Metody optymalizacji. Lista 2

Piotr Berezowski, 236749

3 maja 2020

1 Zadanie 1

1.1 Opis zadania

Opisany w zadaniu problem należy sformułować w postaci zadania programowania całkowitoliczbowego. Następnie korzystając pakietu JuMP rozwiązać problem przy użyciu solvera GLPK lub Cbc.

Chcemy zebrać dane liczbowe na temat m różnych cech populacji. Dane zapisane są w chmurze w n różnych miejscach. Czas potrzebny na przeszukanie i-tego miejsca jest równy T_i. Zakładamy, że czas ten nie zależy od liczby odczytywanych cech. Dane dotyczące każdej z cech mogą znajdować się w kilku miejscach. Chcemy wyznaczyć zbiór miejsc, które trzeba przeszukać, aby zebrać informacje o wszystkich cechach jednocześnie minimalizując całkowity czas potrzebny na odczytanie tych cech.

1.2 Model

Zdefiniujmy zbiór n serwerów jako $S = \{1, 2, \ldots, m\}$, oraz zbiór m cech jako $C = \{1, 2, \ldots, n\}$. Niech $T_j,\ j \in S$, oznacza czas potrzebny na przeszukanie j-tego serwera. Niech $q_{ij},\ i \in C,\ j \in S$ przyjmuje wartość 1 jeśli dane dotyczące i-tej cechy znajdują się na j-tym serwerze. W przeciwnym przypadku $q_{ij} = 0$.

1.2.1 Zmienne decyzyjne

Zmienne decyzyjne określające które serwery należy przeszukać definiujemy jako:

$$x_j = \begin{cases} 1, & \text{jeśli serwer } j \text{ będzie przeszukany} \\ 0, & \text{w przeciwnym przypadku} \end{cases}, \text{ gdzie } j \in S$$

1.2.2 Ograniczenia

Ograniczenia modelują następujące sytuacje:

- Należy odczytać informacje dotyczące wszystkich cech:

$$\sum_{j \in S} x_j q_{ij} \geqslant 1,\ i \in C$$

- Binarność zmeinnych decyzyjnych:

$$x_j \in \{0, 1\}, j \in S$$

1.2.3 Funkcja celu

Funkcja celu przyjmuje postać:

$$\sum_{j \in S} x_j T_j$$

1.3 Rozwiązanie

Implementacja modelu wraz z przykładowymi danymi znajduje się w pliku *zad1.jl*.

Rozwiązano problem dla następujących danych:

- Ilość cech: 7
- Ilość serwerów: 6
- Czasy odczytu dla kolejnych serwerów: $1, 2, 5, 2, 4, 10$
- Wartości q_{ij}:

	1	**2**	**3**	**4**	**5**	**6**	**S**
1	1	0	0	1	0	0	
2	0	1	0	0	1	0	
3	0	0	1	0	0	1	
4	1	0	0	1	0	0	
5	0	1	0	0	1	0	
6	0	0	1	0	0	1	
7	0	0	0	0	0	1	
C							

Tabela 1: Macierz q

Dla podanych danych minimalny czas przeszukiwania serwerów jest równy 13. Należy przeszukać serwery: $1, 2$ i 6.

2 Zadanie 2

2.1 Opis zadania

Opisany w zadaniu problem należy sformułować w postaci zadania programowania całkowitoliczbowego. Następnie korzystając pakietu JuMP rozwiązać problem przy użyciu solvera GLPK lub Cbc.

Niech P_{ij} będzie j-tym podprogramem obliczania funkcji i należącym do biblioteki podprogramów ($i \in \{1, \ldots, m\},\ j \in \{1, \ldots, n\}$). Podprogram P_{ij} zajmuje r_{ij} komórek pamięci i potrzeba na jego wykonanie t_{ij} jednostek czasu. Dla zadanego zbioru funkcji $I,\ I \subseteq \{1, \ldots, m\}$ należy ułożyć program P obliczający zbiór tych funkcji, w taki sposób, żeby zajmował on nie więcej niż M komórek pamięci, oraz jego czas wykonania był minimalny.

2.2 Model

2.2.1 Zmienne decyzyjne

- Zmienne określające czy podprogram P_{ij} wchodzi w skład programu P:

$$x_{ij} = \begin{cases} 1, & \text{jeśli podprogram } P_{ij} \text{ należy do programu } P \\ 0, & \text{w przeciwnym przypadku} \end{cases}, \; i \in \{1, \ldots, m\}, \; j \in \{1, \ldots, n\}$$

2.2.2 Ograniczenia

Ograniczenia modelują następujące sytuacje:

- Wybieramy tylko jeden podprogram dla każdej funkcji obliczanej w P:

$$\sum_{j \in \{1,\ldots,n\}} x_{ij} = 1, \; i \in I$$

- Maksymalne zużycie pamięci nie może przekroczyć wartości M:

$$\sum_{i \in \{1,\ldots,m\}} \sum_{j \in \{1,\ldots,n\}} x_{ij} r_{ij} \leqslant M$$

- Binarność zmiennych decyzyjnych:

$$x_{ij} \in \{0, 1\}, \; i \in \{1, \ldots, m\}, \; j \in \{1, \ldots, n\}$$

2.2.3 Funkcja celu

Funkcja celu przyjmuje postać:

$$\sum_{i \in \{1,\ldots,m\}} \sum_{j \in \{1,\ldots,n\}} x_{ij} t_{ij}$$

2.3 Rozwiązanie

Implementacja modelu wraz z przykładowymi danymi znajduje się w pliku *zad2.jl*.

Rozwiązano problem dla następujących danych:

- Ilość funkcji w bibliotece: $m = 5$
- Ilość podprogramów obliczających poszczególne funkcje: $n = 4$
- Wartości r_{ij}:

	1	**2**	**3**	**4**
1	10	1	1	2
2	2	2	3	1
3	2	3	1	1
4	2	3	1	1
5	2	3	1	1

Tabela 2: Wartości $r_{ij},\ i \in \{1,\ldots,5\},\ j \in \{1,\ldots,4\}$

- Wartości t_{ij}:

	1	**2**	**3**	**4**
1	1	4	5	2
2	1	2	2	5
3	4	2	5	6
4	4	2	5	6
5	4	2	5	6

Tabela 3: Wartości $t_{ij},\ i \in \{1,\ldots,5\},\ j \in \{1,\ldots,4\}$

- Maksymalna dostępna ilość pamięci dla programu: $M = 10$
- Zbiór funkcji z których składa się program P: $I = \{1, 2, 3, 4, 5\}$

Dla podanych danych otrzymano rozwiązanie:

Funkcja	**Podprogram**
1	4
2	1
3	2
4	1
5	3

Tabela 4: Rozwiązanie zadania. Całkowity czas programu jest równy 14.

3 Zadanie 3

3.1 Opis zadania

Opisany w zadaniu problem należy sformułować w postaci zadania programowania całkowitoliczbowego. Następnie korzystając z pakietu JuMP rozwiązać problem przy użyciu solvera GLPK lub Cbc.

Dany jest zbiór zadań $Z = 1, \ldots, n$, które mają być wykonywane na trzech procesorach P_1, P_2 i P_3. Zakładamy, że:

- każdy procesor może wykonywać w danym momencie tylko jedno zadanie,
- każde zadanie musi być wykonywana najpierw na procesorze P_1 następnie na procesorze P_2 i na końcu na procesorze P_3,
- kolejność wykonywania zadań na wszystkich trzech procesorach jest taka sama.

Dla każdego zadania $i \in Z$ są zadane czasy trwania a_i, b_i oraz c_i odpowiednio na procesorach P_1, P_2 i P_3. Wszystkie dane są dodatnimi liczbami całkowitymi. Każdy harmonogram jest jednoznacznie określony przez pewną permutację $\pi = (\pi(1), \ldots, \pi(n))$ zadań należących do zbioru Z. Niech $C_{\pi(k)}$ oznacza czas zakończenia k-go zadania na procesorze P_3 dla permutacji π. Celem jest wyznaczenie permutacji π takiej, że:

$$C_{\max} = C_{\pi(n)} \to \min$$

.

3.2 Model

Niech $P = \{P_1, P_2, P_3\}$ będzie zbiorem procesorów. Niech t_{pj} określa czas trwania zadania j na procesorze p, wtedy $t_{P_1 j} = a_j$, $t_{P_2 j} = b_j$ oraz $t_{P_3 j} = c_j$. Niech B będzie bardzo dużą liczbą.

3.2.1 Zmienne decyzyjne

Wprowadźmy następujące zmienne:

- Zmienne określające moment rozpoczęcia j-tego zadania na i-tym procesorze:
$$x_{ij} \geqslant 0,\ i \in P,\ j \in Z$$
- Zmienne pomocnicze określające czy zadanie i jest wykonywane przed zadaniem j:
$$y_{ij} = \begin{cases} 1, & \text{jeśli zadanie } i \text{ jest wykonywane przed zadaniem } j \\ 0, & \text{w przeciwnym przypadku} \end{cases}, i \in Z,\ j \in Z,\ i \neq j$$
- Zmienna pomocnicza modelująca moment zakończenia ostatniego zadania na procesorze P_3:
$$M \geqslant 0$$

3.2.2 Ograniczenia

Ograniczenia modelują następujące sytuacje:

- Zachowanie kolejności procesorów - zadanie musi zostać wykonane na poprzednim procesorze, aby mogło być liczone na kolejnym:

$$x_{P_1 j} + t_{P_1 j} \leqslant x_{P_2 j},\ j \in Z,$$
$$x_{P_2 j} + t_{P_2 j} \leqslant x_{P_3 j},\ j \in Z$$

- Zadania nie nakładają się - każdy procesor może obliczać jedno zadanie w tym samym czasie:

$$x_{pi} - x_{pj} + By_{ij} \geqslant t_{pj},\ p \in P,\ i \in Z,\ j \in Z$$
$$x_{pj} - x_{pi} + B(1 - y_{ij}) \geqslant t_{pi},\ p \in P,\ i \in Z,\ j \in Z$$

- Ograniczenia modelujące wartość zmiennej $M = \max_{j \in Z} x_{P_3 j} + t_{P_3 j}$:

$$x_{P_3 j} + d_{P_3 j} \leqslant M,\ j \in Z$$

- Binarność zmiennych pomocniczych y_{ij}:

$$y_{ij} \in \{0, 1\},\ i \in Z,\ j \in Z,\ i \neq j$$

3.2.3 Funkcja celu

Funkcja celu przyjmuje postać:

$$M$$

3.3 Rozwiązanie

Implementacja modelu wraz z przykładowymi danymi znajduje się w pliku *zad3.jl*.

Rozwiązano problem dla następujących danych:

- Ilość zadań: $n = 5$
- Czasy wykonania zadań na poszczególnych procesorach:

	1	**2**	**3**	**4**	**5**	**Z**
P_1	1	2	3	4	5	
P_2	2	3	4	3	2	
P_3	1	2	3	1	2	
P						

Tabela 5: Wartości $t_{ij},\ i \in P,\ j \in Z$

Dla podanych danych otrzymano rozwiązanie: $\pi = (2, 4, 3, 1, 5)$