu wszystkich wierzchołków podścieżki $sub_{pk}(d_1(p_k), v_{j-1}^k)$ są umieszczane w grafie wszystkie usunięte łuki i wierzchołki (wiersz 23) oraz analizie zostanie poddana kolejna nieanalizowana ścieżka ze zbioru R (wiersz 24). Po dokonaniu analizy wszystkich ścieżek są usuwane ze zbioru R rozwiązania zdominowane (wiersz 26), który następnie jest zwracany jako wynik wykonania algorytmu (wiersz 27).

3.6. Metoda skalaryzacji

Ostatnim algorytmem wyznaczania połączeń komunikacyjnych jest algorytm *SSP* przedstawiony w pracy [51], w którym do wyznaczania połączeń jest użyta metoda skalaryzacji oraz, podobnie jak w algorytmie *MKSP*, metoda tworzenia drzewa ścieżek. Na podstawie zależności (2) w algorytmie jest wprowadzana zastępcza funkcja kryterialna (12), w której czas przejazdu T jest uwzględniany z wagą λ_T, a koszt przejazdu C z wagą λ_C. Wartości wag λ_T i λ_C są określone zależnościami (13) oraz (14) i są wyznaczane na podstawie układu równań (15). Zależności (13) i (14) obowiązują wyłącznie w sytuacji, kiedy wagi ścieżek

o minimalnym czasie i koszcie przejazdu są różne. Jeżeli są one równe, to wagi λ_T i λ_C przyjmują wartość 0.5.

$$F_Z = \lambda_T T + \lambda_C C \tag{12}$$

$$\lambda_T = \frac{C_{\max} - C_{\min}}{C_{\max} - C_{\min} + T_{\max} - T_{\min}} \tag{13}$$

$$\lambda_C = \frac{T_{\max} - T_{\min}}{C_{\max} - C_{\min} + T_{\max} - T_{\min}} \tag{14}$$

$$\begin{cases} \lambda_T T_{\max} + \lambda_C C_{\min} = \lambda_T T_{\min} + \lambda_C C_{\max} \\ \lambda_T + \lambda_C = 1 \end{cases} \tag{15}$$

```
   Wejście: G = (V, E) - graf skierowany reprezentujący sieć komunikacyjną
            v_s        - wierzchołek reprezentujący przystanek początkowy
            v_e        - wierzchołek reprezentujący przystanek końcowy
            T_s        - godzina rozpoczęcia podróży na przystanku początkowym
   Wyjście: R          - zbiór rozwiązań, tj. ścieżek reprezentujących
                         połączenia komunikacyjne
 1: procedure SSP(G, v_s, v_e, T_s, R)
 2: begin
 3:   R := ∅;   S ⇐ ∅;   koniec := false;
 4:   p := ścieżka z v_s do v_e o minimalnym koszcie przejazdu;   R := R ∪ {p};
 5:   p := ścieżka z v_s do v_e o minimalnym czasie przejazdu;    R := R ∪ {p};
 6:   wyznacz wartości λ_T, λ_C i F_max; p_akt := p; i := 0;
 7:   while not koniec do
 8:     while i < len(p_akt) and nie wyznaczono nowej ścieżki do
 9:       usuń z grafu G łuk (v_i, v_i+1) należący do ścieżki p_akt;
10:       p := ścieżka z v_i do v_e o minimalnej wartości F_Z nie większej niż F_max;
11:       if nie istnieje ścieżka p then
12:         usuń z grafu G wierzchołek v_i;   i := i + 1;
13:       end if
14:     end while
15:     if znaleziono nową ścieżkę p then
16:       R := R ∪ {p};   S ⇐ (p_akt, i);   p_akt := p;    i := d(p_akt);
17:     else
18:       umieść w grafie G usunięte wierzchołki i łuki dla ścieżki p_akt;
19:       if S ≠ ∅ then       {stos nie jest pusty}
20:         (p_akt, i) ⇐ S;    usuń z grafu G wierzchołek v_i;   i := i + 1;
21:       else
22:         koniec := true;   {zakończenie obliczeń}
23:       end if
24:     end if
25:   end while
26:   usuń rozwiązania zdominowane ze zbioru R;
27:   return R;
28: end
```

Wyznaczone rozwiązania są zapamiętywane w zbiorze rozwiązań R, dodatkowo użyty jest stos S zawierający analizowane ścieżki, na podstawie których są wyznaczane nowe ścieżki. W części inicjalizacyjnej algorytmu są wyznaczane i umieszczane w zbiorze rozwiązań R ścieżki o minimalnym koszcie i czasie przejazdu (wiersze 4–5). Na ich podstawie, zgodnie z zależnościami (13) i (14), są wyznaczane wartości wag λ_T i λ_C, a także wartość $F_{\max}$ (wiersz 6), która jest równa wartości zastępczej funkcji kryterialnej F_Z dla ścieżki

o minimalnym czasie przejazdu (16)[6]. W zbiorze rozwiązań R będą zapamiętywane wyłącznie rozwiązania, dla których wartość F_Z nie przekracza $F_{\max}$. Ścieżka o minimalnym czasie przejazdu staje się pierwszą analizowaną ścieżką p_{akt} (wiersz 6), na podstawie której zostanie podjęta próba wyznaczenia nowych ścieżek, a indeks i określa indeks pierwszego analizowanego wierzchołka w ścieżce, a jest nim w tym przypadku wierzchołek v_s.

$$F_{\max} = \lambda_T T_{\min} + \lambda_C C_{\max} \tag{16}$$

Nowe rozwiązania są wyznaczane w iteracji **while** (wiesze 7–25) na podstawie ścieżki p_{akt}. W iteracji **while** (wiersze 8–14) jest podejmowana próba wyznaczenia nowej ścieżki (wiersz 10) z każdego wierzchołka $\{d(p_{\text{akt}}), \ldots, v_{n-1}\}$ należącego do ścieżki p_{akt}, do wierzchołka v_e. Przed wyznaczeniem nowej ścieżki jest usuwany z grafu G łuk (v_i, v_{i+1}) należący do ścieżki p_{akt}, wychodzący z wierzchołka o indeksie i (wiersz 9).

Jeżeli nie ma możliwości wyznaczenia nowej ścieżki z wierzchołka v_i, następuje usunięcie z grafu tego wierzchołka i przejście do analizy następnego wierzchołka należącego do ścieżki p_{akt} (wiersze 11–13). W przypadku wyznaczenia nowej ścieżki p jest ona dodawana do zbioru rozwiązań R, a na stosie S jest umieszczana aktualnie analizowana ścieżka p_{akt} wraz z indeksem i wierzchołka, na którym zakończyła się analiza ścieżki p_{akt} (wiersz 16). Dalsza analiza będzie kontynuowana dla wyznaczonej ścieżki p, która staje się ścieżką p_{akt}, a rozpocznie się od wierzchołka odchylającego $d(p_{\text{akt}})$ (wiersz 16).

W przypadku braku możliwości wyznaczenia nowej ścieżki na podstawie ścieżki p_{akt}, następuje umieszczenie w grafie G wszystkich wierzchołków i łuków, które zostały usunięte podczas analizy bieżącej ścieżki p_{akt} (wiersz 18). Następnie są pobierane ze stosu S ścieżka, która zostanie poddana analizie, oraz indeks i wierzchołka, na którym zakończyła się poprzednia analiza pobranej ścieżki (wiersz 20). Jeżeli stos S jest pusty, to następuje zakończenie wyszukiwania rozwiązań przez nadanie znacznikowi *koniec* wartości **true** (wiersz 22). W ostatnim kroku algorytmu są usuwane rozwiązania zdominowane ze zbioru rozwiązań R (wiersz 26).

3.7. Wyniki badań eksperymentalnych

Badania eksperymentalne z użyciem algorytmów *DFSB*, *BFS*, *MKSP*, *SSP* zostały przeprowadzone z użyciem komputera PC z procesorem Pentium IV 2.4 GHz. W badaniach użyto sieci komunikacyjnej składającej się z 1211 przystanków, podzielonej na 26 stref, w której kursują pojazdy 500 linii komunikacyjnych. Maksymalna długość trasy linii

[6] Wartość $F_{\max}$ jest równa także wartości zastępczej funkcji kryterialnej F_Z dla ścieżki o minimalnym koszcie przejazdu.

komunikacyjnej jest równa 29, a minimalna 6[7]. Graf G opisujący sieć komunikacyjną zawiera 1211 wierzchołków, 5549 łuków i ma następujące parametry:

- maksymalny stopień wejściowy i wyjściowy wierzchołka są równe 13,
- maksymalny stopień wierzchołka jest równy 26,
- minimalny stopień wejściowy i wyjściowy są równe 1,
- minimalny stopień wierzchołka jest równy 2.

Tabela 2

Zależność maksymalnego czasu wyznaczania rozwiązań z użyciem algorytmów *DFSB*, *BFS*, *MKSP*, *SSP* od liczby przesiadek

Liczba przesiadek	Maksymalny czas wyznaczania rozwiązań [min:s]			
	Algorytm *DFSB*	Algorytm *BFS*	Algorytm *MKSP*	Algorytm *SSP*
0	0:07	0:23	0:31	0:02
1	1:22	1:41	2:04	0:10
2	3:35	6:59	9:25	0:12
3	7:47	12:14	13:28	0:50
4	8:47	14:35	14:18	0:52
5	9:03	15:37	15:18	0:55
6	11:28	18:08	17:29	2:18

Tabela 3

Zależność maksymalnego czasu wyznaczania rozwiązań z użyciem algorytmów *DFSB*, *BFS*, *MKSP*, *SSP* od długości trasy przejazdu

Długość trasy	Maksymalny czas wyznaczania rozwiązań [min:s]			
	Algorytm *DFSB*	Algorytm *BFS*	Algorytm *MKSP*	Algorytm *SSP*
1–5	0:07	0:23	0:31	0:02
6–10	0:37	1:01	1:26	0:05
11–15	1:22	1:41	2:04	0:10
16–20	2:03	2:29	3:21	0:09
21–25	3:35	6:59	9:25	0:12
26–30	7:47	12:14	13:28	0:50
31–35	11:28	18:08	17:29	2:18
36–40	8:47	14:35	14:18	0:52
41–45	7:47	12:14	13:28	0:50
46–50	9:03	15:37	15:18	0:55
51–55	11:03	15:53	16:32	1:17
56–60	11:28	18:08	17:29	2:18

Przeprowadzonych zostało 1 371 616 eksperymentów. Pojedynczy eksperyment polegał na wyznaczeniu połączeń komunikacyjnych dla zadanej pary przystanków początkowego i końcowego oraz godziny rozpoczęcia podróży. W tabelach 2 i 3 przedstawiono w formacie

[7] Długość trasy linii komunikacyjnej mierzona jest liczbą przystanków należących do trasy.

min:s maksymalny czas wyznaczania rozwiązania z użyciem każdego z badanych algorytmów. Czas wyznaczania rozwiązania został przedstawiony w zależności od liczby dokonanych przesiadek w rozwiązaniu (tabela 2) i długosci trasy przejazdu (tabela 3).

Najmniejszy czas obliczeń został uzyskany dla algorytmu *SSP*, co wynika z metody skalaryzacji użytej podczas wyznaczania rozwiązań. W metodzie tej jest przeszukiwany najmniejszy obszar przestrzeni rozwiązań, jednak nie ma gwarancji wyznaczenia wszystkich rozwiązań niezdominowanych. W pozostałych algorytmach są wyznaczane wszystkie rozwiązania niezdominowane. Najmniejszy czas obliczeń został uzyskany dla algorytmu *DFSB*, a czasy obliczeń dla algorytmów *BFS* i *MKSP* są porównywalne. W algorytmie *BFS* dla każdego analizowanego wierzchołka są tworzone i zapamiętywane rozwiązania, co wpływa na większy czas obliczeń niż w przypadku algorytmu *DFSB*. W algorytmie *MKSP* wpływ na czas wyznaczania rozwiązań ma każdorazowe wyznaczanie ścieżki z każdego analizowanego wierzchołka do wierzchołka końcowego v_e.

4. Inne badane problemy związane z wyznaczaniem tras przejazdu

Problem opisany w rozdziale 3 został także przedstawiony w pracy [41], w której zaproponowano jego rozwiązanie za pomocą algorytmu wyszukiwania wyczerpującego. Zaproponowany algorytm został przebadany dla rzeczywistej sieci komunikacyjnej obejmującej miasta Gliwice i Zabrze (sieć składająca się z 60 linii autobusowych i ok. 200 przystanków). Ze względu na złożoność czasową zaproponowanego algorytmu nie było możliwe wykonanie badań dla sieci złożonej z większej liczby przystanków.

Przez pracowników Instytutu Informatyki były badane także inne problemy związane z wyznaczaniem tras przejazdu. W pracy [22], na przykładzie jednej z ówczesnych firm spedycyjnych działających na terenie Polski, omówiono problem doboru trasy transportu przesyłki oraz ustalania rozkładu jazdy pojazdów transportowych. Trasa transportu przesyłki jest ustalana w sieci transportowej, która składa się z N węzłów sortujących. Pomiędzy poszczególnymi węzłami są utrzymywane stałe połączenia komunikacyjne umożliwiające transport przesyłek. Węzły sortujące różną się między sobą funkcjami, stąd są podzielone na 3 klasy: A, B i C. W sieci znajduje się N_A węzłów klasy A, N_B węzłów klasy B oraz N_C węzłów klasy C, gdzie $N_A \neq 0$, $N_A + N_B + N_C = N$, $N_A < N_B < N_C$.

W węzłach klasy A odbywa się sortowanie zasadnicze, dlatego każda przesyłka musi przejść przez co najmniej jeden węzeł tej klasy. Każdy węzeł klasy A ma połączenie komunikacyjne z wszystkimi pozostałymi węzłami klasy A oraz z pewną liczbą węzłów klasy B i C. W węzłach klasy B odbywa się sortowanie wtórne, polegające na przeładunku kontenerów z przesyłkami pomiędzy różnymi środkami transportu. Koszt sortowania w tych węzłach

jest znacznie niższy od kosztu sortowania w węzłach klasy *A*. Każdy węzeł klasy *B* ma połączenie komunikacyjne z co najmniej jednym węzłem klasy *A* i co najmniej dwoma węzłami klasy *C*, nie ma natomiast połączeń z innymi węzłami klasy *B*. Najbardziej ograniczone funkcje mają węzły klasy *C*, ich zadaniem jest wyłącznie zbieranie przesyłek z określonego obszaru i rozwożenie ich do odbiorców. Funkcję tę spełniają także węzły klasy *A* i *B*. Każdy węzeł klasy *C* ma tylko jedno połączenie z węzłem klasy *A* lub *B*.

Dla zadanej sieci transportowej należy wyznaczyć trasę transportu przesyłki z węzła źródłowego v_s do docelowego v_e tak, aby czas i koszt transportu były minimalne. Czas transportu jest sumą czasów przejazdu pomiędzy węzłami sortującymi oraz czasów sortowania, przeładunku i załadunku w węzłach, a koszt transportu jest sumą kosztów przejazdu pomiędzy węzłami oraz kosztów sortowania, przeładunku i załadunku w węzłach.

Rozwiązanie problemu doboru trasy transportu przesyłek sprowadza się do rozwiązania dwukryterialnego problemu najkrótszej ścieżki w grafie o zmiennych wagach. W pracy [48] przedstawiono algorytm umożliwiający wyznaczenie wszystkich tras będących rozwiązaniami niezdominowanymi. Algorytm jest modyfikacją algorytmu korygowania etykiet z pamiętaniem rozwiązań częściowych, który został przedstawiony w punkcie 3.4.

Z wyznaczaniem tras przejazdu pojazdów jest także związany problem dostawy z oknami czasowymi (ang. *Vehicle Routing Problem with Time Windows*, VRPTW), który jest znanym problemem optymalizacji dyskretnej, należącym do klasy NP. Dokładne sformułowanie problemu jest dostępne np. w pracy [21]. Zostało opublikowanych wiele prac, w których przedstawiono algorytmy zarówno sekwencyjne, jak i równolegle rozwiązywania problemu VRPTV, a których autorami są pracownicy Instytutu Informatyki. W pracy [9] przedstawiono rozwiązanie problemu za pomocą równoległego algorytmu symulowanego wyżarzania (ang. *simulated annealing*) oraz wyniki badań eksperymentalnych, które zostały przeprowadzone z użyciem zbioru testowego Solomona [55]. Warto podkreślić, że dla danych RC203 uzyskane rozwiązanie jest najlepszym z dotychczas znalezionych rozwiązań (stan na dzień 1.09.2011). Rozwiązanie problemu z użyciem równoległego algorytmu symulowanego wyżarzania przedstawiono także w pracy [52], w której zamieszczono trzy różne algorytmy, które przebadano korzystając także ze zbioru testowego Solomona. Równoległe algorytmy symulowanego wyżarzania zostały przedstawione także w pracach [11, 12], a w pracy [13] przedstawiono teoretyczną analizę uzyskanej wartości przyspieszenia dla równoległego algorytmu symulowanego wyżarzania. Rozważania teoretyczne zostały zweryfikowane przez przeprowadzone badania eksperymentalne, podczas których użyto równoległego algorytmu symulowanego wyżarzania dla problemu VRPTW i zbioru danych testowych Solomona.

W pracy [16] przedstawiono i dokonano interpretacji wyników dla danych testowych Hombergera [54]. Badania zostały przeprowadzone z użyciem sekwencyjnego dwufazowego

algorytmu symulowanego wyżarzania będącego adaptacją algorytmu przedstawionego w pracy [9]. Celem badań było określenie przydatności sekwencyjnego algorytmu do rozwiązywania problemu dla liczby klientów od 200 do 1000. Z kolei w pracy [17] przedstawione jest rozumienie trudności instancji problemu w odniesieniu do złożoności obliczeniowej średniej i pesymistycznej algorytmów. Zostały zaproponowane miary dla trudności w przypadku heurystyki symulowanego wyżarzania do rozwiązywania problemu VRPTW. Badana jest zależność pomiędzy sformułowanymi miarami trudności a wartościami charakteryzującymi jakość otrzymanych rozwiązań, które pochodzą z serii obliczeń wykonanych z użyciem zbioru danych testowych Hombergera.

Uproszczonym wariantem VRPTW jest problem dostawy, w którym nie są uwzględniane okna czasowe (ang. *Vehicle Routing Problem*, VRP). W pracy [40] zostało przedstawione jego rozwiązanie z użyciem sekwencyjnego algorytmu przeszukiwania tabu (ang. *tabu serach*). Zostały zaproponowane 4 warianty algorytmu różniące się sposobem przeszukiwania sąsiedztwa bieżącego rozwiązania. Do rozwiązania problemu VRP został także użyty algorytm mrówkowy (ang. *ant colony*), co zostało przedstawione w pracy [35]. W pracy tej skupiono się na badaniu algorytmu z modyfikacją widoczności klienta i z klientem j, przy czym zostały zbadane dwa warianty. W pierwszym wariancie widoczność jest równa odwrotności odległości klienta i od j, natomiast w drugim wariancie jako funkcja liniowa odległości tych klientów. Rozwiązania o lepszej jakości zostały uzyskane dla drugiego wariantu widoczności.

5. Podsumowanie

W niniejszej pracy zostało przedstawione podsumowanie wyników badań, jakie zostały wykonane w latach 2001–2011, w ramach których dokonano rozwiązania problemu wyznaczania połączeń w sieciach komunikacyjnych. Celem badanego problemu jest, dla zadanej pary przystanków początkowego i końcowego oraz godziny rozpoczęcia podróży, wyznaczenie połączeń komunikacyjnych minimalizując jednocześnie czas i koszt przejazdu. Rozwiązanie niniejszego problemu sprowadza się do rozwiązania dwukryterialnego problemu wyznaczania najkrótszej ścieżki w grafie o zmiennych wagach. Problem ten jest przykładem zadania optymalizacji wielokryterialnej, którego rozwiązaniem w ogólnym przypadku jest zbiór rozwiązań niezdominowanych. W badanym problemie kryteriami jakości, na podstawie których jest dokonywana ocena rozwiązań (połączeń komunikacyjnych), są czas i koszt przejazdu.

Zostały przedstawione 4 algorytmy umożliwiające wyznaczenie połączeń należących do zbioru rozwiązań niezdominowanych: algorytmy z usuwaniem (*DFSB*) oraz pamiętaniem

rozwiązań częściowych (*BFS*), algorytm, w którym rozwiązania są wyznaczane na podstawie wyznaczenia *K* najkrótszych ścieżek (*MKSP*) oraz algorytm, w którym do wyznaczania rozwiązań jest użyta metoda skalaryzacji (*SSP*). Pierwsze 3 algorytmy umożliwiają wyznaczenie wszystkich połączeń stanowiących zbiór rozwiązań niezdominowanych, natomiast w algorytmie *SSP* nie ma gwarancji wyznaczenia wszystkich rozwiązań niezdominowanych, co wynika z własności użytej metody.

Oprócz problemu wyznaczania połączeń w sieciach komunikacyjnych, przez pracowników Instytutu Informatyki były także badane inne problemy związane z wyznaczaniem tras przejazdu pojazdów. Wśród badanych problemów znalazły się: problem wyznaczania optymalnej trasy transportu przesyłek przez firmę kurierską, problem dostawy z oknami czasowymi oraz bez uwzględniania okien czasowych. Dla problemu dostawy zostały opracowane zarówno algorytmy sekwencyjne, jak i równoległe.

BIBLIOGRAFIA

1. Aho A. V., Hopcroft J. E., Ullman J. D.: Algorytmy i struktury danych. Wydawnictwo Helion, Gliwice 2003.
2. Aho A. V., Hopcroft J. E., Ullman J. D.: Projektowanie i analiza algorytmów. Wydawnictwo Helion, Gliwice 2003.
3. Brumbaugh–Smith J., Shier S.: An empirical investigation of some bicriterion shortest path algorithms. European Journal of Operational Research, Vol. 43, 1989, s. 216÷224.
4. Climaco J. C., Martins E. Q. V.: A bicriterion shortest path algorithm. European Journal of Operational Research, Vol. 11, No. 4, 1982, s. 399÷404.
5. Coello Coello C. A., van Veldhuizen D. A., Lamont G. B.: Evolutionary algorithms for solving multi–objective problems. Kluwer Academic Publishers, New York 2002.
6. Corley H. W., Moon I. D.: Shortest paths in networks with vector weights. Journal of Optimization Theory and Application, Vol. 46, No. 1, 1985, s. 79÷86.
7. Coutinho–Rodrigues J. M., Climaco J. C., Current J. R.: An interactive bi–objective shortest path approach: searching for unsupportet nondominated solutions. Computers & Operations Research, Vol. 26, No. 8, 1999, s. 789÷798.
8. Cormen T. H., Leiserson Ch. E., Rivest R. L.: Wprowadzenie do algorytmów. WNT, Warszawa 2000.
9. Czech Z. J., Czarnas P.: Parallel Simulated Annealing for the Vehicle Routing Problem with Time Windows. Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network–based Processing, Canary Islands, Spain, January 9–11, 2002, s. 376÷383.

10. Daellenbach H. G., De Kluyver C. A.: Note on multiple objective dynamic programming. Journal of the Operational Research Society, Vol. 31, No. 7, 1980, s. 591÷594.
11. Debudaj–Grabysz A.: Równoległe algorytmy symulowanego wyżarzania. Rozprawa doktorska, Politechnika Śląska, Gliwice 2006.
12. Debudaj–Grabysz A., Czech Z. J.: A Concurrent Implementation of Simulated Annealing and Its Application to the VRPTW Optimization Problem. Distributed and parallel systems, The Kluwer International Series in Engineering and Computer Science, Vol. 777, Part VI, 2005, s. 201÷209.
13. Debudaj–Grabysz A., Czech Z. J.: Theoretical and practical issues of parallel simulated annealing. Parallel Processing and Applied Mathematics, Lecture Notes in Computer Science, Vol. 4967, Springer, Berlin 2008, s. 189÷198.
14. Dell'Olmo P., Gentili M., Scozzari A.: On finding dissimilar Pareto–optimal paths. European Journal of Operational Research, Vol. 162, No. 1, 2005, s. 70÷82.
15. Dijkstra E. W.: A note on two problems in connexion with graphs. Numerical Mathematics, Vol. 1, 1959, s. 269÷271.
16. Gandor T.: Przybliżone rozwiązania problemu trasowania pojazdów z przedziałami czasowymi dla dużej liczby klientów. Systemy Wspomagania Decyzji, Instytut Informatyki Uniwersytetu Śląskiego, Katowice 2009, s. 257÷277.
17. Gandor T.: Ocena trudności egzemplarzy problemu dostawy z oknami czasowymi. Systemy Wspomagania Decyzji, Instytut Informatyki Uniwersytetu Śląskiego, Katowice 2010, s. 153÷168.
18. Garey M., Johnson D.: Computers and intractibility: a guide to the theory of NP–completeness. W. H. Freeman & Co., New York, USA, 1990.
19. Hansen P.: Bicriterion path problems. Multiple Criteria Decision Making: Theory and Application, Springer–Verlag , Berlin, Germany, 1980, s. 109÷127.
20. Jungnickel D.: Graphs, networks and algorithms. Springer, Berlin 1999.
21. Kohl N., Madsen O. B. G.: An Optimization Algorithm for with Time Windows based on Lagrangian Relaxation. Operations Research, Vol. 45, No. 3, 1997, s. 395÷406.
22. Kuczora M.: Optymalizacja transportu przesyłek przy wykorzystaniu stałych linii komunikacyjnych i wyróżnionych węzłów sortujących. Studia Informatica, Vol. 23, No. 4(51), 2001, s. 105÷124.
23. Lipski W.: Kombinatoryka dla programistów. WNT, Warszawa 1989.
24. Machuca E., Mandow L., Pérez de la Cruz J. L.: An Evaluation of Heuristic Functions for Bicriterion Shortest Path Problems. Proceedings of the XIV Portuguese Conference on Artificial Inteligence (EPIA 2009), Universidade de Aveiro, 2009, s. 205÷216.

25. Mandow L., Pérez de la Cruz J. L.: Frontier search for Bicriterion Shortest Path Problems. Proceeding of the 2008 conference on ECAI 2008 18th European Conference on Artificial Intelligence, IOS Press, 2008, s. 480÷484.
26. Mandow L., Pérez de la Cruz J. L.: Path recovery in frontier search for multiobjective shortest path problems. Journal of Intelligent Manufacturing, Vol. 21, No. 1, 2008, s. 89÷99.
27. Martins E. Q. V.: On a multicriteria shortest path problem. European Journal of Operational Research, Vol. 16, No. 2, 1982, s. 236÷245.
28. Martins E. Q. V., Pascoal M. M. B.: An algorithm for ranking optimal paths. Departamento de Matematica, Universidade de Coimbra, Technical Report 01/004, CISUC, 2000 (http://www.mat.uc.pt/~marta/Publicacoes/rank_optimal.ps.gz).
29. Martí R., González Velarde J. L., Duarte A.: Heuristics for the bi–objective path dissimilarity problem. Computers & Operations Research, Vol. 36, No. 11, 2009, s. 2905÷2912.
30. Mote J., Murthy I., Olson D. L.: A parametric approach to solving bicriterion shortest path problems. European Journal of Operational Research, Vol. 53, No. 1, 1991, s. 81÷92.
31. Pareto V.: Course d'Economie Politique, F. Rouge, Lausanne 1896.
32. Peschel M., Riedel C.: Polioptymalizacja. Metody podejmowania decyzji kompromisowych w zagadnieniach inżynieryjno-technicznych. WNT, Warszawa 1979.
33. Raith A., Ehrgott M.: A comparison of solution strategies for biobjective shortest path problems. Journal Computers and Operations Research, Vol. 36, No. 4, 2009, s. 1299÷1331.
34. Reingold E. M., Nievergelt J., Deo N.: Algorytmy kombinatoryczne. PWN, Warszawa 1985.
35. Skórczyński A.: Modyfikacja widoczności w algorytmach mrówkowych rozwiązujących problem dostawy. Studia Informatica, Vol. 22, nr 4(46) , 2001, s. 235÷244.
36. Skriver A. J. V., Andersen K.A.: A classification of bicriteria shortest path (BSP) algorithms. Asia–Pacific Journal of Operational Research, Vol. 17, 2000, s. 199÷212.
37. Skriver A. J. V., Andersen K.A.: A label correcting approach for solving bicriterion shortest-path problems. Computers & Operations Research, Vol. 27, 2000, s. 507÷524.
38. Stadler W.: A survey of Multicriteria Otimization or the Vector Maximum Problem. Part I: 1776–1960, Journal of Optimization Theory & Aplication, Vol. 29, No. 1, 1979, s. 1÷52.
39. Sysło M. M., Deo N., Kowalik J. S.: Algorytmy optymalizacji dyskretnej z programami w języku Pascal. Wydawnictwo Naukowe PWN, Warszawa 1995.

40. Szołtysek M.: Rozwiązanie problemu dostawy za pomocą algorytmu przeszukiwania tabu. ZN Pol. Śląskiej, seria: Informatyka, z. 37, nr kol. 1421, 1999, s. 209÷221.
41. Szołtysek M.: Problem wyznaczania połączeń autobusowych. Studia Informatica, Vol. 21, No. 3, 2000, s. 187÷200.
42. Tung C. T., Chew K. L.: A bicriterion Pareto–optimal path algorithm. Asia–Pacific Journal of Operational Research, Vol. 5, 1988, s. 166÷172.
43. Ulungu E. L., Teghem J.: Multi–objective combinatorial optimization problems: a survey. Journal of Multi-Criteria Decision Analysis, Vol. 3, No. 2, 1994, s. 83÷104.
44. Voorneveld M.: Characterization of Pareto dominance. Operations Research Letters, Vol. 31, No. 1, 2003, s. 7÷11.
45. White D. J.: The set of efficient solutions for multiple objective shortest path problems. Computers & Operations Research, Vol. 9, No. 2, 1982, s. 101÷107.
46. Widuch J.: Problem wyznaczania połączeń w sieciach komunikacyjnych. Studia Informatica, Vol. 22, No. 4(46), Gliwice 2001, s. 117÷134.
47. Widuch J.: Wyznaczanie połączeń w sieciach komunikacyjnych o zmiennych wagach. Studia Informatica, Vol. 23, No, 4(51), Gliwice 2002, s. 85÷104.
48. Widuch J.: Problem wyznaczania optymalnej trasy transportu przesyłek. Studia Informatica, Vol. 24, No. 4(56), Gliwice 2003, s. 67÷84.
49. Widuch J.: Algorytmy optymalizacji wielokryterialnej w problemach komunikacyjnych. Rozprawa doktorska, Politechnika Śląska, Gliwice 2008.
50. Widuch J.: Rozwiązanie problemu wyznaczania połączeń w sieciach komunikacyjnych za pomocą zmodyfikowanego algorytmu wyznaczania *K* najkrótszych ścieżek. Studia Informatica, Vol. 31, No. 1(88), Gliwice 2010, s. 55÷70.
51. Widuch J.: Rozwiązanie problemu wyznaczania połączeń w sieciach komunikacyjnych z zastosowaniem metody skalaryzacji. Studia Informatica, Vol. 32, No. 4A(100), Gliwice 2011.
52. Wieczorek B.: Równoległe algorytmy symulowanego wyżarzania dla problemu trasowania pojazdów z ograniczeniami czasowymi. Studia Informatica, Vol. 26, No 1(62), Gliwice 2005, s. 93÷105.
53. Wilson R. J.: Wprowadzenie do teorii grafów. Wydawnictwo Naukowe PWN, Warszawa 1998.
54. Zbiory testowe Hombergera dla problemu VRPTW (1.09.2011): http://www.fernuni-hagen.de/WINF/touren/inhalte/probinst.htm
55. Zbiory testowe Solomona dla problemu VRPTW (1.09.2011): http://web.cba.neu.edu-/~msolomon/problems.htm.

Recenzent: Dr hab. inż. Mariusz Boryczka, prof. UŚ

Wpłynęło do Redakcji 10 października 2011 r.

Abstract

The routing problem is one of the studied transportation problems. The transportation network is represented by directed graph $G = (V, E)$ with the weight function $w : E \rightarrow R$, and solving the routing problem consists to determining the path with minimum weight in the graph G. In many cases, using only single the weight function is insufficient because the determined solution does not include all the essential features of a given problem. Therefore k ($k > 1$) the weight functions are used, which are also taken into consideration during determination of the path in the graph. When $k = 2$, then the problem is called bicriterion shortest path (BSP) problem. The BSP problem is an example of mulitcriteria optimization problem which the solution is the set of non–dominated solutions.

In the paper, which is a summary of research that have been made in the years 2001–2011, the communication networks routing (CNR) problem is presented as an example of the BSP problem. The goal of the CNR is finding a route from the start stop to the final stop minimizing the time and the cost of travel. Additionally the time of starting travel at the start stop is given. Four algorithms for determination routes belong to the set of non–dominated solutions are shown. The following algorithms have been presented: a label correcting algorithm with deleting partial solutions, a label correcting algorithm with storing partial solutions, the K–th shortest path algorithm and algorithm where the solutions are determined using scalarization method. Presented algorithms were implemented and tested for a random generated bus network. The computational results are also presented. We take into consideration only two criterion functions which are the time and the cost of travel but it is possible to add other criterion functions and use presented algorithms.

Additionally, other routing problems that have been researched by staff of the Institute of Informatics are discussed in the paper. The problem of determining the optimal route of packages performed by courier, Vehicle Routing Problem with Time Windows and Vehicle Routing Problem where time windows are not complied are presented.

Adres

Jacek WIDUCH: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16,
44-100 Gliwice, Polska, jacek.widuch@polsl.pl.