Wspomaganie Decyzji II

Roman Słowiński

Zakład Inteligentnych Systemów
Wspomagania Decyzji

Instytut Informatyki

Politechniki Poznańskiej

Specjalne problemy programowania liniowego

- Problemy decyzyjne modelowane w kategoriach programowania matematycznego, sprowadzalne do programowania liniowego
- Problem programowania ilorazowego

$$z = \frac{\underline{c}\underline{x} + c_0}{\underline{d}\underline{x} + d_0} \rightarrow MIN$$

przy ograniczeniach :

$$A\underline{x} = \underline{b}$$

$$\underline{x} \geq \underline{0}, \quad \underline{d}\underline{x} + d_0 > 0$$

gdzie

$$\underline{x} = [x_1, \ldots, x_n]^T, \quad \underline{b} = [b_1, \ldots, b_m]^T$$

$$\underline{c} = [c_1, \ldots, c_n], \quad \underline{d} = [d_1, \ldots, d_n]$$

Transformacja Charnesa-Coopera

Specjalne problemy programowania liniowego

- Problem programowania celowego

$$z = \sum_{h=1}^{k} w_h \left| \underline{c}_h \underline{x} - c_{h0} \right| \rightarrow MIN$$

przy ograniczeniach :

$$A\underline{x} = \underline{b}$$

$$\underline{x} \geq \underline{0}$$

gdzie

$$\underline{x} = [x_1, \ldots, x_n]^T, \quad \underline{b} = [b_1, \ldots, b_m]^T$$

$$\underline{c}_h = [c_{h1}, \ldots, c_{hn}], \quad h = 1, \ldots, k$$

Specjalne problemy programowania liniowego

- Problem programowania min-max (problem Czebyszewa)

$$z = \underset{h=1,\ldots,k}{MAX} \left\{ w_h \left(\underline{c}_h \underline{x} - c_{h0} \right) \right\} \rightarrow MIN$$

przy ograniczeniach :

$$A\underline{x} = \underline{b}$$

$$\underline{x} \geq \underline{0}$$

gdzie

$$\underline{x} = \left[x_1, \ldots, x_n \right]^T, \quad \underline{b} = \left[b_1, \ldots, b_m \right]^T$$

$$\underline{c}_h = \left[c_{h1}, \ldots, c_{hn} \right], \quad h = 1, \ldots, k$$

Specjalne problemy programowania liniowego

- Problem transportowy

  m – liczba punktów nadawczych (magazyny)

  n – liczba punktów odbiorczych (klienci)

  c_{ij} – koszt transportu jednostki towaru z magazynu i do odbiorcy j

  d_j – zapotrzebowanie odbiorcy j

  s_i – ilość towaru w magazynie i

  Należy zminimalizować łączne koszty transportu

  Zmienna decyzyjna: x_{ij} – ilość towaru przesłana z magazynu i do odbiorcy j

Specjalne problemy programowania liniowego

- Problem transportowy

$$z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \rightarrow MIN$$

przy ograniczeniach :

$$\sum_{j=1}^{n} x_{ij} = s_i \qquad i = 1,\ldots,m$$

$$\sum_{i=1}^{m} x_{ij} = d_j \qquad j = 1,\ldots,n$$

$$x_{ij} \geq 0 \qquad i = 1,\ldots,m, \quad j = 1,\ldots,n$$

gdzie

$$\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j$$

Problem ten można rozwiązać metodą sympleksów

Specjalne problemy programowania liniowego

- Problem przydziału

  m – liczba zadań (programy)

  m – liczba wykonawców (procesory)

  c_{ij} – koszt wykonania zadania i przez wykonawcę j

  Elementy c_{ij} tworzą macierz efektywności $C=[c_{ij}]$ o wymiarach $m \times m$

  Każde zadanie może być wykonywane przez co najwyżej jednego wykonawcę

  Każdy wykonawca może wykonać tylko jedno zadanie

  Należy tak przydzielić zadania do wykonawców, by zminimalizować łączny koszt wykonania wszystkich zadań

Specjalne problemy programowania liniowego

Zmienna decyzyjna:

$$x_{ij} = \begin{cases} 0, & \textit{gdy wykonawca } i \textit{ nie jest przydzielony do zadania } j \\ 1, & \textit{w przeciwnym razie} \end{cases}$$

$$z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \rightarrow MIN$$

przy ograniczeniach :

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad i = 1,\ldots,m$$

$$\sum_{i=1}^{m} x_{ij} = 1 \qquad j = 1,\ldots,m$$

$$x_{ij} \in \{0,1\} \qquad i = 1,\ldots,m, \quad j = 1,\ldots,m$$

Jest to problem 0-1 programowania liniowego

Jest to także przypadek szczególny problemu transportowego, gdzie $s_i = d_j = 1$, $n=m$ i zmienna decyzyjna jest 0-1

Specjalne problemy programowania liniowego

- Problem transportowy ma tę korzystną własność, że jeżeli s_i i d_j są liczbami całkowitymi i istnieje choćby jedno rozwiązanie dopuszczalne, to istnieje rozwiązanie optymalne, w którym x_{ij} są wszystkie liczbami całkowitymi lub zerami
- Metoda sympleksowa znajduje to całkowitoliczbowe rozwiązanie optymalne
- Dzieje się tak dlatego, gdyż macierz współczynników ograniczeń A problemu transportowego jest unimodularna (wyznacznik dowolnej podmacierzy kwadratowej macierzy A jest = 0, 1 lub -1), a wektor prawych stron ograniczeń $\underline{b}$ jest złożony z liczb całkowitych ($\underline{x}=B^{-1}\underline{b}$)
- Problem przydziału ma zatem tę samą własność
- Istnieje jednak prostsza metoda rozwiązania problemu przydziału

Specjalne problemy programowania liniowego

- Macierz efektywności $C=[c_{ij}]$ o wymiarach $m \times m$

c_{ij}	zadania			
wykonawcy	2	10	9	7
	15	4	14	8
	13	14	16	11
	4	15	13	9

przydział niedopuszczalny

- Przydział wykonawcy do zadania polega na wyborze konkretnego elementu c_{ij}
- Tych m wybranych elementów c_{ij} ma dać minimalną sumę

Specjalne problemy programowania liniowego

- Operacje arytmetyczne na macierzy efektywności $C=[c_{ij}]$ nie powodujące zmiany rozwiązania optymalnego:

c_{ij}	zadania			
wykonawcy	2	10	9	7
	15	4	14	8
	13	14	16	11
	4	15	13	9

Dodanie lub odjęcie dowolnej stałej od dowolnego wiersza lub kolumny macierzy $C=[c_{ij}]$

Np. odejmując 3 od wiersza *i* oraz dodając 2 do kolumny *j* otrzymamy:

$$z = \sum_{i=1}^{m}\sum_{j=1}^{m} c_{ij}x_{ij} - 3\sum_{j=1}^{m} x_{ij} + 2\sum_{i=1}^{m} x_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{m} c_{ij}x_{ij} - 3 + 2$$

gdyż $\sum_{j=1}^{m} x_{ij} = \sum_{i=1}^{m} x_{ij} = 1$, *tzn. funkcja celu ulega tylko przesunięciu o pewna stałą*

Specjalne problemy programowania liniowego

- Odejmijmy zatem najmniejszy element różny od zera w każdym wierszu i w każdej kolumnie:

c_{ij}	zadania				
wykonawcy	2	10	9	7	-2
	15	4	14	8	-4
	13	14	16	11	-11
	4	15	13	9	-4

Specjalne problemy programowania liniowego

- Odejmijmy zatem najmniejszy element różny od zera w każdym wierszu i w każdej kolumnie:

c_{ij}	zadania			
wykonawcy	0	8	7	5
	11	0	10	4
	2	3	5	0
	0	11	9	5

-5

Specjalne problemy programowania liniowego

- Otrzymujemy macierz efektywności $C=[c_{ij}]$ z co najmniej m elementami zerowymi:

c_{ij}	zadania			
wykonawcy	0	8	2	5
	11	0	5	4
	2	3	0	0
	0	11	4	5

- Warunkiem koniecznym optymalnego przydziału jest dokonanie go według współrzędnych elementów zerowych macierzy $C=[c_{ij}]$
- Warunkiem dostatecznym optymalnego przydziału jest niezależność m wybranych elementów zerowych macierzy $C=[c_{ij}]$, według których nastąpił przydział (para elementów niezależnych = elementy w dwóch różnych wierszach i w dwóch różnych kolumnach)

Specjalne problemy programowania liniowego

- Twierdzenie Königa:

  Maksymalna liczba niezależnych elementów zerowych dowolnej macierzy *C* równa jest minimalnej liczbie linii koniecznych do pokrycia wszystkich elementów zerowych tej macierzy

c_{ij}	zadania			
wykonawcy	0	8	2	5
	11	0	5	4
	2	3	0	0
	0	11	4	5

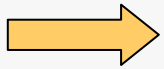
- Na powyższych elementach zerowych macierzy *C* nie da się stworzyć optymalnego przydziału (3 linie = 3 elementy zerowe niezależne < 4)

Specjalne problemy programowania liniowego

- Algorytm *metody węgierskiej* (dla minimalizacji łącznego kosztu przydziału)

1. W każdym wierszu wyznacz minimalny element i odejmij go od każdego elementu tego wiersza.

c_{ij}	zadania				
wykonawcy	2	10	9	7	-2
	15	4	14	8	-4
	13	14	16	11	-11
	4	15	13	9	-4

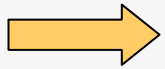


c_{ij}	zadania			
wykonawcy	0	8	7	5
	11	0	10	4
	2	3	5	0
	0	11	9	5

Specjalne problemy programowania liniowego

2. W każdej kolumnie wyznacz minimalny element i odejmij go od każdego elementu tej kolumny.

c_{ij}	zadania			
wykonawcy	0	8	7	5
	11	0	10	4
	2	3	5	0
	0	11	9	5



c_{ij}	zadania			
wykonawcy	0	8	2	5
	11	0	5	4
	2	3	0	0
	0	11	4	5

-5

3. Jeśli w danym wierszu jest dokł. jedno nienaznaczone zero, to naznacz je symbolem Δ i skreśl inne zera w odpowiadającej mu kolumnie.

4. Jeśli w danej kolumnie jest dokł. jedno nienaznaczone zero, to naznacz je symbolem Δ i skreśl inne zera w odpowiadającym mu wierszu.

Specjalne problemy programowania liniowego

5. Kroki 3 – 4 powtarzaj do wyczerpania. Jeśli znaleziono przydział do *m* zer, to jest on optymalny → STOP.
6. Jeśli są jeszcze nienaznaczone zera, to naznacz jedno z nich symbolem Δ („północno-zachodnie”) i skreśl inne zera w odpowiadającej mu kolumnie i wierszu

np.

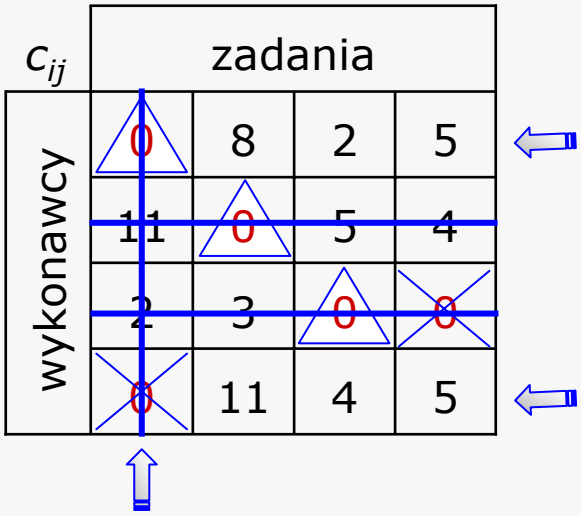
c_{ij}	zadania			
wykonawcy	#	#	#	#
	#	0	~~0~~	~~0~~
	#	~~0~~	0	0
	#	#	#	#

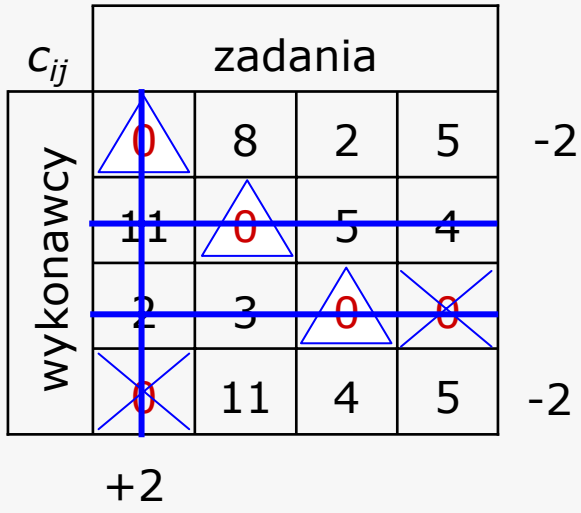
c_{ij}	zadania			
wykonawcy	0	8	2	5
	11	0	5	4
	2	3	0	~~0~~
	~~0~~	11	4	5

7. Zaznacz wiersze bez przydziału Δ.
8. Zaznacz kolumny, które mają zero w dowolnym zaznaczonym wierszu.
9. Zaznacz wiersze, które mają przydział w zaznaczonych kolumnach.
10. Powtarzaj kroki 8 – 9 do wyczerpania.

Specjalne problemy programowania liniowego

11. Pokryj liniami niezaznaczone wiersze i zaznaczone kolumny.



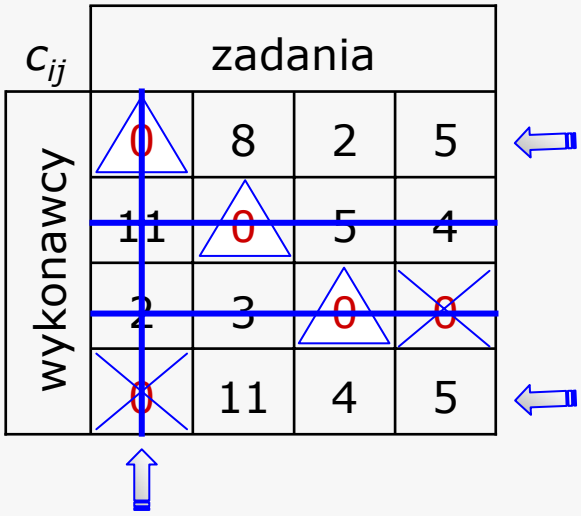


12. Jeśli liczba linii pokrywających wszystkie zera jest równa liczbie naznaczonych zer, to znajdź minimalny niepokryty element, odejmij go od niepokrytych (zaznaczonych) wierszy i dodaj do pokrytych (zaznaczonych) kolumn, po czym wróć do kroku 3.

    W przeciwnym razie przejdź do kroku 13.

Specjalne problemy programowania liniowego

11. Pokryj liniami niezaznaczone wiersze i zaznaczone kolumny.



c_{ij}	zadania			
wykonawcy	0	6	0	3
	13	0	5	4
	4	3	0	0
	0	9	2	3

12. Jeśli liczba linii pokrywających wszystkie zera jest równa liczbie naznaczonych zer, to znajdź minimalny niepokryty element, odejmij go od niepokrytych (zaznaczonych) wierszy i dodaj do pokrytych (zaznaczonych) kolumn, po czym wróć do kroku 3.

    W przeciwnym razie przejdź do kroku 13.

Specjalne problemy programowania liniowego

Kroki 3, 4, 5:

c_{ij}	zadania			
wykonawcy	0	6	0	3
	13	0	5	4
	4	3	0	0
	0	9	2	3

c_{ij}	zadania			
wykonawcy	~~0~~	6	△0	3
	13	△0	5	4
	4	3	~~0~~	△0
	△0	9	2	3

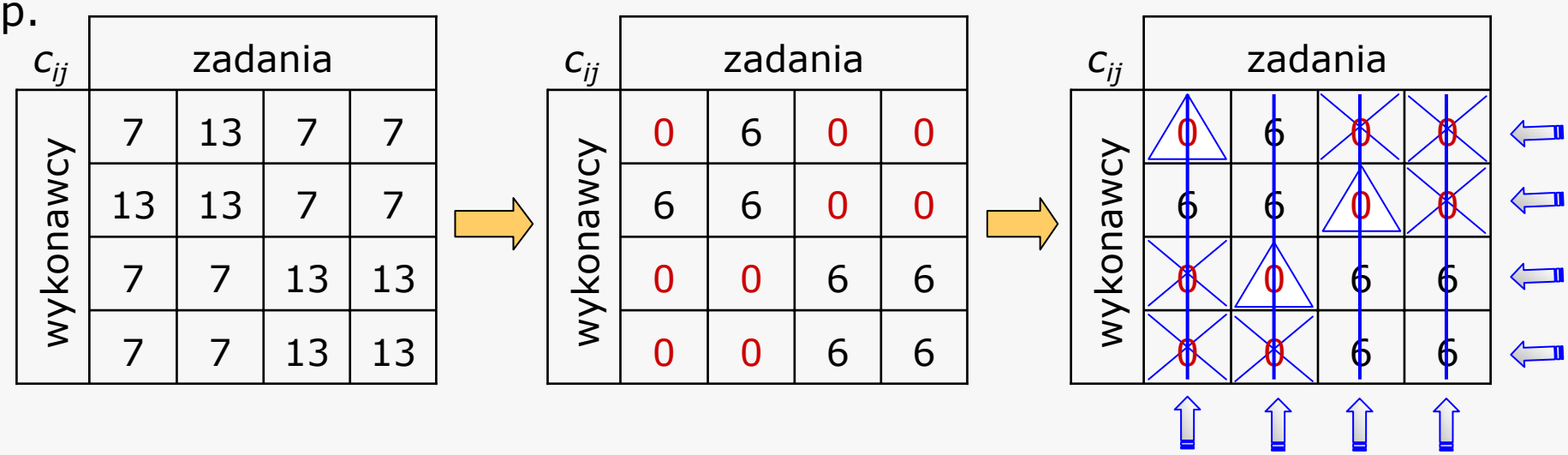
Optymalny przydział:

wykonawca	⇔	zadanie	koszt
w1	⇔	z3	9
			+
w2	⇔	z2	4
			+
w3	⇔	z4	11
			+
w4	⇔	z1	4
			28

Specjalne problemy programowania liniowego

13. Skonstruuj graf skierowany o $2m+2$ wierzchołkach:
s, $w_1,\ldots,w_n$, $z_1,\ldots,z_n$, t. Dla każdego naznaczonego zera (Δ) o współrzędnych (i,j) utwórz łuk $z_j \to w_i$; dla zer skreślonych ($\times$) o współrzędnych (i,j) – łuk $w_i \to z_j$; dla wierszy i bez przydziału – łuk $s \to w_i$; dla kolumn j bez przydziału – łuk $z_j \to t$.

np.

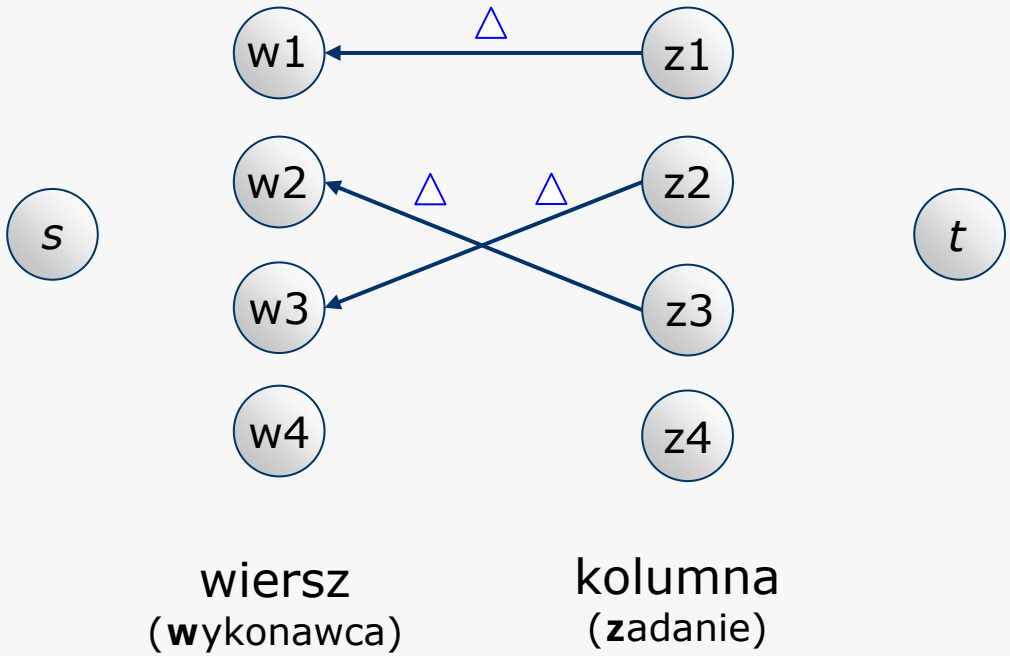


Konieczne są 4 linie do pokrycia wszystkich zer, czyli istnieją 4 zera niezależne, a przydziału dokonano tylko do 3 zer.

Specjalne problemy programowania liniowego

13. Skonstruuj graf skierowany o $2m+2$ wierzchołkach:
s, $w_1,\ldots,w_n$, $z_1,\ldots,z_n$, t. Dla każdego naznaczonego zera (Δ) o współrzędnych (i,j) utwórz łuk $z_j \to w_i$; dla zer skreślonych ($\times$) o współrzędnych (i,j) – łuk $w_i \to z_j$; dla wierszy i bez przydziału – łuk $s \to w_i$; dla kolumn j bez przydziału – łuk $z_j \to t$.

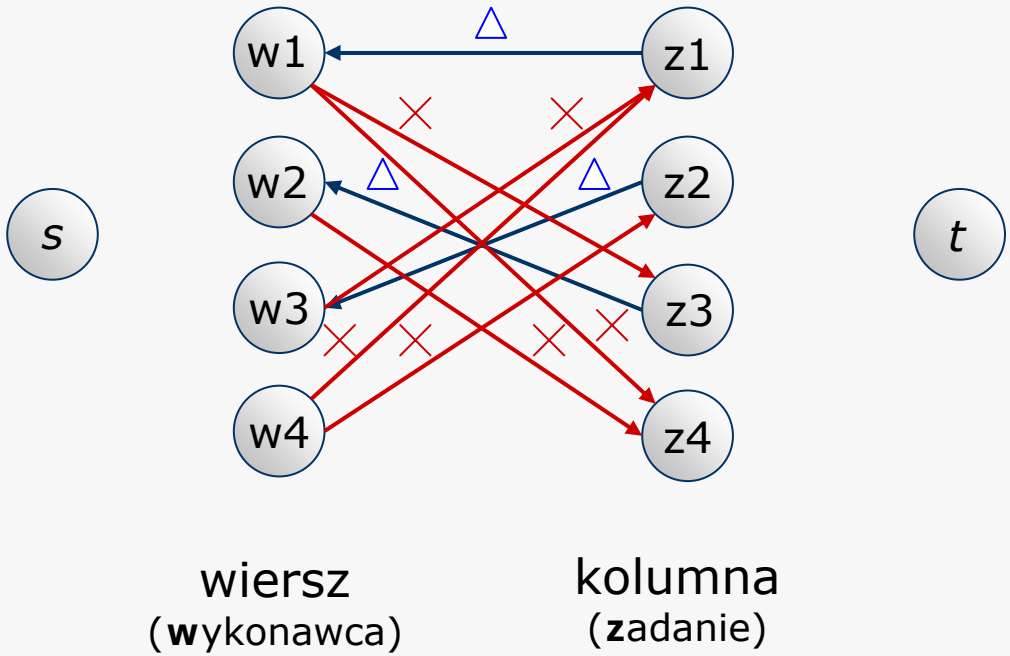
c_{ij}	zadania			
wykonawcy	0	6	0	0
	6	6	0	0
	0	0	6	6
	0	0	6	6



Specjalne problemy programowania liniowego

13. Skonstruuj graf skierowany o $2m+2$ wierzchołkach:
s, $w_1,\ldots,w_n$, $z_1,\ldots,z_n$, t. Dla każdego naznaczonego zera (Δ) o współrzędnych (i,j) utwórz łuk $z_j \to w_i$; dla zer skreślonych ($\times$) o współrzędnych (i,j) – łuk $w_i \to z_j$; dla wierszy i bez przydziału – łuk $s \to w_i$; dla kolumn j bez przydziału – łuk $z_j \to t$.

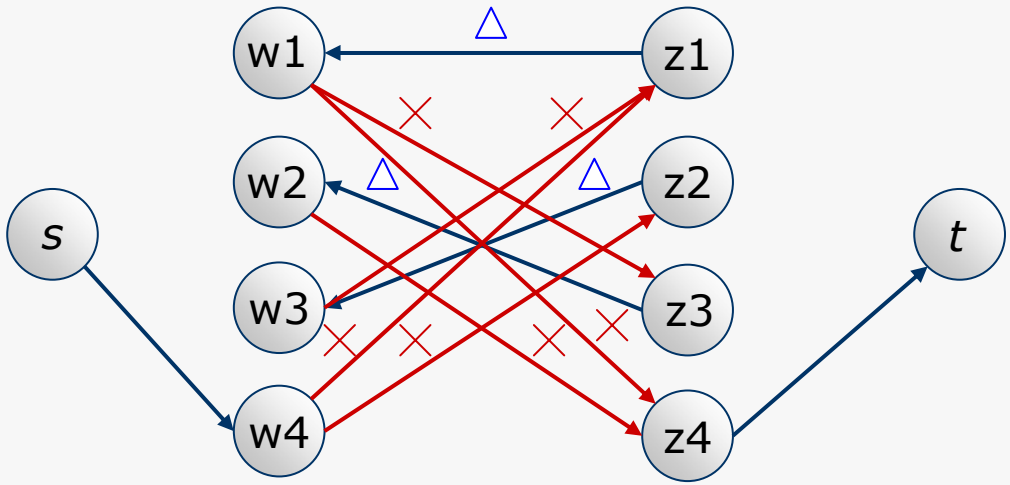
c_{ij}	zadania			
wykonawcy	0	6	0	0
	6	6	0	0
	0	0	6	6
	0	0	6	6



Specjalne problemy programowania liniowego

13. Skonstruuj graf skierowany o $2m+2$ wierzchołkach:
s, $w_1,\ldots,w_n$, $z_1,\ldots,z_n$, t. Dla każdego naznaczonego zera (Δ) o współrzędnych (i,j) utwórz łuk $z_j \to w_i$; dla zer skreślonych ($\times$) o współrzędnych (i,j) – łuk $w_i \to z_j$; dla wierszy i bez przydziału – łuk $s \to w_i$; dla kolumn j bez przydziału – łuk $z_j \to t$.

c_{ij}	zadania			
wykonawcy	0	6	0	0
	6	6	0	0
	0	0	6	6
	0	0	6	6



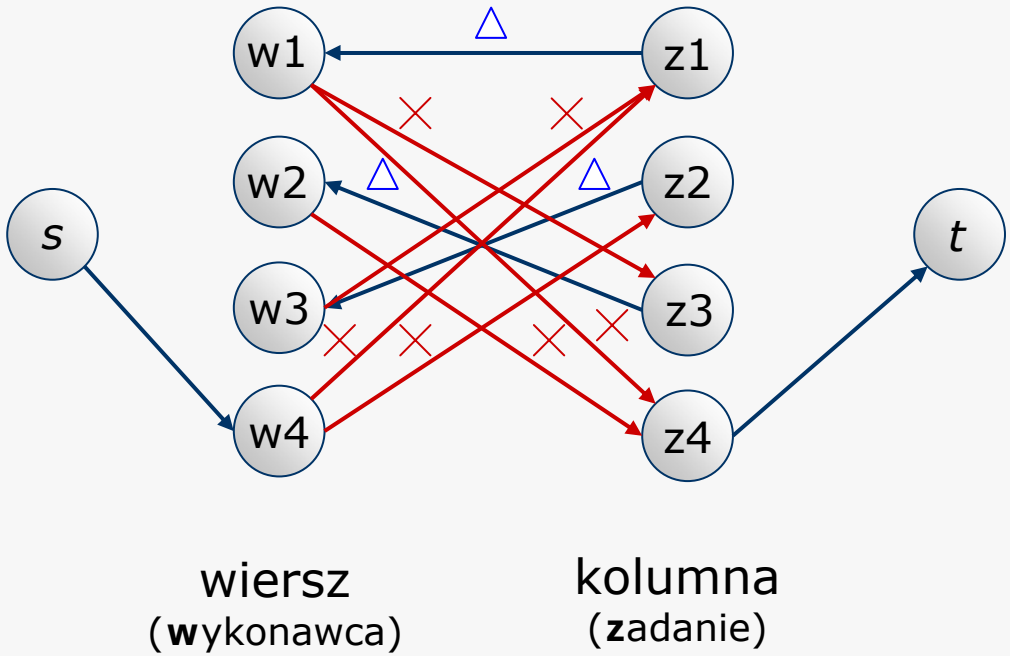
wiersz
(**w**ykonawca)

kolumna
(**z**adanie)

Specjalne problemy programowania liniowego

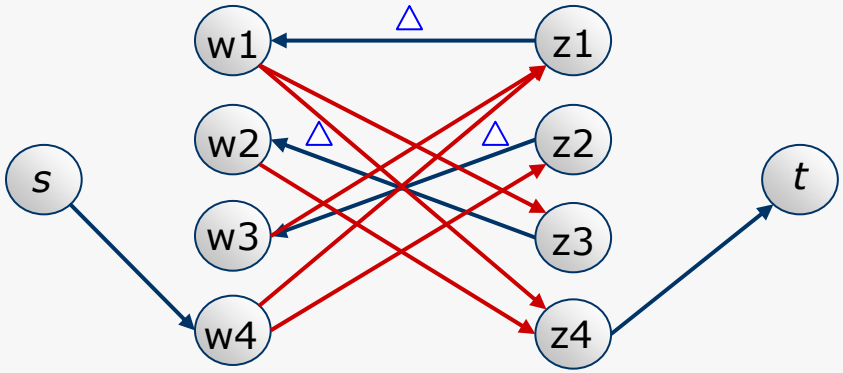
14. Z otrzymanego grafu utwórz sieć warstwową: do pierwszej warstwy wstaw wierzchołek *s*, do warstwy *i*+1 wstaw każdy wierzchołek, którego nie ma w warstwie wcześniejszej, i do którego dochodzi łuk z dowolnego wierzchołka warstwy *i*. W ostatniej warstwie znajdzie się wierzchołek *t*.

c_{ij}	zadania			
wykonawcy	0	6	0	0
	6	6	0	0
	0	0	6	6
	0	0	6	6

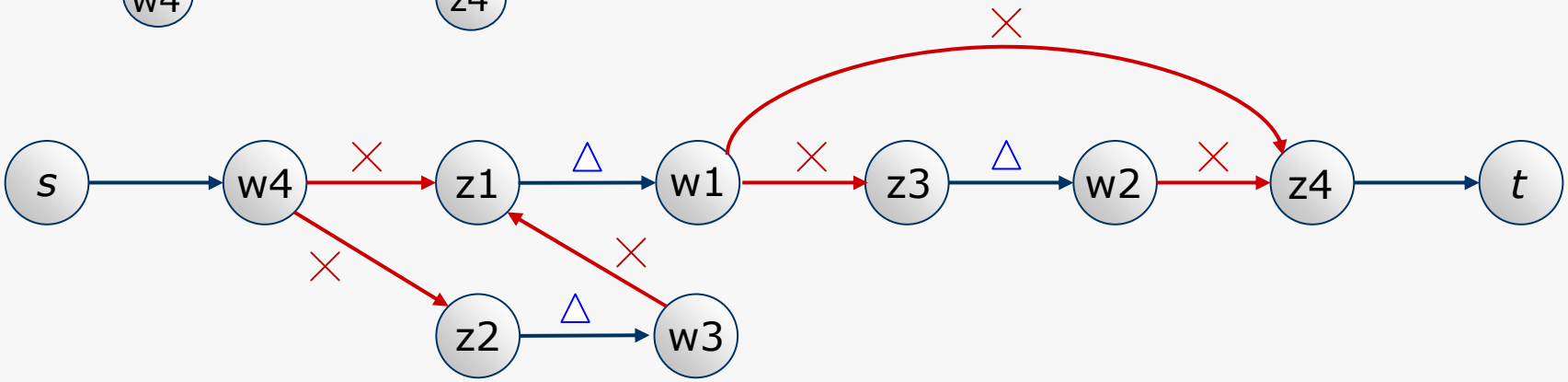


Specjalne problemy programowania liniowego

14. Z otrzymanego grafu utwórz sieć warstwową: do pierwszej warstwy wstaw wierzchołek *s*, do warstwy *i*+1 wstaw każdy wierzchołek, którego nie ma w warstwie wcześniejszej, i do którego dochodzi łuk z dowolnego wierzchołka warstwy *i*. W ostatniej warstwie znajdzie się wierzchołek *t*.



sieć warstwowa

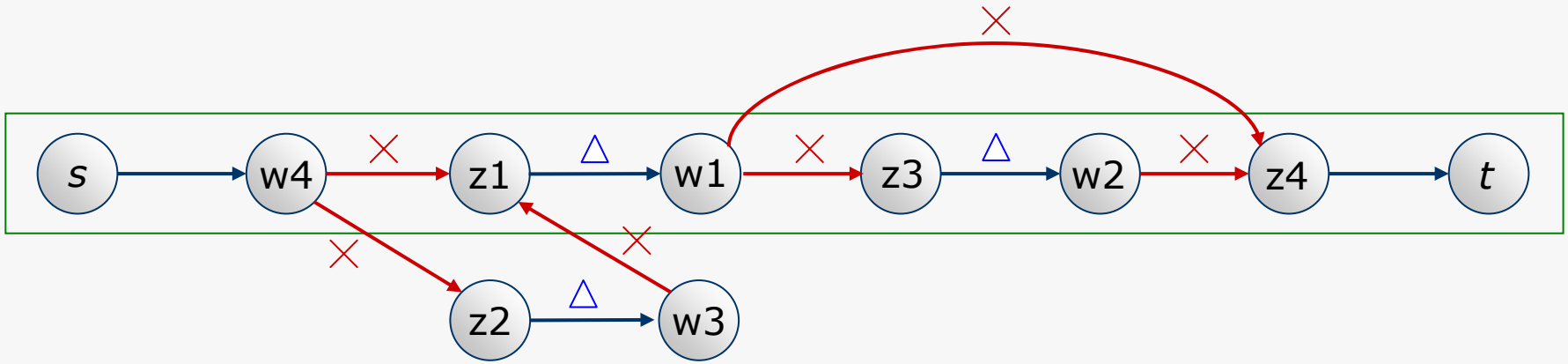


Specjalne problemy programowania liniowego

15. Przesuwając się po dowolnej ścieżce od t do s utwórz tzw. ścieżkę powiększającą przepływ. Naznacz (Δ) zera skreślone o współrzędnych (i,j), odpowiadające łukowi $w_i \rightarrow z_j$ na tej ścieżce; cofnij ($\times$) przydział zer naznaczonych o współrzędnych (i,j), odpowiadających łukowi $z_j \rightarrow w_i$ na tej ścieżce.

W poniższej sieci są 3 ścieżki powiększające przepływ

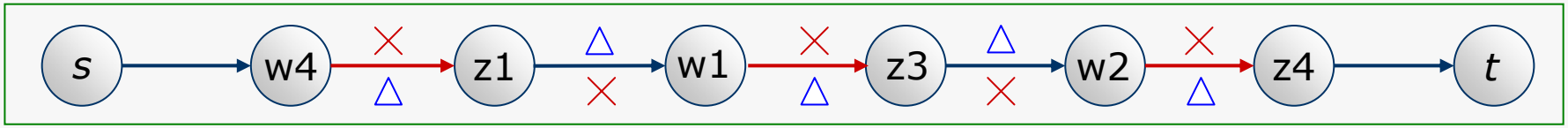
sieć warstwowa



Specjalne problemy programowania liniowego

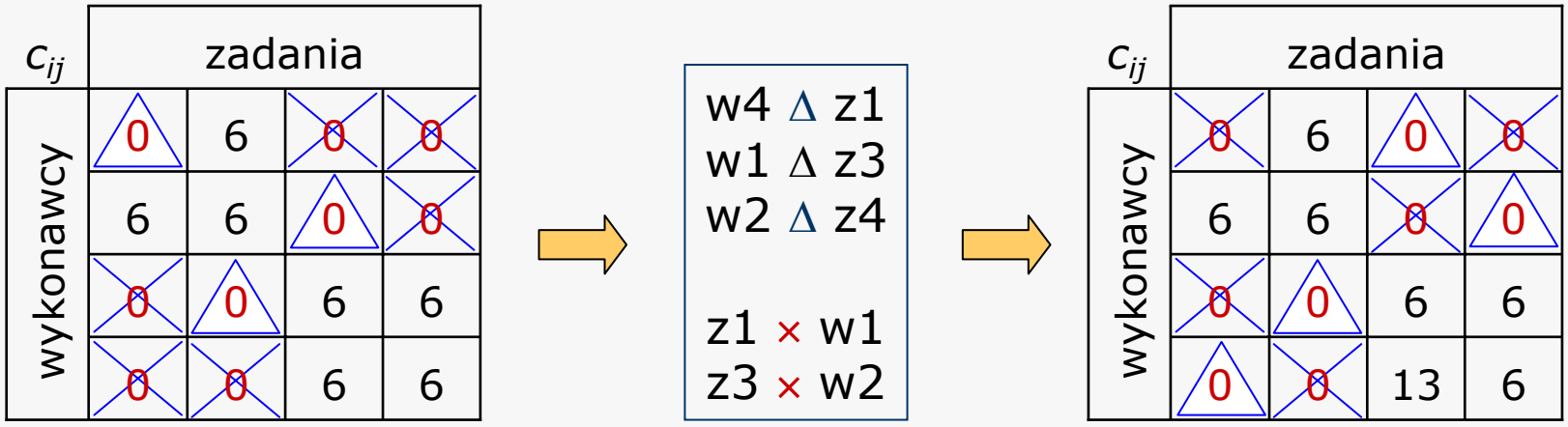
15. Przesuwając się po dowolnej ścieżce od t do s utwórz tzw. ścieżkę powiększającą przepływ. Naznacz (Δ) zera skreślone o współrzędnych (i,j), odpowiadające łukowi $w_i \rightarrow z_j$ na tej ścieżce; cofnij ($\times$) przydział zer naznaczonych o współrzędnych (i,j), odpowiadających łukowi $z_j \rightarrow w_i$ na tej ścieżce.

ścieżka powiększająca przepływ



Specjalne problemy programowania liniowego

15. Przesuwając się po dowolnej ścieżce od *t* do *s* utwórz tzw. ścieżkę powiększającą przepływ. Naznacz (Δ) zera skreślone o współrzędnych (*i*,*j*), odpowiadające łukowi $w_i \rightarrow z_j$ na tej ścieżce; cofnij (×) przydział zer naznaczonych o współrzędnych (*i*,*j*), odpowiadających łukowi $z_j \rightarrow w_i$ na tej ścieżce.
16. Jeśli znaleziono przydział do *m* zer, to jest on optymalny → STOP.
17. Wróć do kroku 7.



Przydział optymalny

Specjalne problemy programowania liniowego

- Uwagi uzupełniające:
  - Ten sam algorytm można zastosować dla problemu maksymalizacji funkcji celu (zysku), jeśli zamieni się znaki elementów macierzy $C=[c_{ij}]$ na przeciwne.
  - Jeśli macierz $C=[c_{ij}]$ nie jest kwadratowa, to można ją uzupełnić elementami zerowymi w kolumnach lub wierszach, tak aby uzyskać macierz kwadratową.
  - Jeśli przydział wykonawcy *i* do zadania *j* jest zakazany, to przyjmujemy $c_{ij} = \infty$.