

Badania Operacyjne

dr inż. Krzysztof Pieńkosz

Instytut Automatyki i Informatyki Stosowanej
Politechniki Warszawskiej

pok. 560^A

tel.: 660-78-64

e-mail: K.Pienkosz@ia.pw.edu.pl

Badania Operacyjne

Literatura uzupełniająca

- Sysło M. M., Deo N., Kowalik J.S.: *Algorytmy optymalizacji dyskretnej*
- Siudak M.: *Badania operacyjne*
- Wagner H. M.: *Badania operacyjne*
- Walukiewicz S.: *Programowanie dyskretne*
- Błażewicz J., Cellary W., Słowiński R., Węglarz J.: *Badania operacyjne dla informatyków*
- Jędrzejczyk Z., Kukuła K., Skrzypek J., Walkosz A.: *Badania operacyjne w przykładach i zadaniach*

Badania Operacyjne

Badanie
Modelowanie
Analiza
Rozwiązywanie

Problemów decyzyjnych
(dyskretnych, dyskretno-ciągłych)

przy uwarunkowaniach
(ograniczeniach zasobowych, czasowych,
relacjach poprzedzania, itp.)

w celu
spełnienia zadanych kryteriów decyzyjnych

Badania Operacyjne

Wybrane dziedziny zastosowań

- Planowanie przedsięwzięć
- Zagadnienia dystrybucyjne i transportowe (np. planowanie tras dostaw)
- Szeregowanie i harmonogramowanie zadań
- Układanie rozkładów zajęć, pociągów, itp.
- Planowanie i zarządzanie produkcją (zapasami)
- Zarządzanie systemami masowej obsługi
- Problemy rozkroju i pakowania
- Projektowanie lokalizacji, rozmieszczenia i powiązania obiektów (np. w sieci)

Przykład zastosowań

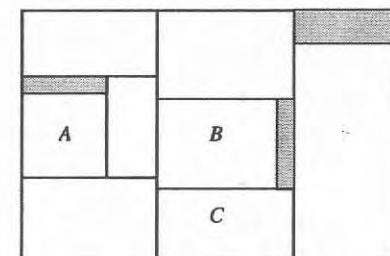
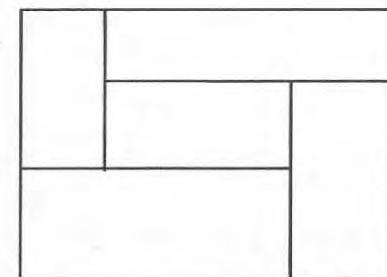
Komputerowy system optymalizacji rozkroju szklanych tafli

Zamówienie: Zakład Szyb Zespolonych

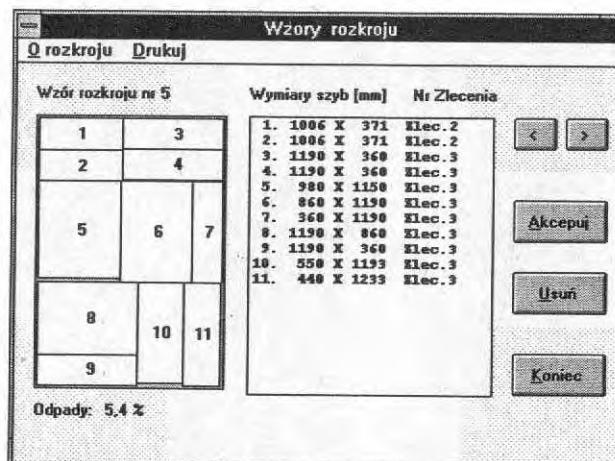
Wymagania:

- Tafla wyjściowa 3210 mm na 2250 mm
- 4 rodzaje grubości szkła
- Zlecenia na szyby mają charakter jednostkowy – bardzo różne wymiary szyb (od 100 mm na 300 mm) w małych ilościach (najczęściej 2 szyby)
- Szyby prostokątne
- Dopuszczalne tylko cięcia gilotynowe
- Używana maszyna umożliwia wykonanie tylko 3 stopni cięć
- Marginesy na brzegach tafli i przy „łamaniu” szyb

Przykłady niedopuszczalnych wzorów rozkroju



Otrzymywane wzory rozkroju



Badania Operacyjne

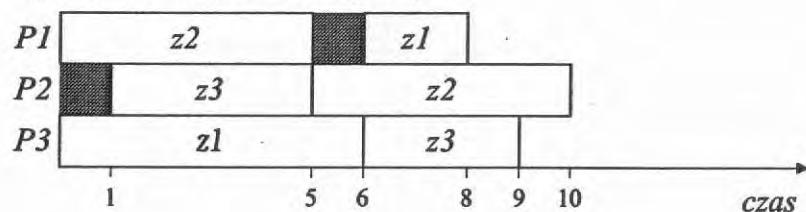
Podstawowe pojęcia

- **model problemu (procesu)** - reprezentacja (opis) problemu (zwykle uproszczony) w pewnym zapisie matematycznym.
 - **model optymalizacyjny** - zawiera kryteria określające jakość podejmowanych decyzji
- **proces** - przebieg kolejnych zmian stanu systemu
 - proces ciągły - stan zmienia się w sposób ciągły
 - proces dyskretny - stan zmienia się w sposób nieciągły (skokowy)
- **operacja** - elementarna czynność w procesie (dyskretnym) wykonywana z wykorzystaniem ustalonego zestawu zasobów. Atrybuty operacji:
 - czas trwania
 - używane zasoby
- **zdarzenie** - chwila wystąpienia zmiany stanu w systemie.

- **zasób** - pewna ilość środków niezbędnych do realizacji planowanych zadań (operacji)
 - zasoby zużywalne** - zużywane w trakcie wykonywania operacji, np. energia, pieniądze, czas, itd.
 - zasoby odnawialne** (wykorzystywane chwilowo)
 - udostępniane z powrotem po zakończeniu wykonania operacji, np. maszyna, procesor, pamięć komputerowa, kanał transmisyjny, itd.

- **Optymalizacja** - wybór najlepszych, w sensie ustalonych kryteriów oceny, wariantów ze zbioru rozwiązań dopuszczalnych.
- **Logistyka** - ogół działań służących zabezpieczeniu dostępności wymaganych zasobów w celu niezawodnej i efektywnej realizacji planowanych procesów.
- **Szeregowanie zadań** - wyznaczanie kolejności realizacji zadań (operacji).
- **Harmonogramowanie** - wyznaczanie najlepszych sposobów realizacji operacji w czasie i rozdziału zasobów wymaganych przez te operacje.

Wykres Gantta (Harmonogram)



- **Symulacja** - określanie przebiegu realizacji procesu na podstawie modelu.

Badania Operacyjne

Ogólna metodyka postępowania

1. opis (identyfikacja) problemu
2. tworzenie modelu
3. wybranie lub opracowanie metody rozwiązywania (algorytmu)
4. analiza rozwiązań (pod względem poprawności, dokładności, szybkości obliczeń, itd.)
5. ewentualna modyfikacja modelu lub algorytmu
6. wdrożenie

Badania Operacyjne

Podstawowe modele i metody

- metody optymalizacji (dyskretnej)
 - modele i algorytmy grafowe
 - modele przepływów w sieciach
 - programowanie liniowe i całkowitoliczbowe
 - programowanie dynamiczne
 - heurystyki
- metody sztucznej inteligencji
 - metody przeszukiwania
 - metaheurystyki
 - metody wnioskowania
- modele i metody symulacyjne
- decyzyjne łańcuchy Markowa
- systemy masowej obsługi i sieci kolejkowe
- modele teorii gier

Przykład modeli grafowych

$G(V, E)$ - graf nieskierowany

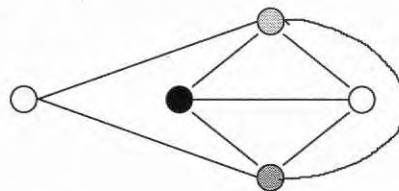
V - zbiór wierzchołków, E - zbiór krawędzi

$$n = |V|, \quad m = |E|$$

• Problem kolorowania wierzchołków w grafie

Należy pokolorować wierzchołki grafu minimalną liczbą kolorów, tak aby wierzchołki sąsiednie miały różne kolory.

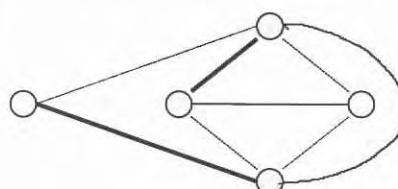
$\chi(G)$ - liczba chromatyczna grafu



• Problem kolorowania krawędzi grafu (multigrafu)

Należy pokolorować krawędzie grafu (multigrafu) minimalną liczbą kolorów, tak aby krawędzie incydentne miały różne kolory.

$I(G)$ - indeks chromatyczny grafu



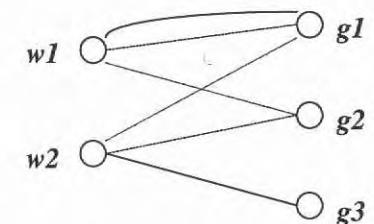
Zastosowania modeli kolorowania grafów

• Układanie planu zajęć

$\{w1, w2\}$ - wykładowcy

$\{g1, g2, g3\}$ - grupy studenckie

	$g1$	$g2$	$g3$
$w1$	2	1	
$w2$	1	1	1

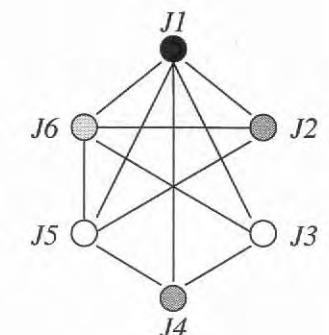


• Szeregowanie zadań

$\{J1, J2, J3, J4, J5, J6\}$ - zadania

$\{r1, r2, r3, r4\}$ - zasoby

	$J1$	$J2$	$J3$	$J4$	$J5$	$J6$
$r1$	*			*	*	
$r2$	*		*			*
$r3$			*	*		
$r4$	*	*			*	*



Parametry modeli

- deterministyczne
- niepewne
 - losowe (podejście stochastyczne)
 - w przedziałach niepewności
 - rozmyte

Planowanie przedsięwzięć

Model przedsięwzięcia

- lista operacji
- relacje poprzedzania operacji
- modele operacji
- funkcja celu planowania

Relacje poprzedzania operacji ($A \rightarrow B$)

- Koniec A - Start B – operacja B nie może się rozpocząć przed zakończeniem A
- Koniec A - Koniec B – operacja B nie może się zakończyć przed zakończeniem A
- Start A - Start B – operacja B nie może się rozpoczęć przed rozpoczęciem A
- Start A - Koniec B – operacja B nie może się zakończyć przed rozpoczęciem A

Modele operacji

- czasy trwania
 - deterministyczne
 - deterministyczne zależne od wielkości przydzielonego zasobu
 - losowe
- wymagania zasobowe

Funkcje celu planowania

- minimalizacja czasu realizacji przedsięwzięcia (przy zadanych zasobach)
- minimalizacja wykorzystania zasobów (przy zadanym czasie realizacji przedsięwzięcia)

Przykład

Projekt implementacji ośmiu modułów systemu informatycznego

- lista operacji
M1, M2, M3, M4, M5, M6, M7, M8
gdzie M_i , $i=1,\dots,8$ – oznacza operację implementacji modułu i
- relacje poprzedzania i czasy trwania operacji

operacja	operacje poprzedzające	czas trwania [tygodnie]
M1	–	13
M2	–	8
M3	M1	9
M4	M1	15
M5	M2	20
M6	M2, M3	3
M7	M2, M3	4
M8	M4, M6	10

Metoda ścieżki krytycznej CPM (Critical Path Method)

Model przedsięwzięcia

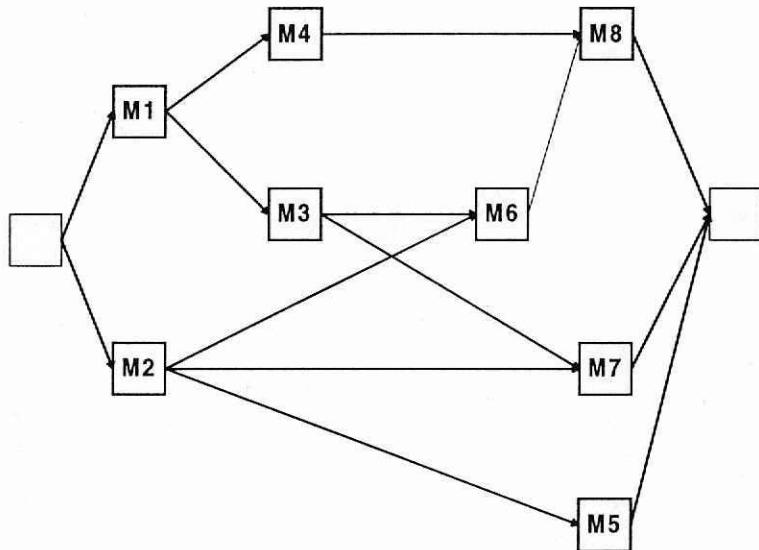
- relacje poprzedzania: Koniec-Start
- model operacji
 - czasy trwania: deterministyczne
 - brak wymagań zasobowych
- funkcja celu: minimalizacja czasu realizacji przedsięwzięcia

Schemat metody

- 1.Konstrukcja sieci przedsięwzięcia
- 2.Sortowanie topologiczne wierzchołków sieci
- 3.Wyznaczenie najwcześniejszych terminów (chwil) zdarzeń i ścieżki krytycznej
- 4.Wyznaczenie najpóźniejszych terminów (chwil) zdarzeń i luzów czasowych operacji

Sieć przedsięwzięcia (w reprezentacji wierzchołkowej – diagram PERT)

wierzchołki sieci: operacje
łuki sieci: relacje poprzedzania

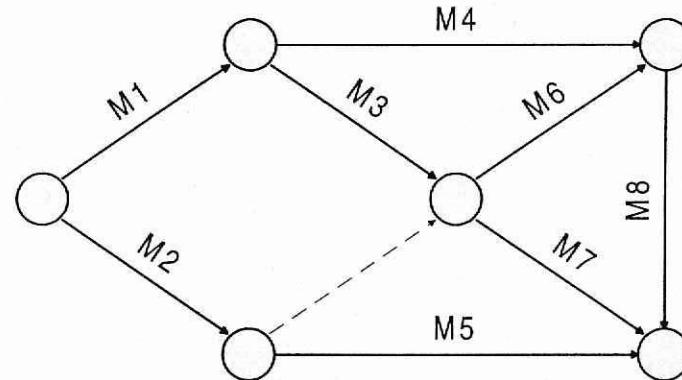


Właściwości

- Graf acykliczny

Sieć przedsięwzięcia (w reprezentacji łukowej)

wierzchołki sieci: zdarzenia
łuki sieci: operacje



Właściwości

- Graf acykliczny
- mogą występować sztuczne (fikcyjne) łuki

Sortowanie topologiczne

Wierzchołki grafu $G(V,E)$ są ponumerowane topologicznie, jeżeli $(i,j) \in E \Leftrightarrow num(i) < num(j)$

Algorytm sortowania (numerowania)

Założenia:

- na początku wszystkie wierzchołki są bez numeru
- $n \geq 1$

1. $num := 1$;
2. Znajdź wierzchołek bez numeru do którego nie dochodzi żaden łuk. Jeżeli taki wierzchołek istnieje, to przypisz mu numer num i idź do 3. W przeciwnym przypadku STOP – w grafie jest cykl;
3. Jeżeli wszystkie wierzchołki są ponumerowane to STOP. W przeciwnym przypadku usuń łuki wychodzące z ponumerowanego wierzchołka, $num := num + 1$ i idź do 2.

Najwcześniejsze terminy zdarzeń

Oznaczenia:

V – zbiór wierzchołków sieci przedsięwzięcia
(posortowanych topologicznie)

$$n = |V|$$

E – zbiór łuków (operacji)

p_{ij} – czas trwania operacji $(i,j) \in E$

s_i – najwcześniejszy możliwy termin wystąpienia zdarzenia $i \in V$ (wszystkie poprzedzające operacje muszą być zakończone)

Aby operacje poprzedzające "zdążyły" się wykonać musi zachodzić

$$s_i + p_{ij} \leq s_j \text{ dla każdej operacji } (i,j) \in E$$

Wnioski

- s_i = długość najdłuższej ścieżki z wierzchołka 1 do i
- minimalny czas realizacji przedsięwzięcia określa s_n czyli najdłuższa ścieżka z wierzchołka 1 do n (ścieżka krytyczna).
- dla wszystkich operacji (i,j) na ścieżce krytycznej zachodzi

$$s_i + p_{ij} = s_j$$

Wyznaczanie najwcześniejszych terminów zdarzeń

(znajdowanie najdłuższej ścieżki w grafie posortowanym topologicznie – złożoność $O(|E|)$)

$$s_1 := 0;$$

for $j := 2$ to n do

$$s_j := \max_{i:(i,j) \in E} \{s_i + p_{ij}\}$$

Wyznaczanie minimalnego czasu realizacji przedsięwzięcia

- $T = s_n$ (najdłuższa ścieżka od wierzchołka 1 do n)
- Model programowania liniowego

$$\min T = t_n$$

przy ograniczeniach

$$t_i + p_{ij} \leq t_j \quad (i, j) \in E$$

$$t_i \geq 0 \quad i \in V$$

gdzie t_i – zmienna określająca termin zdarzenia $i \in V$

Na ścieżce krytycznej $t_i = s_i$ (w szczególności $t_n = s_n$)

Najpóźniejsze terminy zdarzeń

Oznaczenia:

l_i – najpóźniejszy dopuszczalny termin zdarzenia $i \in V$ (przy założeniu, że przedsięwzięcie jest realizowane w najkrótszym możliwym czasie tzn., $l_n = s_n$)

Aby operacje następne "zdążyły" się wykonać bez wydłużania terminu realizacji przedsięwzięcia $l_n = s_n$ musi zachodzić

$$l_i + p_{ij} \leq l_j \text{ dla każdej operacji } (i, j) \in E$$

Wnioski

- $l_i = s_n$ – (długość najdłuższej ścieżki z i do n)
- $l_i \geq s_i$ dla każdego wierzchołka $i \in V$
- dla wszystkich operacji (i, j) na ścieżce krytycznej zachodzi

$$l_i + p_{ij} = l_j$$

- dla wszystkich wierzchołków i na ścieżce krytycznej zachodzi

$$l_i = s_i$$

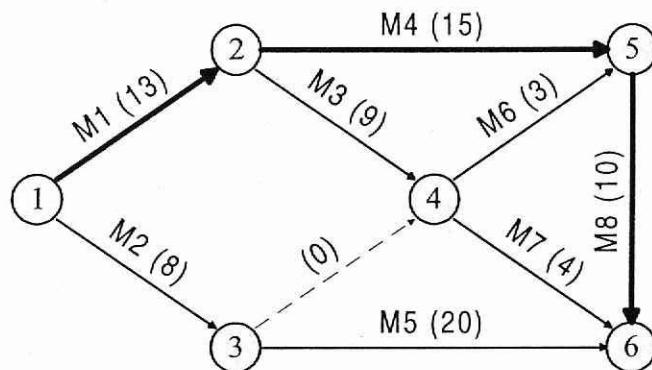
Wyznaczanie najpóźniejszych terminów zdarzeń

$$l_n := s_n;$$

for $i := n - 1$ to 1

$$l_i := \min_{j:(i,j) \in E} \{l_j - p_{ij}\}$$

Przykład



Terminy zdarzeń

i (nr wierzchołka - zdarzenia)

	1	2	3	4	5	6
s_i	0	13	8	22	28	38
l_i	0	13	18	25	28	38

s_i – termin najwcześniejszy możliwy

l_i – termin najpóźniejszy dopuszczalny

Najwcześniejsze i najpóźniejsze terminy rozpoczęcia i zakończenia operacji

- Najwcześniejszy możliwy termin rozpoczęcia operacji (i, j)
 $NWR(i, j) = s_i$
- Najpóźniejszy dopuszczalny termin rozpoczęcia operacji (i, j)
 $NPR(i, j) = l_j - p_{ij}$
- Najwcześniejszy możliwy termin zakończenia operacji (i, j)
 $NWZ(i, j) = s_i + p_{ij}$
- Najpóźniejszy dopuszczalny termin zakończenia operacji (i, j)
 $NPZ(i, j) = l_j$

Terminy rozpoczęcia i zakończenia operacji

operacja	M1	M2	M3	M4	M5	M6	M7	M8
NWR	0	0	13	13	8	22	22	28
NPR	0	10	16	13	18	25	34	28
NWZ	13	8	22	28	28	25	26	38
NPZ	13	18	25	28	38	28	38	38

Luzy (zapasy) czasowe operacji

- Luz (zapas) całkowity – maksymalne opóźnienie operacji nie powodujące opóźnienia przedsięwzięcia

$$LC(i, j) = l_j - s_i - p_{ij}$$

$$LC(i, j) = NPR(i, j) - NWR(i, j) = NPZ(i, j) - NWZ(i, j)$$

- Luz (zapas) swobodny – maksymalne opóźnienie operacji nie wpływające na czas rozpoczęcia następnych operacji

$$LS(i, j) = s_j - s_i - p_{ij}$$

Luzy

operacja	M1	M2	M3	M4	M5	M6	M7	M8
LC	0	10	3	0	10	3	12	0
LS	0	0	0	0	10	3	12	0

Na ścieżce krytycznej luzy zerowe

Metoda PERT

(Program Evaluation and Review Technique)

Model przedsięwzięcia

- relacje poprzedzania: Koniec-Start
- model operacji
 - brak wymagań zasobowych
 - czasy trwania: zmienna losowa (rozkład beta) parametry:
 - a – ocena optymistyczna
 - m – czas najbardziej prawdopodobny
 - b – ocena pesymistyczna

Wzory aproksymujące

$$E(p) = \frac{a + 4m + b}{6}, \quad \sigma(p) = \frac{b - a}{6}$$

Schemat metody

- 1.Konstrukcja sieci przedsięwzięcia
- 2.Sortowanie topologiczne wierzchołków sieci
- 3.Wyznaczenie ścieżki krytycznej dla wartości oczekiwanych czasów operacji
- 4.Analiza czasu realizacji przedsięwzięcia w oparciu o rozkład normalny

Czas trwania przedsięwzięcia w metodzie PERT

Zmienna losowa:

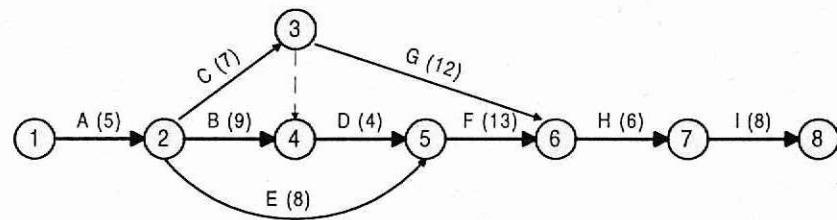
$$T = \sum_{(i,j) \in Kryt} p_{ij}$$

Suma dużej liczby niezależnych zmiennych losowych o jednakowym rozkładzie ma zgodnie z centralnym twierdzeniem granicznym rozkład zbliżony do normalnego

$$E(T) = \sum_{(i,j) \in Kryt} E(p_{ij}) \quad \sigma^2(T) = \sum_{(i,j) \in Kryt} \sigma^2(p_{ij})$$

Metoda PERT – przykład

operacja	poprzed.	a	m	b	$E(p_{ij})$	$\sigma(p_{ij})$	$\sigma^2(p_{ij})$
A	-	2	5	8	5	1	1
B	A	6	9	12	9	1	1
C	A	6	7	8	7	1/3	1/9
D	B,C	1	4	7	4	1	1
E	A	8	8	8	8	0	0
F	D,E	5	14	17	13	2	4
G	C	3	12	21	12	3	9
H	F,G	3	6	9	6	1	1
I	H	5	8	11	8	1	1



Ścieżka krytyczna: A – B – D – F – H – I

$$E(T) = 5 + 9 + 4 + 13 + 6 + 8 = 45$$

$$\sigma^2(T) = 1 + 1 + 1 + 4 + 1 + 1 = 9$$

$$\sigma(T) = \sqrt{\sigma^2(T)} = 3$$

$$P(T \leq 50) = P(Z \leq \frac{50 - 45}{3}) = P(Z \leq 1.67) = 0.95$$

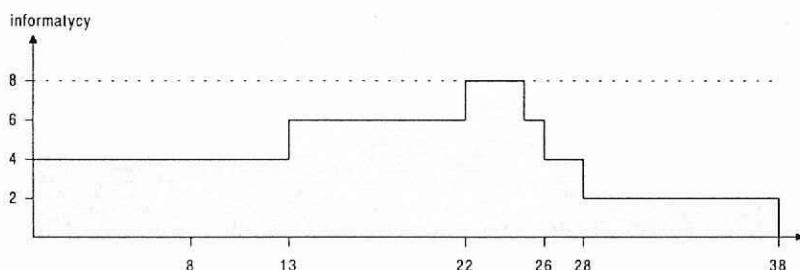
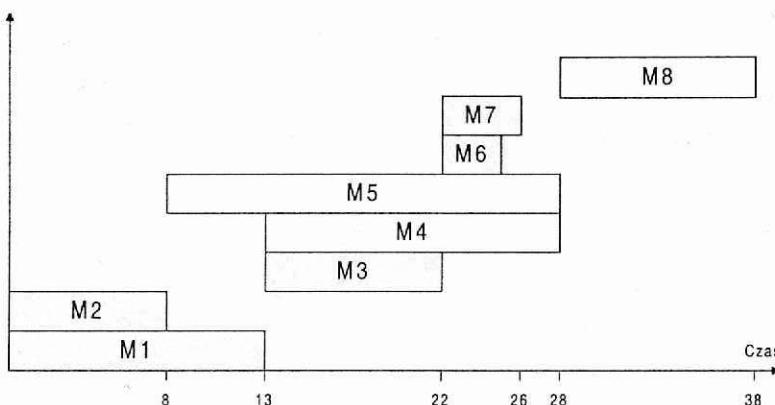
$$P(T \leq 41) = P(Z \leq \frac{41 - 45}{3}) = P(Z \leq -1.33) = 0.09$$

Ograniczenia zasobowe

Przykład – przydział zasobów odnawialnych

Każda z operacji M1, M2, ..., M8 wymaga pracy dwóch informatyków.

Harmonogram i zużycie zasobów przy najwcześniejszych terminach rozpoczęcia operacji



Kryteria planowania

- minimalizacja liczby zatrudnianych pracowników (przy zadanym czasie realizacji przedsięwzięcia)

czas realizacji	38-44	45-81	≥ 82
pracownicy	6	4	2

- minimalizacja czasu realizacji przedsięwzięcia (przy zadanej liczbie pracowników)

pracownicy	< 2	2-3	4-5	≥ 6
czas realizacji	–	82	45	38

- analiza dwukryterialna
rozwiązania niezdominowane (Pareto-optymalne)

czas realizacji	38	45	82
pracownicy	6	4	2

Ograniczenia zasobowe .

Przydział zasobów zużywalnych

Czas wykonania poszczególnych operacji można skrócić jeżeli przydzieli się jej pewną ilość zasobu (np. środków pieniężnych)

Zalożenia

p_{ij}^{max} – nominalny (maksymalny) czas trwania operacji (i, j) bez przydziału zasobu

k_{ij} – koszt skrócenia czasu wykonywania operacji (i, j) o jednostkę czasu (współczynnik zużycia zasobu)

p_{ij}^{min} – minimalny czas trwania operacji, którego nie można dalej skrócić poprzez przydział zasobu

Minimalizacja czasu trwania przedsięwzięcia przy zadanej (ograniczonej) wielkości zasobu

Z – wielkość zasobu (ilość dostępnych środków)

Podejścia

- skracanie czasu trwania przedsięwzięcia poprzez przydział kolejnych jednostek zasobu do najmniej „kosztownych” operacji na ścieżce krytycznej
- model Programowania Liniowego

$$\min T = t_n$$

przy ograniczeniach

$$t_i + p_{ij} \leq t_j \quad (i, j) \in E$$

$$p_{ij} = p_{ij}^{max} - (1/k_{ij}) z_{ij} \quad (i, j) \in E$$

$$\sum_{(i,j) \in E} z_{ij} \leq Z$$

$$t_i \geq 0 \quad i \in V$$

$$p_{ij} \geq p_{ij}^{min} \quad (i, j) \in E$$

$$z_{ij} \geq 0 \quad (i, j) \in E$$

Zmienne decyzyjne:

t_i – termin zdarzenia $i \in V$

p_{ij} – czas trwania operacji po ewentualnym skróceniu

z_{ij} – wielkość zasobu przydzielona do operacji (i, j)

Minimalizacja zużycia zasobu przy zadanym maksymalnym czasie trwania przedsięwzięcia

T – maksymalny dopuszczalny czas trwania przedsięwzięcia

- model Programowania Liniowego

$$\min Z$$

przy ograniczeniach

$$t_i + p_{ij} \leq t_j \quad (i, j) \in E$$

$$t_n \leq T$$

$$p_{ij} = p_{ij}^{\max} - (1/k_{ij}) z_{ij} \quad (i, j) \in E$$

$$\sum_{(i,j) \in E} z_{ij} = Z$$

$$t_i \geq 0 \quad i \in V$$

$$p_{ij} \geq p_{ij}^{\min} \quad (i, j) \in E$$

$$z_{ij} \geq 0 \quad (i, j) \in E$$

Zmienne decyzyjne: t_i, p_{ij}, z_{ij}, Z

Zadanie Programowania Liniowego (postać standardowa)

Polega na znalezieniu optymalnych wartości zmiennych decyzyjnych $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$ maksymalizujących funkcję celu

$$\max_{\mathbf{x}} x_0 = \sum_{j=1}^n c_j x_j$$

przy spełnieniu ograniczeń

$$\begin{aligned} \sum_{j=1}^n a_{ij} x_j &= b_i \quad i = 1, \dots, m \\ x_j &\geq 0 \quad j = 1, \dots, n \end{aligned}$$

gdzie c_j, a_{ij}, b_i znane współczynniki oraz $m < n$

Zapis wektorowy

$$\begin{aligned} \max_{\mathbf{x}} x_0 &= \mathbf{c}^T \mathbf{x} \\ \mathbf{A}\mathbf{x} &= \mathbf{b} \\ \mathbf{x} &\geq 0 \end{aligned}$$

gdzie

- $\mathbf{c} = (c_1, c_2, \dots, c_n)^T$
- $\mathbf{b} = (b_1, b_2, \dots, b_m)^T$
- $\mathbf{A} = [a_{ij}]$ – macierz ograniczeń $m \times n$

Inne równoważne postaci zadania PL

- „min” zamiast „max” w funkcji celu
- znaki „ \geq ” lub „ \leq ” zamiast „ $=$ ” w ograniczeniach
- zmienne decyzyjne x_j nieograniczone lub ograniczone dowolnymi wartościami od dołu i/lub od góry

Reguły równoważnych przekształceń

- zamiana rodzaju ekstremów
 $\min \mathbf{c}^T \mathbf{x} = -(\max -\mathbf{c}^T \mathbf{x})$
- zamiana ograniczeń nierównościowych na równościowe
 wprowadzenie zmiennej dopełniającej x_d
 $\mathbf{a}^T \mathbf{x} \leq b \Rightarrow \mathbf{a}^T \mathbf{x} + x_d = b, \quad x_d \geq 0$
 $\mathbf{a}^T \mathbf{x} \geq b \Rightarrow \mathbf{a}^T \mathbf{x} - x_d = b, \quad x_d \geq 0$
- zamiana ograniczeń równościowych na nierównościowe
 $\mathbf{a}^T \mathbf{x} = b \Rightarrow \mathbf{a}^T \mathbf{x} \leq b \text{ i } \mathbf{a}^T \mathbf{x} \geq b$
- zamiana zmiennych nieograniczonych na zmienne nieujemne
 wprowadzenie par zmiennych x_j^+ , x_j^- i podstawienie
 $x_j := x_j^+ - x_j^-, \quad x_j^+ \geq 0 \text{ i } x_j^- \geq 0$

Przykład

Firma wytwarza dwa rodzaje farb – do malowania wnętrz (W) i na zewnętrz (Z). Do produkcji tych farb niezbędne są dwa podstawowe składniki A i B . Maksymalne dzienne zapasy tych składników wynoszą odpowiednio 6 i 8 ton, natomiast ich zużycie na 1 tonę farby jest następujące:

Składniki	Zużycie składników (w tonach)	
	farba W	farba Z
A	2	1
B	1	2

Badania rynkowe pokazały, że dzienny zbyt na farbę W nigdy nie przekracza 2 ton i nie jest wyższy od zbytu na farbę Z o więcej niż 1 tонę. Cena sprzedaży 1 tony farby W wynosi 20 tys. zł, a farby Z 30 tys. zł.

Należy określić dzienne wielkości produkcji farb W i Z przynoszące największe dochody.

Model Programowania Liniowego

- zmienne decyzyjne

$$x_W - \text{dzienna wielkość produkcji farby } W \text{ (w tonach)}$$

$$x_Z - \text{dzienna wielkość produkcji farby } Z \text{ (w tonach)}$$

- funkcja celu

$$\max x_0 = 20x_W + 30x_Z$$

- ograniczenia

$$2x_W + x_Z \leq 6$$

$$x_W + 2x_Z \leq 8$$

$$x_W - x_Z \leq 1$$

$$x_W \leq 2$$

$$x_W \geq 0, x_Z \geq 0$$

Interpretacja graficzna zadania PL

$$\max x_0 = 20x_W + 30x_Z \quad (1)$$

przy ograniczeniach

$$2x_W + x_Z \leq 6 \quad (2)$$

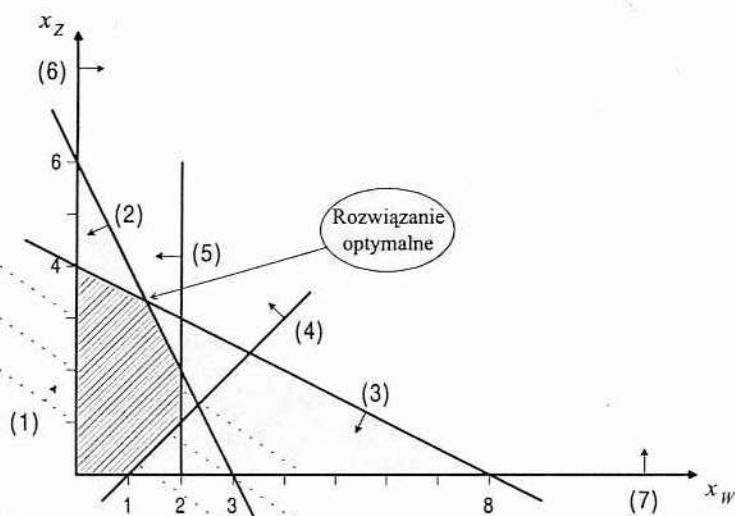
$$x_W + 2x_Z \leq 8 \quad (3)$$

$$x_W - x_Z \leq 1 \quad (4)$$

$$x_W \leq 2 \quad (5)$$

$$x_W \geq 0 \quad (6)$$

$$x_Z \geq 0 \quad (7)$$



Analiza parametryczna

$$\max x_0 = cx_W + 30x_Z \quad c - \text{parametr}$$

przy ograniczeniach

$$2x_W + x_Z \leq 6$$

$$x_W + 2x_Z \leq 8$$

$$x_W - x_Z \leq 1$$

$$x_W \leq 2$$

$$x_W \geq 0, x_Z \geq 0$$

wartość c	rozwiązanie optymalne	optymalna wartość f. celu
$(-\infty, 15]$	$x_W=0, x_Z=4$	120
$[15, 60]$	$x_W=1\frac{1}{3}, x_Z=3\frac{1}{3}$	$1\frac{1}{3}c + 100$
$[60, \infty)$	$x_W=2, x_Z=2$	$2c + 60$

Dualność

Zadanie pierwotne

$$\begin{array}{l} \max_{\mathbf{x}} x_0 = \mathbf{c}^T \mathbf{x} \\ \mathbf{A} \mathbf{x} = \mathbf{b} \\ \mathbf{x} \geq \mathbf{0} \end{array}$$

$$\begin{array}{l} \min_{\mathbf{x}} x_0 = \mathbf{c}^T \mathbf{x} \\ \mathbf{A} \mathbf{x} = \mathbf{b} \\ \mathbf{x} \geq \mathbf{0} \end{array}$$

$$\begin{array}{l} \max_{\mathbf{x}} x_0 = \mathbf{c}^T \mathbf{x} \\ \mathbf{A} \mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0} \end{array}$$

Zadanie dualne

$$\begin{array}{l} \min_{\mathbf{v}} v_0 = \mathbf{b}^T \mathbf{v} \\ \mathbf{A}^T \mathbf{v} \geq \mathbf{c} \\ \mathbf{v} - nieograniczone \end{array}$$

$$\begin{array}{l} \max_{\mathbf{v}} v_0 = \mathbf{b}^T \mathbf{v} \\ \mathbf{A}^T \mathbf{v} \leq \mathbf{c} \\ \mathbf{v} - nieograniczone \end{array}$$

$$\begin{array}{l} \min_{\mathbf{v}} v_0 = \mathbf{b}^T \mathbf{v} \\ \mathbf{A}^T \mathbf{v} \geq \mathbf{c} \\ \mathbf{v} \geq \mathbf{0} \end{array}$$

Właściwości zadań dualnych

Między zadaniem pierwotnym a zadaniem dualnym zachodzi dokładnie jeden z następujących związków:

- oba zadania mają skończone rozwiązania optymalne.
Wtedy $\max_{\mathbf{x}} \mathbf{c}^T \mathbf{x} = \min_{\mathbf{v}} \mathbf{b}^T \mathbf{v}$,
- jedno z zadań jest niedopuszczalne, a drugie nieograniczone,
- oba zadania są niedopuszczalne.

Jeżeli

\mathbf{x} - dowolne rozwiązanie dopuszczalne problemu pierwotnego

\mathbf{v} - dowolne rozwiązanie dopuszczalne problemu dualnego

to

$$\boxed{\mathbf{c}^T \mathbf{x} \leq \mathbf{b}^T \mathbf{v}}$$

Zadaniem dualnym do dualnego jest zadanie pierwotne

Analiza parametryczna

$$\max x_0 = 20x_W + 30x_Z$$

przy ograniczeniach

$$2x_W + x_Z \leq b$$

b – parametr

$$x_W + 2x_Z \leq 8$$

$$x_W - x_Z \leq 1$$

$$x_W \leq 2$$

$$x_W \geq 0, x_Z \geq 0$$

wartość b	rozwiązanie optymalne	wartość f. celu
($-\infty, 0]$	–	–
[0, 4]	$x_W=0, x_Z=b$	$30b$
[4, 7]	$x_W=(2b-8)/3, x_Z=(-b+16)/3$	$(10b + 320)/3$
[7, $\infty)$	$x_W=2, x_Z=3$	130

Przypadki w Programowaniu Liniowym

- zbiór rozwiązań dopuszczalnych jest pusty (ograniczenia są sprzeczne) – *brak rozwiązania*
- zbiór rozwiązań dopuszczalnych jest niepusty oraz funkcja celu jest ograniczona od góry (dla problemu maksymalizacji) – *istnieje co najmniej jedno rozwiązanie optymalne w punkcie wierzchołkowym*
- funkcja celu jest nieograniczona z góry na zbiorze rozwiązań dopuszczalnych – *zadanie jest nieograniczone, brak skończonego rozwiązania optymalnego*

Wniosek

Rozwiązań optymalnych zadania PL można szukać wśród punktów wierzchołkowych (dopuszczalnych rozwiązań bazowych)

Ogólna idea algorytmu sympleks

1. Wyznacz początkowy punkt wierzchołkowy (początkowe dopuszczalne rozwiązanie bazowe)
2. Test optymalności – czy rozwiązanie gorsze od sąsiednich? Jeżeli nie, to STOP – znaleziono rozwiązanie optymalne, jeżeli tak idź do 3.
3. Przejdź do sąsiedniego punktu wierzchołkowego, dającego lepszą wartość funkcji celu. Idź do 2.

Szczegółowe elementy algorytmu sympleks

- wyznaczanie początkowego rozwiązania bazowego
- warunki optymalności rozwiązania bazowego
- wykrywanie niedopuszczalności i nieograniczoneści
- sposób przechodzenia z bazy do bazy
- postępowanie w przypadku degeneracji

Początkowe rozwiązanie bazowe

ZPL

$$\begin{aligned} \max_{\mathbf{x}} x_0 &= \mathbf{c}^T \mathbf{x} \\ \mathbf{Ax} &= \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \end{aligned}$$

- Metoda kar

$$\begin{aligned} \max_{\mathbf{x}, \mathbf{x}_s} x_0 &= \mathbf{c}^T \mathbf{x} - M \mathbf{1}^T \mathbf{x}_s \\ \mathbf{Ax} + \mathbf{Ix}_s &= \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0}, \mathbf{x}_s \geq \mathbf{0} \end{aligned}$$

- Metoda dwufazowa

1. faza - wyznaczenie dopuszczalnego rozwiązania bazowego

$$\begin{aligned} \max_{\mathbf{x}, \mathbf{x}_s} z_0 &= -\mathbf{1}^T \mathbf{x}_s \\ \mathbf{Ax} + \mathbf{Ix}_s &= \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0}, \mathbf{x}_s \geq \mathbf{0} \end{aligned}$$

2. faza - algorytm sympleks

Modele sieciowe

Siecią (przepływową) nazywamy graf skierowany $G(V,E)$, w którym każdy łuk $(u,v) \in E$ ma nieujemną **przepustowość** $c_{uv} \geq 0$. (jeżeli $(u,v) \notin E$, to przyjmujemy $c_{uv}=0$).

W sieci wyróżniamy dwa wierzchołki: **źródło s i ujście t**.

Przepływem o **wartości F** ze źródła s do ujścia t nazywamy dowolną funkcję $f: E \rightarrow R^+$ spełniającą następujące warunki.

- $\sum_{z \in V} f(v,z) - \sum_{u \in V} f(u,v) = \begin{cases} F & \text{dla } v = s \\ 0 & \text{dla } v \in V - \{s,t\} \\ -F & \text{dla } v = t \end{cases}$
- $0 \leq f(u,v) \leq c_{uv}$ dla każdego $(u,v) \in V$

Podstawowe modele sieciowe

• problem maksymalnego przepływu

Należy znaleźć przepływ o maksymalnej wartości ze źródła s do ujścia t .

• problem najtańszego przepływu

Dane są koszty jednostkowe przepływów w łukach. Należy znaleźć przepływ o zadanej wartości ze źródła s do ujścia t i minimalnym sumarycznym koszcie.

Właściwości modeli sieciowych

- dobrze modelują wiele rzeczywistych problemów decyzyjnych
- istnieją bardzo efektywne (wielomianowe) algorytmy rozwiązywania
- jeżeli parametry (dane liczbowe) charakteryzujące sieć są całkowitoliczbowe, to istnieją rozwiązania całkowitoliczbowe
- są szczególnym przypadkiem Zadań Programowania Liniowego

Schemat algorytmu ścieżek powiększających Forda-Fulkersona dla problemu maksymalnego przepływu

(w nawiasach modyfikacje dla problemu najtańszego przepływu)

1. Startujemy z przepływu dopuszczalnego np. zerowego
2. Budujemy graf użytecznych łuków dla aktualnego przepływu f (*koszty łuków przeciwnych ujemne*)
3. Znajdujemy dowolną (*najtańszą*) ścieżkę powiększającą łączącą s z t .
4. Zwiększamy przepływ wzdłuż ścieżki powiększającej, tak aby
 - nie przekroczyć przepustowości ścieżki
 - (*sumaryczny przepływ nie był większy od zadanej wartości przepływu*)
5. Aktualizujemy przepływ i wracamy do p. 2.

Koniec algorytmu

- nie ma ścieżek powiększających od s do t
- (*osiągamy zadaną wartość przepływu*)

Przekroje w sieciach

Niech S i $T = V - S$ będzie dowolnym podziałem zbioru wierzchołków sieci $G(V, E)$ takim, że $s \in S$ i $t \in T$.

Zbiór łuków (u, v) takich, że $u \in S$ i $v \in T$ nazywamy **przekrojem** i oznaczamy (S, T) .

Przepustowość przekroju $c(S, T)$

$$c(S, T) = \sum_{(u, v) \in (S, T)} c(u, v)$$

Tw.1 Wartość dowolnego przepływu w sieci nie jest większa niż przepustowość dowolnego przekroju.

Tw.2 Wartość maksymalnego przepływu w sieci G jest równa przepustowości minimalnego przekroju w sieci G .

Zapis problemów sieciowych w postaci Zadań Programowania Liniowego

zmienne decyzyjne – przepływy $f_{u,v}$ w poszczególnych łukach
ograniczenia – bilans przepływu w wierzchołkach sieci

- **problem maksymalnego przepływu**

$$\begin{aligned} & \max F \\ & \sum_{z \in V} f_{vz} - \sum_{u \in V} f_{uv} = 0 && v \neq s, v \neq t \\ & \sum_{z \in V} f_{sz} - F = 0 \\ & 0 \leq f_{uv} \leq c_{uv} && (u, v) \in E \end{aligned}$$

- **problem najtańszego przepływu**

$$\begin{aligned} & \min \sum_{(u, v) \in E} d_{uv} f_{uv} \\ & \sum_{z \in V} f_{vz} - \sum_{u \in V} f_{uv} = 0 && v \neq s, v \neq t \\ & \sum_{z \in V} f_{sz} = F \\ & 0 \leq f_{uv} \leq c_{uv} && (u, v) \in E \end{aligned}$$

Przykładowe klasy zadań, które można sformułować w postaci modeli sieciowych

- zadania transportowe
- zadania przydziału
- zadania znajdowania najkrótszej ścieżki

Problem znajdowania najkrótszej ścieżki (szczególny przypadek zadania najtańszego przepływu)

Algorytmy

- jednakowe długości łuków – przeszukiwanie wszerz
- grafy acykliczne – jak w metodzie ścieżki krytycznej
- nieujemne długości łuków – algorytm *Dijkstry*
- ścieżki między wszystkimi parami wierzchołków – algorytm *Floyda-Warshalla*

Algorytm Dijkstry

wyznaczania odległości od wybranego wierzchołka do pozostałych w sieci z **nieujemnymi długościami łuków**

Dane:

- graf skierowany $G(V,E)$
- macierz długości łuków $A[u,v]$ $1 \leq u,v \leq n$ grafu. Zakładamy $A[u,v] \geq 0$. Przyjmujemy $A[u,u]=0$ oraz $A[u,v]=\infty$, gdy nie ma łuku między wierzchołkami u oraz v
- s – wierzchołek z którego wyznaczamy odległości.

Algorytm:

```
for  $u \in V$  do  $D[u]:=A[s,u]$  ;
 $S := \{s\}$  ;
while  $S \neq V$  do
     $v :=$  wierzchołek ze zbioru  $V-S$  taki, że
         $D[v]=\min\{D[u] : u \in V-S\}$  ;
     $S := S \cup \{v\}$  ;
    for  $u \in V-S$  do  $D[u]:= \min\{D[u], D[v] + D[v,u]\}$  ;
```

Wynik: Tablica długości najkrótszych połączeń $D[u]$ z wierzchołka s do wierzchołka u .

Złożoność: $O(n^2)$

Algorytm Floyda-Warshalla

wyznaczania długości najkrótszych połączeń między wierzchołkami grafu

Dane: macierz długości łuków $A[u,v]$ $1 \leq u,v \leq n$ grafu skierowanego (bez cykli o ujemnej długości). Przyjmujemy $A[u,u]=0$ oraz $A[u,v]=\infty$, gdy nie ma łuku między wierzchołkami u oraz v .

Algorytm:

```
for  $u:=1$  to  $n$  do
    for  $v:=1$  to  $n$  do  $D[u, v]:=A[u, v]$  ;
for  $m:=1$  to  $n$  do
    for  $u:=1$  to  $n$  do
        for  $v:=1$  to  $n$  do
             $D[u, v]:= \min\{D[u, v], D[u, m] + D[m, v]\}$ ;
```

Wynik: Tablica długości najkrótszych połączeń $D[u, v]$ między parami wierzchołków

Złożoność: $O(n^3)$

Programowanie dyskretne

Dotyczy problemów decyzyjnych, w których pewne zmienne decyzyjne mogą przyjmować tylko dyskretne wartości np.:

- całkowitoliczbowe
- binarne (0 lub 1)

W zależności od rodzaju występujących zmiennych wyróżniamy zadania programowania

- całkowitoliczbowego
- binarnego
- mieszanego – występują zmienne ciągłe i dyskretne

Przykład

$$\begin{aligned} \max z &= 7x_1 + 8x_2 \\ 3x_1 + 4x_2 &\leq 4 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Rozwiżanie: $x_1 = 4/3$, $x_2 = 0$, $z = 28/3$.

Przy dodatkowym warunku
 x_1 oraz x_2 całkowitoliczbowe

Rozwiżanie: $x_1 = 0$, $x_2 = 1$, $z = 8$.

Sytuacje, w których stosuje się zmienne dyskretne

- natura poszukiwanych rozwiązań jest dyskretna
 - wyznaczanie liczby niepodzielnych obiektów np. procesorów, zadań, pracowników itp.
 - określanie permutacji lub kombinacji pewnego zbioru obiektów – problemy kombinatoryczne
- wybór decyzji spośród wielu wariantów
- modelowanie funkcji kawałkami liniowymi, np. uwzględnianie kosztów stałych

Metody rozwiązywania problemów dyskretnych

- algorytmy specjalizowane
- programowanie sieciowe
- metody przeglądu
 - metoda podziału i oszacowań (*branch & bound*)
 - *branch & cut, branch & price* i inne odmiany
 - programowanie w logice ograniczeń
- metody odcięć
- programowanie dynamiczne
- metody heurystyczne specjalizowane
- metaheurystyki
 - algorytmy genetyczne (ewolucyjne)
 - przeszukiwanie „tabu search”
 - symulowane wyżarzanie (wychładzanie)
 - algorytmy randomizowane, np. GRASP

Wykorzystywane „narzędzia”

- solver'y Zadań Programowania Mieszanego
 - wymagają zapisania problemu w postaci zadania programowania liniowego z ewentualnymi warunkami całkowitoliczbowości
- pakiety Programowania w Logice Ograniczeń
 - wymagają zapisania problemu w stosownym języku, zwykle dość „elastycznym”
 - dla uzyskania efektywności potrzebują sformułowania dobrych reguł sterowania procesem przeglądu
- programy specjalizowane dla danej klasy problemów
 - wymagają zapisania danych wejściowych w stosownym formacie
- samodzielna implementacja algorytmów (zaczerpniętych z literatury, opracowanych bądź dostosowanych przez siebie)

Problem P

$$z(P) = \max_{x \in F(P)} f(x)$$

Relaksacja

Problem A

$$z(A) = \max_{x \in F(A)} g(x)$$

jest relaksacją problemu P jeżeli

- (i) $f(x) \leq g(x)$ dla każdego $x \in F(P)$
- (ii) $F(P) \subseteq F(A)$

Restrykcja

Problem B

$$v(B) = \max_{x \in F(B)} h(x)$$

jest restrykcją problemu P jeżeli

- (i) $f(x) \geq h(x)$ dla każdego $x \in F(B)$
- (ii) $F(P) \supseteq F(B)$

Metoda podziału i oszacowań

(dla zadania maksymalizacji)

1. Wybór podproblemu P_i do analizy (na początku $P_0 = P$).
2. Oszacowanie optymalnej wartości funkcji celu dla podproblemu P_i
 - od dołu \underline{z}_i – z rozwiązania dopuszczalnego (dobrego);
 - od góry \bar{z}_i – z relaksacji (np. relaksacji liniowej);
3. Sondaż podproblemu P_i
 - gdy P_i niedopuszczalny zamykamy jego analizę;
 - gdy $\underline{z}_i = \bar{z}_i$ podproblem rozwiązyany – zamykamy jego analizę. Jeżeli $\underline{z}_i > \underline{z}_0$, to znaleźliśmy lepsze rozwiązanie dopuszczalne problemu P , a więc $\underline{z}_0 := \underline{z}_i$;
 - gdy $\bar{z}_i \leq \underline{z}_0$ – w podproblemie nie ma lepszych rozwiązań niż dotychczas znalezione (o wartości funkcji celu \underline{z}_0). Zamykamy analizę podproblemu;
 - gdy $\bar{z}_i > \underline{z}_0$, $\underline{z}_i < \bar{z}_i$ dokonujemy podziału podproblemu P_i na podproblemy P_j (nowe gałęzie drzewa), tak aby

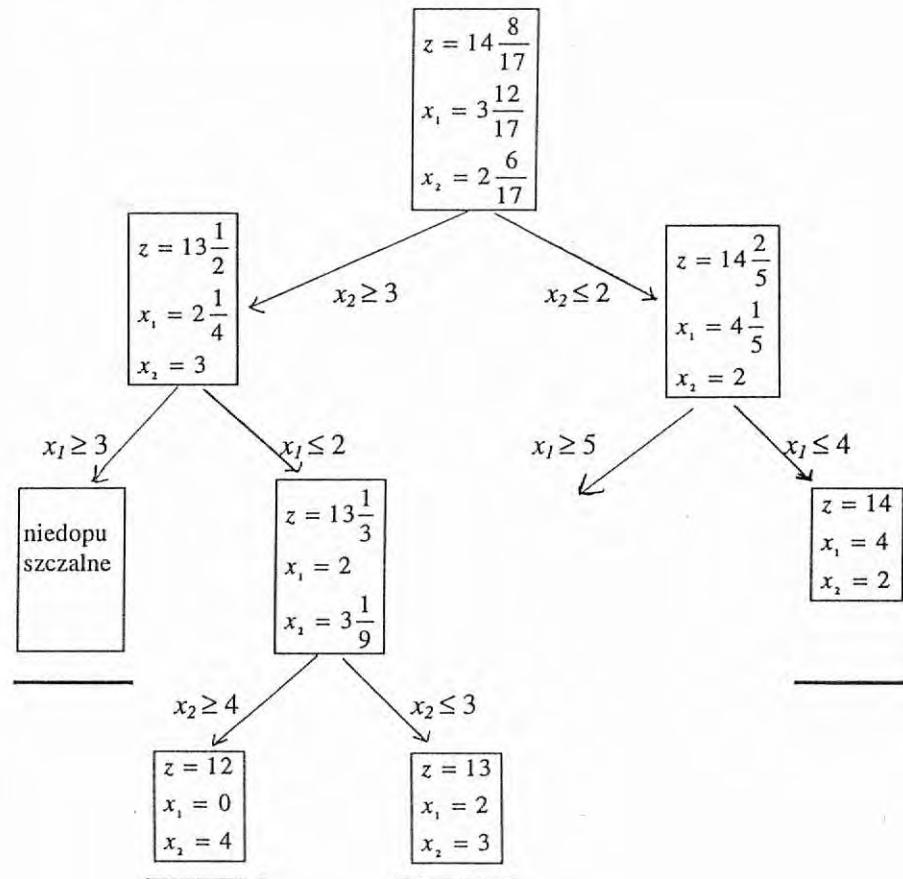
$$\underset{j}{\text{UF}}(P_j) = F(P_i)$$

4. Jeżeli wszystkie podproblemy są zamknięte to STOP.
W przeciwnym przypadku idziemy do 1.

Dla zadania minimalizacji analiza oszacowań jest oparta na odwrotnych relacjach.

Przykład

$$\begin{aligned} \max z &= 2x_1 + 3x_2 \\ 5x_1 + 7x_2 &\leq 35 \\ 4x_1 + 9x_2 &\leq 36 \\ x_1, x_2 &\geq 0 \quad \text{całkowite} \end{aligned}$$



Kwestie do rozstrzygnięcia

- wybór podproblemu do analizy
- rodzaj stosowanych oszacowań
 - rodzaj relaksacji – kompromis między czasem obliczeń a dokładnością
 - strategia wyznaczania dobrych rozwiązań dopuszczalnych
- sposób podziału na podproblemy – drzewo binarne, wielogązowe itd.
- wybór zmiennej, względem której następuje podział

Przykład – problem plecakowy

Sformułowanie

Które spośród n przedmiotów o wartościach c_1, c_2, \dots, c_n i ważących odpowiednio a_1, a_2, \dots, a_n należy zapakować do plecaka o ładowności b , aby łączna wartość zapakowanych przedmiotów była jak największa ?

Model Programowania Binarnego

$$\begin{aligned} \max z &= \sum_{i=1}^n c_i x_i \\ \sum_{i=1}^n a_i x_i &\leq b \\ x_i &\in \{0,1\}, \quad i = 1, \dots, n \end{aligned}$$

Relaksacja liniowa problemu plecakowego

$$\max z = \sum_{i=1}^n c_i x_i$$

$$\sum_{i=1}^n a_i x_i \leq b$$

$$0 \leq x_i \leq 1, \quad i = 1, \dots, n$$

Właściwości relaksacji liniowej

- Niech uporządkowanie przedmiotów będzie takie, że

$$c_1/a_1 \geq c_2/a_2 \geq \dots \geq c_n/a_n$$

Wtedy rozwiązanie

$$x_1=1, x_2=1, \dots, x_{p-1}=1, x_p = (b - \sum_{i=1}^{p-1} a_i)/a_p, x_{p+1}=0, \dots, x_n=0$$

$$\text{gdzie } p = \min\{ j : \sum_{i=1}^j a_i > b \}$$

jest optymalnym rozwiązaniem relaksacji liniowej problemu plecakowego

- Zachłanny algorytm rozwiązywania

Powłoka wypukła

Najmniejszy zbiór wypukły zawierający zbiór rozwiązań dopuszczalny problemu dyskretnego

Właściwości

Relaksacja liniowa problemu ograniczonego do powłoki wypukłej daje rozwiązanie optymalne problemu dyskretnego.

Idea metod odcięć

(kolejne „przybliżanie” powłoki wypukłej)

- rozwiązanie relaksacji liniowej;
- jeżeli rozwiązanie dopuszczalne (całkowitoliczbowe), to STOP.
W przeciwnym przypadku idź do 3;
- generacja i dodawanie ograniczenia odcinającego uzyskanie rozwiązanie, ale nie naruszające powłoki wypukłej. Idź do 1;

Szeregowanie zadań

- Operacje – pogrupowane w Zadania
- Procesory
- Dodatkowe zasoby

Ograniczenia

- żadna operacja nie może być jednocześnie wykonywana na więcej niż jednym procesorze
- żaden procesor nie może jednocześnie wykonywać więcej niż jedną operację
- inne – np. kolejnościowe, czasowe itd.

Klasyczny problem szeregowania

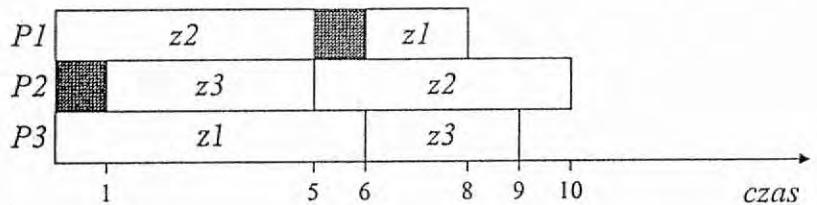
Należy ustalić kolejność wykonywania zadań na poszczególnych procesorach (oraz ewentualnie przydział dodatkowych zasobów), tak aby zachowując ograniczenia optymalizować zadane kryterium uszeregowania

Typowe parametry zadania (operacji)

- p_{lj} – czas wykonywania na procesorze l
- r_j – termin gotowości do wykonania
- d_j – żądaný termin ukończenia
- w_j – priorytet (waga)
- ograniczenia kolejnościowe
- możliwość przerywania (podzielność)
- wymagania zasobowe

Uszeregowanie (harmonogram)

Prezentacja graficzna – wykres Gantta



Parametry oceny uszeregowania

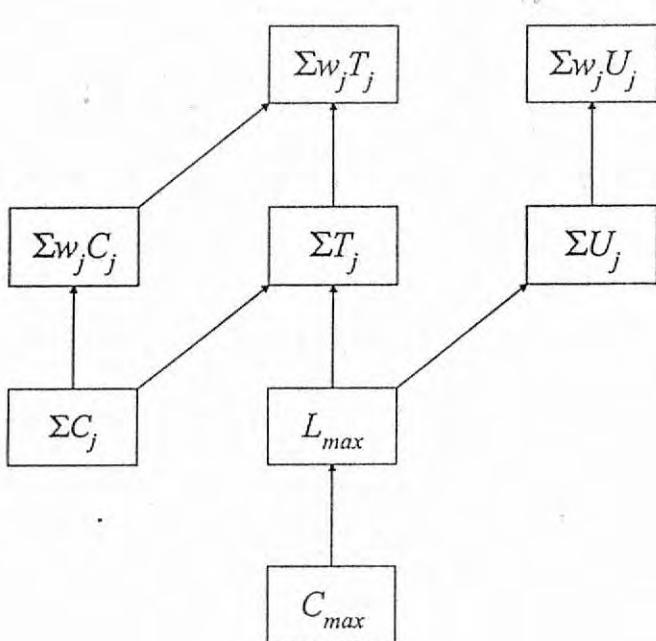
- C_j – termin (moment) ukończenia
- F_j – czas przepływu $F_j = C_j - r_j$
- L_j – nieterminowość $L_j = C_j - d_j$
- T_j – spóźnienie $T_j = \max\{C_j - d_j, 0\}$
- U_j – czy zadanie (operacja) spóźnione

Typowe kryteria uszeregowania (minimalizowane)

- C_{\max} – długość uszeregowania $C_{\max} = \max\{C_j\}$
- L_{\max} – maksymalna nieterminowość $L_{\max} = \max\{L_j\}$
- \bar{F} – średni czas przepływu ($1/n \sum F_j$)
- \bar{F}_w – średni ważony czas przepływu ($\sum w_j F_j / \sum w_j$)
- $\sum T_j$ – sumaryczne (średnie) spóźnienie
- $\sum w_j T_j$ – sumaryczne (średnie) ważone spóźnienie
- $\sum U_j$ – liczba zadań spóźnionych
- $\sum w_j U_j$ – sumaryczne kary za spóźnienie

Relacje między kryteriami

- kryteria \bar{F} , $\sum C_j$, $\sum F_j$, $\sum L_j$ są równoważne
- kryteria, \bar{F}_w , $\sum w_j C_j$, $\sum w_j F_j$, $\sum w_j L_j$ są równoważne
- uszeregowanie minimalizujące L_{\max} minimalizuje również T_{\max} , ale kryteria te nie są równoważne



Klasyfikacja problemów szeregowania

- szeregowanie na jednym procesorze
- szeregowanie na procesorach równoległych
 - jednakowych (P)
 - jednorodnych (Q)
 - różnych (R)
- szeregowanie w systemie przepływowym
 - permutacyjnym (PF)
 - ogólnym (F)
- szeregowanie w systemie gniazdowym (J)
- szeregowanie w systemie otwartym (O)

Notacja

$\alpha \mid \beta \mid \gamma$

- α – rodzaj systemu i liczba procesorów,
 - * rodzaj systemu np., P, Q, R, F, PF, J, O
 - * drugie pole puste gdy liczba procesorów nie jest ustalona
- β – charakterystyka zadań
 - * gdy pole puste – zadania niepodzielne, niezależne, z zerowymi (jednakowymi) terminami gotowości, bez dodatkowych wymagań zasobowych
 - * $pmtn$ – zadania podzielne
 - * $prec$ – występują ograniczenia kolejnościowe (zadania zależne)
 - * r_j – różne terminy gotowości zadań
 - * res – występują wymagania na dodatkowe zasoby
- γ – rodzaj kryterium szeregowania, np.

$C_{\max}, L_{\max}, \Sigma C_j, \Sigma w_j C_j, \Sigma T_j, \Sigma w_j T_j, \Sigma U_j, \Sigma w_j U_j$

Szeregowanie na jednym procesorze

- $1 \mid r_j \mid C_{\max}$ – szeregowanie według niemalejących r_j
- $1 \mid \mid L_{\max}$ – reguła EDD (*Earliest Due Date*)
- $1 \mid r_j \mid L_{\max}$ – problem NP-trudny
- $1 \mid \mid \Sigma C_j$ – reguła SPT (*Shortest Processing Time*)
- $1 \mid \mid \Sigma w_j C_j$ – szeregowanie według niemalejących współczynników p_j / w_j
- $1 \mid \mid \Sigma T_j$ – problem NP-trudny
- $1 \mid \mid \Sigma U_j$ – algorytm Moore'a (Hodgsona)
 1. uszereguj zadania według niemalejących d_j ;
 2. znajdź pierwsze zadanie j , które jest opóźnione. Jeżeli nie ma zadań opóźnionych to STOP;
 3. spośród uszeregowanych zadań $1, \dots, j$ usuń zadanie z najdłuższym czasem wykonywania p_j ;
Usunięte zadania umieść na końcu uszeregowania.
- $1 \mid \mid \Sigma w_j U_j$ – problem NP-trudny
- problem z przebrojeniami – NP-trudny

Szeregowanie na procesorach równoległych

- $P2 \parallel C_{\max}$ – problem NP-trudny
- $P \mid pmtn \mid C_{\max}$ – przy m procesorach
 $C^*_{\max} = \max \{ \max \{p_j\}, 1/m \sum p_j \}$
- $R \mid pmtn \mid C_{\max}$ – metoda dwufazowa
- $P \parallel \Sigma C_j$ – uogólniona reguła SPT
- $R \parallel \Sigma C_j$ – model sieciowy
- $P2 \parallel \Sigma w_j C_j$ – problem NP-trudny

Szeregowanie czasooptymalne zadań podzielnych na procesorach równoległych

Problem $R \mid pmtn \mid C_{\max}$

Metoda dwufazowa

- faza pierwsza – wyznaczenie minimalnej długości uszeregowania i przydział zadań do procesorów

t_{lj} – zmienna decyzyjna oznaczająca sumaryczny czas realizacji zadania j na procesorze l

$$\begin{aligned} & \min C_{\max} \\ & \text{przy ograniczeniach} \\ & \sum_j t_{lj} \leq C_{\max} \quad \forall l \\ & \sum_l t_{lj} \leq C_{\max} \quad \forall j \\ & \sum_l \frac{1}{p_{lj}} t_{lj} = 1 \quad \forall j \\ & t_{lj} \geq 0 \quad \forall l, j \end{aligned}$$

- faza druga – określenie uszeregowania problem $O \mid pmtn \mid C_{\max}$

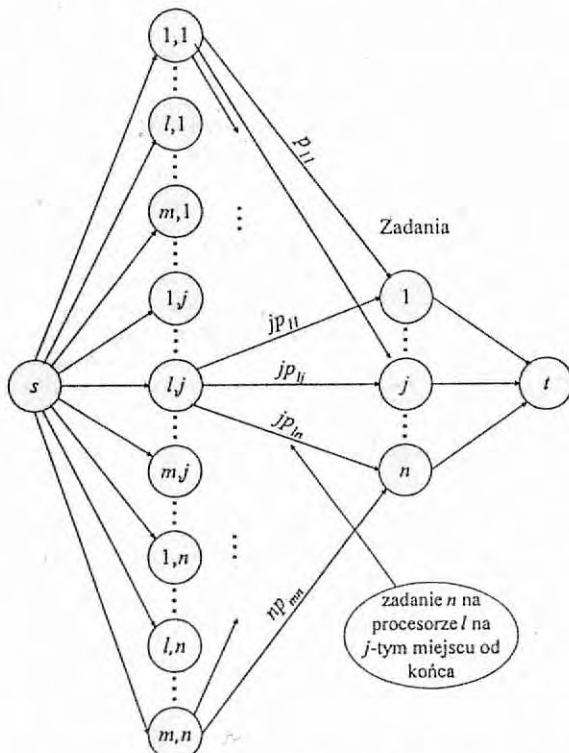
model sieciowy (w przypadku całkowitych czasów wykonywania operacji – model kolorowania krawędzi multigrafu dwudzielnego)

Szeregowanie zadań na procesorach równoległych z kryterium ΣC_j

Problem $R \parallel \Sigma C_j$

Model sieciowy

n zadań, m procesorów równoległych



Przepustowości wszystkich łuków $[0, 1]$

Wartość przepływu od s do t równa n

Problem przepływowy (flow shop) (minimalizacja C_{\max})

- permutacyjny – wszystkie procesory wykonują zadania w tej samej kolejności
- ogólny – każdy z procesorów może wykonywać zadania w dowolnej kolejności

Istnieje rozwiązanie optymalne ogólnego problemu przepływowego, w którym

- kolejność wykonywania zadań na pierwszych dwóch procesorach jest taka sama
- kolejność wykonywania zadań na ostatnich dwóch procesorach jest taka sama

Wniosek

Przy liczbie procesorów $m \leq 3$ rozwiązań optymalnych ogólnego problemu przepływowego wystarczy poszukiwać wśród uszeregowień permutacyjnych.

- $F2 \parallel C_{\max}$ – algorytm Johnsona
- $F3 \parallel C_{\max}$ – problem NP-trudny

Problem przepływowy, 2 procesory

$F2 \mid \mid C_{\max}$

Algorytm Johnsona

1. podziel zadania na dwa zbiorы

- $S1 \leftarrow \{ j : p_{1j} \leq p_{2j} \}$
- $S2 \leftarrow \{ j : p_{1j} > p_{2j} \}$

2. wykonaj najpierw zadania ze zbioru $S1$ w kolejności niemalejących wartości p_{1j}

3. następnie wykonaj zadania ze zbioru $S2$ w kolejności niesosnących wartości p_{2j}

Kolejność wykonywania zadań na obu procesorach taka sama

Szeregowanie w systemie otwartym

- $O2 \mid \mid C_{\max}$ – algorytm Gonzalez i Sahni'ego
- $O3 \mid \mid C_{\max}$ – problem NP-trudny
- $O \mid pmtn \mid C_{\max}$

$$C_{\max}^* = \max \{ \max_t \{ \sum_j p_{tj} \}, \max_j \{ \sum_t p_{tj} \} \}$$

model sieciowy (w przypadku całkowitych czasów wykonywania operacji – model kolorowania krawędzi multigrafu dwudzielnego)

Dynamiczne reguły szeregowania

- **FIFO** (*First-In First-Out*), **FCFS** (*First-Come First-Served*) – według kolejności pojawiania się zadań
- **SPT** (*Shortest Processing Time*) – najpierw wykonywane najkrótsze zadania

Reguła korzystna w przypadku:

- minimalizacji średniego czasu przepływu
- minimalizacji średniej liczby zadań przebywających w systemie
- minimalizacji średniego spóźnienia
- minimalizacji liczby zadań spóźnionych

Niedogodności:

- blokuje zadania o długich czasach wykonywania

- **LPT** (*Longest Processing Time*) – najpierw wykonywane najdłuższe zadania

Reguła korzystna w przypadku:

- szeregowania czasoptymalnego na procesorach równoległych, np. dla problemu $P_m \parallel C_{\max}$

$$C_{\max}(LPT) \leq \left(\frac{4}{3} - \frac{1}{3m}\right) C_{\max}^*$$

Dynamiczne reguły szeregowania dla zadań z żądanymi terminami ukończenia

- **EDD** (*Earliest Due Date*) – najpierw zadania z najwcześniejszym żądanym terminem ukończenia
- **MOD** – szeregowanie według zmodyfikowanego terminu zakończenia $MOD = \max\{d_j, t + P_j(t)\}$, gdzie
 - * t – aktualna chwila
 - * $P_j(t)$ – suma czasów realizacji pozostałych do wykonania operacji zadania
- **CR** (*Critical Ratio*) – szeregowanie według wskaźnika

$$CR = \frac{d_j - t}{P_j(t)}$$

- **STO** (*Slack Time per Operation*) – szeregowanie według wskaźnika

$$STO = \frac{d_j - t - P_j(t)}{NOP} \text{ gdzie}$$

NOP – liczba operacji zadania pozostałych do wykonania