Wrocław, 29 kwietnia 2021

Zaawansowane **T**echniki **O**ptymalizacji

Problemy optymalizacji dyskretnej i ciągłej

prowadzący: *dr inż. Jarosław Rudy*

---

1 Generator liczb pseudolosowych

Do niektórych laboratoriów będzie istniała konieczność wygenerowania określonych instancji problemów. Instancje będziemy identyfikować za pomocą ich rozmiaru (jeden lub kilka parametrów, zależnie od problemu) tzw. ziarna (seed). Sposób generacji instancji podany jest przy poszczególnych problemach. Do losowaniach danych będziemy korzystać z generatora liczb pseudolosowych zaopatrzonego w funkcje generacji następnej liczby całkowitej oraz następnej liczby rzeczywistej (z zadanych przedziałów obustronnie domkniętych). Generator korzysta z zadanego ziarna. Na stronie kursu umieszczone są kody źródłowe tego generatora dla czterech języków programowania: `C/C++`, `python`, `C#`, oraz `Java`. Nie należy losować instancji z użyciem innych generatorów. W dalszej części opisu przez $\mathrm{init}(Z)$ rozumiemy inicjalizację generatora z użyciem ziarna Z, zaś przez $\mathrm{nextInt}(a, b)$ oraz $\mathrm{nextFloat}(a, b)$ losowanie odpowiednio liczby całkowitej i rzeczywistej z przedziału $[a, b]$.

2 Problemy optymalizacji dyskretnej

2.1 Problem sumy podzbioru (Subset Sum Problem, SSP)

Dany jest zbiór $S = \{s_1, s_2, \ldots, s_n\}$ zawierający $n \in \mathbb{N}_+$ liczb całkowitych tj. $s_i \in \mathbb{Z}$. Należy zaznaczyć, że s_i może być ujemne. Dana jest też wartość $T \in \mathbb{Z}$. W podstawowej wersji optymalizacyjnej należy znaleźć taki podzbiór X zbioru S aby suma jego elementów była możliwie bliska wartości T tj.:

$$\min_{X \in 2^S} |T - \sum_{x \in X} x|, \tag{1}$$

gdzie 2^S to zbiór potęgowy S, czyli zbiór wszystkich podzbiorów S.

Generacja instancji Dla parametru n oraz ziarna Z:

1. $\mathrm{init}(Z)$.
2. Dla i od 1 do n:
   2.1. $s_i \leftarrow \mathrm{nextInt}(-100, 100)$.
3. $T \leftarrow \mathrm{nextInt}(-50 \cdot n, 50 \cdot n)$.

2.2 Zagadnienie transportowe (Transportation Problem, TP)

Dane jest $n \in \mathbb{N}_+$ dostawców i $m \in \mathbb{N}_+$ odbiorców. Podaż (produkcja) dostawcy i wynosi $S_i \in \mathbb{N}_+$, zaś popyt (zapotrzebowanie) odbiorcy j wynosi $D_j \in \mathbb{N}_+$. Dodatkowo wiemy, że suma popytów równa jest sumie podaży:

$$\sum_{i=1}^{n} S_i = \sum_{j=1}^{m} D_j. \tag{2}$$

Koszt transportu 1 jednostki towaru od dostawcy i do odbiorcy j wynosi $k_{ij} \in \mathbb{N}_+$. Należy dla każdej pary (i, j) określić wartość $x_{ij} \in \mathbb{N}_0$ (liczba jednostek transportowanych z i do j), tak by zminimalizować łączny koszt transportu. Innymi słowy:

$$\textbf{minimize} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} k_{ij}, \tag{3}$$

$$\textbf{subject to} \quad \sum_{i=1}^{n} x_{ij} = D_j \qquad \text{dla } j = 1, 2, \ldots, m, \tag{4}$$

$$\sum_{j=1}^{m} x_{ij} = S_i \qquad \text{dla } i = 1, 2, \ldots, n. \tag{5}$$

Generacja instancji Dla parametrów n i m oraz ziarna Z:

1. init(Z).
2. $K \leftarrow min(n, m)$.
3. Dla i od 1 do K:
   - 3.1. $S_i \leftarrow$ nextInt$(1, 20)$.
   - 3.2. $D_i \leftarrow S_i$.
4. Jeżeli $n > m$ to dla i od $K + 1$ do n:
   - 4.1. $r \leftarrow$ nextInt$(1, 20)$.
   - 4.2. $S_i \leftarrow r$.
   - 4.3. $j \leftarrow$ nextInt$(1, m)$.
   - 4.4. $D_j \leftarrow D_j + r$.
5. Jeżeli $m > n$ to dla j od $K + 1$ do m:
   - 5.1. $r \leftarrow$ nextInt$(1, 20)$.
   - 5.2. $D_j \leftarrow r$.
   - 5.3. $i \leftarrow$ nextInt$(1, n)$.
   - 5.4. $S_i \leftarrow S_i + r$.
6. Dla i od 1 do n:
   - 6.1. Dla j od 1 do m:
     - 6.1.1. $k_{ij} \leftarrow$ nextInt$(1, 30)$.

2.3 Zagadnienie przydziału (Assignment Problem, AP)

Dane jest $n \in \mathbb{N}_+$ zadań i tyle samo wykonawców. Koszt wykonania zadania i przez wykonawcę j dany jest przez $k(i, j) \in \mathbb{N}_+$. Należy określić przydział zadań do wykonawców (każde zadanie ma mieć dokładnie jednego wykonawcę i na odwrót), tak by minimalizować koszt wykonania wszystkich zadań:

$$\min_f \sum_{i=1}^{n} k(i, f(i)), \tag{6}$$

gdzie $f(i)$ jest wykonawcą, do którego przydzielono zadanie i.

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:

   2.1. Dla j od 1 do n:

      2.1.1. $k(i, j) \leftarrow \text{nextInt}(1, 50)$.

2.4 Kwadratowe zagadnienie przydziału (Quadratic Assignment Problem, QAP)

Dane jest $n \in \mathbb{N}_+$ zakładów i tyle samo lokalizacji. Dla każdej pary zakładów (i, j) określony jest przepływ $w(i, j) \in \mathbb{N}_+$, zaś dla każdej pary lokalizacji określona jest odległość $d(i, j) \in \mathbb{N}_+$. Należy określić przydział zakładów do lokalizacji (każdy zakład ma mieć dokładnie jedną lokalizację i na odwrót), tak by minimalizować koszt dany jako suma iloczynów przepływów i odległości zakładów:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} w(i, j) d(f(i), f(j)), \tag{7}$$

gdzie $f(i)$ jest lokalizacją, do której przydzielono zakład i.

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:

   2.1. Dla j od 1 do n:

      2.1.1. $w(i, j) \leftarrow \text{nextInt}(1, 50)$.

      2.1.2. $d(i, j) \leftarrow \text{nextInt}(1, 50)$.

2.5 Dyskretny problem plecakowy (Discrete Knapsack Problem, DKP)

Dane jest zbiór $P = \{1, 2, \ldots, n\}$ zawierający $n \in \mathbb{N}_+$ przedmiotów. Dla każdego przedmiotu $i \in P$ dana jest jego wartość $c_i \in \mathbb{N}_+$ oraz waga $w_i \in \mathbb{N}_+$. Oprócz tego dana jest pojemność plecaka $B \in \mathbb{N}_+$. W podstawowej wersji zadanie polega na wyznaczeniu takiego podzbioru X zbioru P, że suma wag przedmiotów z X jest nie większa niż B zaś suma ich wartości jest maksymalna tj.:

$$\textbf{maximize} \quad \sum_{p \in P} x_i c_i, \tag{8}$$

$$\textbf{subject to} \quad \sum_{p \in P} x_i w_i \leqslant B, \tag{9}$$

gdzie $x_i = 1$ jeśli przedmiot i jest w plecaku oraz $x_i = 0$ w przeciwnym wypadku.

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   2.1. $c_i \leftarrow \text{nextInt}(1, 30)$.
   2.2. $w_i \leftarrow \text{nextInt}(1, 30)$.
3. $B \leftarrow \text{nextInt}(5 \cdot n, 10 \cdot n)$.

2.6 Problem komiwojażera (Travelling Salesman Problem, TSP)

Dane jest $n \in \mathbb{N}_+$ miast. Odległość od miasta i do miasta j dana jest przez $d_{ij} \in \mathbb{N}_+$, przy czym nie musi zachodzi $d_{ij} = d_{ji}$, zaś odległości postaci d_{ii} (z danego miasta do siebie samego) są nieistotne. Należy określić kolejność (permutację) odwiedzania miast, tak by zminimalizować długość cyklu Hamiltona:

$$\min_f d(f(n), f(1)) + \sum_{i=1}^{n-1} d(f(i), f(i+1)), \tag{10}$$

gdzie $f(i)$ określa które w kolejności odwiedzania jest miasto i.

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   2.1. Dla j od 1 do n:
      2.1.1. $d_{ij} \leftarrow \text{nextInt}(1, 30)$.

2.7 Szeregowanie zadań na jednej maszynie z ważoną sumą opóźnień (Scheduling on a single machine with total weighted tardiness, $1||w_iT_i$)

Dane jest $n \in \mathbb{N}_+$ zadań. Zadanie i ma czas wykonania równy $p_i \in \mathbb{N}_+$, wagę $w_i \in \mathbb{N}_+$ oraz pożądany termin zakończenia (deadline) równy $d_i \in \mathbb{N}_+$. Należy określić kolejność wykonywania zadań π na maszynie, tak aby zminimalizować ważoną sumę spóźnień:

$$\min_\pi \sum_{i=1}^{n} w_{\pi(i)} T_{\pi(i)}, \tag{11}$$

gdzie:

- $T_{\pi(i)} = \max\{0, C_{\pi(i)} - d_{\pi(i)}\}$,
- $C_{\pi(i)} = C_{\pi(i-1)} + p_{\pi(i)}$ dla $i > 1$,
- $C_{\pi(1)} = p_{\pi(1)}$,
- $\pi(i)$ określa które w kolejności wykonywania jest zadanie i.

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   2.1. $p_i \leftarrow$ nextInt$(1, 30)$.
   2.2. $w_i \leftarrow$ nextInt$(1, 30)$.
3. $S \leftarrow \sum_{i=1}^{n} p_i$.
4. Dla i od 1 do n:
   4.1. $d_i \leftarrow$ nextInt$(1, S)$.

2.8 Permutacyjny problem przepływowy (Permutation Flow Shop Problem, $PF||C_{\max}$

Dane jest $n \in \mathbb{N}_+$ zadań i $m \in \mathbb{N}_+$ maszyn. Czas wykonywania zadania j na maszynie i wynosi $p_j^i \in \mathbb{N}_+$. Występują ograniczenia:

- zadanie należy zakończyć na maszynie i zanim będzie można je rozpocząć na maszynie $i+1$,
- maszyna może wykonywać co najwyżej jedno zadanie naraz.

Należy określić kolejność wykonywania zadań π tak aby zminimalizować czas zakończenia wszystkich zadań:

$$\min_{\pi} C^{m}_{\pi(n)}, \tag{12}$$

gdzie:

- $C^i_{\pi(j)} = \max\{C^{i-1}_{\pi(j)}, C^i_{\pi(j-1)}\} + p^i_{\pi(j)}$, dla $j > 1 \wedge i > 1$,
- $C^i_{\pi(j)} = C^{i-1}_{\pi(j)} + p^i_{\pi(j)}$, dla $j = 1 \wedge i > 1$,
- $C^i_{\pi(j)} = C^i_{\pi(j-1)} + p^i_{\pi(j)}$, dla $j > 1 \wedge i = 1$,
- $C^i_{\pi(j)} = p^i_{\pi(j)}$, dla $j = 1 \wedge i = 1$,
- $\pi(j)$ określa które w kolejności wykonywania jest zadanie j.

Generacja instancji Dla parametru n i m oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   2.1. Dla j od 1 do m:
      2.1.1. $p^i_j \leftarrow$ nextInt$(1, 99)$.

3 Funkcje ciągłe dwóch zmiennych

1. Himmelblau's function:

$$\begin{aligned} \textbf{minimize} \quad & f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2, \\ & x_1 \in [-5, 5], \quad x_2 \in [-5, 5]. \end{aligned} \tag{13}$$

2. Bukin function N. 6:

$$\begin{aligned} \textbf{minimize} \quad & f(x_1, x_2) = 100\sqrt{|x_2 - 0.01x_1^2|} + 0.01|x_1 + 10|, \\ & x_1 \in [-15, -5], \quad x_2 \in [-3, 3]. \end{aligned} \tag{14}$$

3. Cross-in-Tray function:

$$\begin{aligned} \textbf{minimize} \quad & f(x_1, x_2) = -0.0001\left[\left|\sin(x_1)\sin(x_2)\exp\left(\left|100 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right|\right)\right| + 1\right]^{0.1}, \\ & x_1 \in [-10, 10], \quad x_2 \in [-10, 10]. \end{aligned} \tag{15}$$

4. Eggholder function:

$$\begin{aligned} \textbf{minimize} \quad & f(x_1, x_2) = -(x_2 + 47)\sin\left(\sqrt{\left|\frac{x_1}{2} + (x_2 + 47)\right|}\right) - \\ & - x_1 \sin(\sqrt{|x_1 - (x_2 + 47)|}), \\ & x_1 \in [-512, 512], \quad x_2 \in [-512, 512]. \end{aligned} \tag{16}$$

5. Hölder table function:

$$\begin{aligned} \textbf{minimize} \quad & f(x_1, x_2) = -\left|\sin(x_1)\cos(x_2)\exp\left(\left|1 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right|\right)\right|, \\ & x_1 \in [-10, 10], \quad x_2 \in [-10, 10]. \end{aligned} \tag{17}$$

6. Schaffer function N. 4:

$$\begin{aligned} \textbf{minimize} \quad & f(x_1, x_2) = 0.5 + \frac{\cos^2\left[\sin\left(|x_1^2 - x_2^2|\right)\right] - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2}, \\ & x_1 \in [-100, 100], \quad x_2 \in [-100, 100]. \end{aligned} \tag{18}$$

4 Funkcje ciągłe n zmiennych

1. Styblinski–Tang function:

$$\begin{aligned} \textbf{minimize} \quad & f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_n) = \frac{1}{2}\sum_{i=1}^{n}(x_i^4 - 16x_i^2 + 5x_i), \\ & x_i \in [-5, 5], \quad \text{dla } i = 1, 2, \ldots, n. \end{aligned} \tag{19}$$

2. Rastrigin function:

$$\begin{aligned} \textbf{minimize} \quad & f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_n) = 10n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)], \\ & x_i \in [-5.12, 5.12], \quad \text{dla } i = 1, 2, \ldots, n. \end{aligned} \tag{20}$$

3. Rosenbrock function:

$$\textbf{minimize} \quad f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n-1} \left[100\ (x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right], \tag{21}$$
$$x_i \in [-100, 100], \quad \text{dla } i = 1, 2, \ldots, n.$$

4. Sphere function:

$$\textbf{minimize} \quad f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} x_i^2, \tag{22}$$
$$x_i \in [-100, 100], \quad \text{dla } i = 1, 2, \ldots, n.$$

5 Problemy ciągłe bazowane na problemach dyskretnych

5.1 Problem inspekcji

Problem inspekcji rozpatrujemy jako pewną modyfikację dyskretnego problemu komiwojażera. Dane jest $n \in \mathbb{N}_+$ kół tak że koło i-te przedstawione jest w postaci trójki (a_i, b_i, r_i), gdzie $a_i \in \mathbb{R}$ oraz $b_i \in \mathbb{R}$ to współrzędne koła, zaś $r_i \in \mathbb{R}$ to jego promień. Oprócz tego dany jest punkt startowy $(x_0, y_0) \in \mathbb{R}^2$. Jako rozwiązanie należy podać n punktów od (x_1, y_1) do (x_n, y_n) tak aby:

1. Punkt (x_i, y_i) znajdował się wewnątrz (lub na brzegu) koła i-tego (dla $i = 1, 2, \ldots n$).
2. Całkowita długość trasy (cyklu) pomiędzy punktami $0 \to 1 \to 2 \to \ldots \to n \to 0$ była jak najmniejsza.

Problem ten modeluje sytuację, gdy dron startuje z bazy i ma za zadanie zrobić zdjęcia wszystkich pojazdów na parkingu, a do zrobienia zdjęcia potrzebuje znaleźć się odpowiednio blisko pojazdu).

Uwaga: domyślnym założeniem jest to że odległość między punktami jest liczona jako odległość euklidesowa (tzw. norma ℓ_2). Akceptowane są jednak również rozwiązania gdzie zamiast tego zostanie wykorzystana norma Manhattan (ℓ_1) lub norma Czebyszewa (ℓ_∞).

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   2.1. $a_i \leftarrow$ nextFloat$(5, 35)$.
   2.2. $b_i \leftarrow$ nextFloat$(5, 35)$.
   2.3. $r_i \leftarrow$ nextFloat$(1, 4)$.
3. $x_0 \leftarrow$ nextFloat$(5, 35)$.
4. $y_0 \leftarrow$ nextFloat$(5, 35)$.

5.2 Kwadratowe zagadnienie przydziału

Problem ten przedstawiamy jako pewną modyfikację dyskretnego kwadratowego zagadnienia przydziału. Dane jest n zakładów. Każdy zakład i ma dane koło określone przez trójkę (a_i, b_i, r_i), gdzie $a_i \in \mathbb{R}$ oraz $b_i \in \mathbb{R}$ to współrzędne koła, a $r_i \in \mathbb{R}$ to jego promień. Ponadto dla każdej pary (i, j) zakładów dany jest przepływ $f_{i,j} \in \mathbb{R}$ (przy czym dla $i = j$ zakładamy $f_{i,j} = 0$). Jako rozwiązanie należy podać n punktów od (x_1, y_1) do (x_n, y_n) tak aby:

1. Punkt (x_i, y_i) znajdował się wewnątrz (lub na brzegu) koła i-tego (dla $i = 1, 2, \ldots n$).
2. Wartość wyrażenia:
$$\sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} \cdot d_{i,j} \tag{23}$$
była najmniejsza z możliwych, gdzie $d_{i,j}$ jest odległością punktów i oraz j.

Uwaga: domyślnym założeniem jest to że odległość między punktami jest liczona jako odległość euklidesowa (tzw. norma ℓ_2). Akceptowane są jednak również rozwiązania gdzie zamiast tego zostanie wykorzystana norma Manhattan (ℓ_1) lub norma Czebyszewa (ℓ_∞).

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   - 2.1. $a_i \leftarrow$ nextFloat$(5, 35)$.
   - 2.2. $b_i \leftarrow$ nextFloat$(5, 35)$.
   - 2.3. $r_i \leftarrow$ nextFloat$(1, 7)$.
3. Dla i od 1 do n:
   - 3.1. Dla j od 1 do n:
     - 3.1.1. $f_{i,j} \leftarrow$ nextFloat$(1, 20)$.

5.3 Problem dwuplecakowy

Zadanie to jest modyfikacją dyskretnego problemu plecakowego. Dane jest $n \in \mathbb{N}_+$ przedmiotów. Przedmiot i opisany jest przez trójkę (c_i, w_i, v_i), gdzie $c_i \in \mathbb{R}$ oznacza wartość przedmiotu, zaś $w_i \in \mathbb{R}$ oraz $v_i \in \mathbb{R}$ oznaczają ile miejsca przedmiot zająłby, gdyby w całości umieścić go w odpowiednio pierwszym i drugim plecaku. Oprócz tego dana jest wartość B będąca pojemnością plecaków (oba mają taką samą). Należy dla każdego przedmiotu i określić dwie wartości ciągłe (x_i, y_i), gdzie $x_i \in [0, 1]$ określa jaką część przedmiotu umieszczamy w plecaku pierwszym, zaś $y_i \in [0, 1]$ określa jaką część umieszczamy w plecaku drugim, tak aby:

1. $x_i + y_i \leqslant 1$ (dla $i = 1, 2, \ldots, n$),
2. $\sum_{i=1}^{n} x_i w_i \leqslant B$,
3. $\sum_{i=1}^{n} y_i v_i \leqslant B$,
4. Wartość $\sum_{i=1}^{n} (x_i + y_i) c_i$ była maksymalna.

Generacja instancji Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   2.1. $c_i \leftarrow \text{nextFloat}(1, 10)$.
   2.2. $w_i \leftarrow \text{nextFloat}(1, 10)$.
   2.3. $v_i \leftarrow \text{nextFloat}(1, 10)$.
3. $B \leftarrow \text{nextFloat}(n, 4n)$.

5.4 Problem szeregowania na równoległych maszynach

Dane jest $n \in \mathbb{N}_+$ zadań oraz $m \in \mathbb{N}_+$ maszyn. Dla każdego zadania i dany jest jego bazowy czas wykonywania $p_i \in \mathbb{R}$ oraz maszyna na której go uszeregowano $a_i \in \{1, 2, \ldots, m\}$. Oprócz tego dana jest również łączna szybkość wszystkich maszyn $S \in \mathbb{R}$. Należy określić wartości od s_1 do s_m, gdzie s_i to szybkość maszyny i, tak aby:

1. $s_i > 0$ (dla $i = 1, 2, \ldots, m$),
2. $\sum_{i=1}^{m} s_i = S$,
3. Czas zakończenia pracy maszyny która pracuje najdłużej był jak najmniejszy (tj. minimalizacja „maksymalnej” maszyny).

Faktyczny czas wykonania zadania i to $p_i s_{a_i}$ (czas bazowy razy szybkość maszyny na której go zaszeregowano). Zakładamy że maszyna wykonuje na raz co najwyżej jedno zadanie.

Generacja instancji Dla parametrów n i m oraz ziarna Z:

1. init(Z).
2. Dla i od 1 do n:
   2.1. $p_i \leftarrow \text{nextFloat}(1, 20)$.
   2.2. $a_i \leftarrow \text{nextInt}(1, m)$.
3. $S \leftarrow \text{nextFloat}(m, 2m)$.