Metody Optymalizacji, Lista 2, rozwiązania

Błażej Wróbel, 250070, W4N, 4. rok

Zadanie 1

W zadaniu 1 było danych m różnych cech danej populacji i wszystkie te cechy były zapisane w n różnych miejscach. Przeszukanie miejsca j zajmuje czas T_j gdzie ($j \in \{1 \dots n\}$). Ponadto wystepują duplikaty danych dotyczących niektórych cech tzn. dane dotyczące takiej cechy są zapisane w różnych miejscach. Taka sytuacja jest opisana za pomocą macierzy Q gdzie $q_{ij} = 1$ oznacza, że dane dotyczące cechy i ($i \in \{1 \dots m\}$) są zapisane w miejscu (na serwerze) j ($j \in \{1 \dots m\}$) i $q_{ij} = 0$ w przeciwnym przypadku. Mając te dane, należy wyznaczyć spośród n miejsc te, które należy przeszukać, aby odczytać dane dotyczące wszystkich m cech w jak najmniejszym czasie.

Do rozwiązania zadania wykorzystuję wektor zmiennych decyzyjnych $x \in \{0, 1\}^n$ gdzie każda zmienna decyzyjna x_i ($i \in \{1 \dots n\}$) jest zmienną binarną ($x_i \in \{0, 1\}$ - stąd całkowitoliczbowośc modelu). Jeśli $x_i = 1$ to oznacza, że odczytujemy dane z miejsca (serwera) i, w przeciwnym przypadku ($x_i = 0$) nie odczytujemy danych z miejsca i. W przypadku tego problemu zmienne decyzyjne wyrażają decyzję o odczytaniu danych z i-tego miejsca, zatem nie muszą mieć żadnych jednostek.

Ze sformułowania problemu wynika, że funkcją celu jest łączny czas odczytania danych dotyczących wszystkich m cech. Zatem minimalizowana funkcja celu ma postać:

$$F = \sum_{j=1}^{n} T_j \cdot x_j \to min$$

T_j oznacza czas przeszukania miejsca j ($j \in \{1 \dots n\}$) a x_j oznacza decyzję (tak/nie - 0/1) o przeszukiwaniu danego miejsca. Zatem powyższy wzór wyraża łączny czas przeszukania wybranych miejsc (i z treści problemu ten czas jest minimalizowany).

Ograniczeniem, które wynika ze sformułowania problemu jest to, że należy odczytać dane dotyczące wszystkich m cech. To ograniczenie wyrażamy następująco:

$$\sum_{j=1}^{n} q_{ij} \cdot x_j \geq 1 \quad \forall i \in \{1 \dots m\}$$

Wartość q_{ij} wyraża to, czy dane dotyczące cechy i są zapisane w miejscu j. Jeśli wartość iloczynu $q_{ij} \cdot x_j = 1$ to oznacza, że cecha i została odczytana z miejsca j.

Zatem powyższe ograniczenie wyraża fakt, że każda cecha musi być odczytana co najmniej raz (z dowolnego miejsca, w którym jest zapisana).

Model przetestowałem na następujących danych ($m = 6$ oraz $n = 5$):
Wektor czasów przeszukiwania T:

$$T = \begin{bmatrix} 12 & 9 & 98 & 45 & 10 \end{bmatrix}$$

Macierz z danymi o lokalizacjach (i duplikatach) Q:

$$Q = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Po uruchomieniu rozwiązywania modelu na tych danych otrzymałem następujące rozwiązanie :

$$x^T = \begin{bmatrix} 1.0 & 1.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

Wartość funkcji celu wynosi $F = 21.0$. Widać, że otrzymane rozwiązanie jest dopuszczalne, ponieważ odczytaliśmy dane dotyczące wszystkich m cech oraz wybraliśmy miejsca, których czasy przeszukiwania są najmniejsze, zatem rozwiązanie jest również optymalne.

Zadanie 2

W zadaniu 2 mieliśmy dane m funkcji i n podprogramów (dla każdej funkcji), które służą do obliczania tych funkcji. Zatem j-ty podprogram do obliczania i-tej funkcji oznaczamy jako P_{ij}. Każdy podprogram P_{ij} zajmuje pewną liczbę komórek pamięci r_{ij} oraz czas potrzebny na wykonanie tego podprogramu wynosi t_{ij}. Problemem jest ułożenie programu sekwencyjnego, który oblicza zadany zbiór funkcji $I \subseteq \{1 \dots m\}$ tak, aby zajmował nie więcej niż M komórek pamięci oraz czas jego wykonania był minimalny.

Zmienne decyzyjne reprezentuję jako macierz $X \in \{0,1\}^{m \times n}$ ($i \in \{1 \dots m\}$ oraz $j \in \{1 \dots n\}$), gdzie każda zmienna decyzyjna $x_{ij} \in \{0,1\}$ jest zmienną binarną (stąd całkowitoliczbowość modelu). Jeśli $x_{ij} = 1$, to do obliczenia funkcji i użyje podprogramu j, w przeciwnym przypadku nie używam podprogramu j. W tym przypadku zmienne decyzyjne wyrażają decyzję o obliczeniu funkcji i za pomocą podprogramu j, zatem nie muszą przyjmować żadnych jednostek.

Ze sformułowania problemu wynika, że minimalizowaną funkcją celu jest czas działania ułożonego programu sekwencyjnego P. Zatem minimalizowana funkcja celu ma postać:

$$F = \sum_{i \in I} \sum_{j=1}^{n} t_{ij} x_{ij} \to min$$

Z tej postaci wynika, że funkcja celu wyraża czas wykonania programu P przez wybrane podprogramy P_{ij} (przy czym używamy tylko podprogramów do obliczania funkcji ze zbioru I).

Pierwszym ograniczeniem wynikającym bezpośrednio ze sformułowania problemu jest to, że ułożony program P nie może zajmować więcej niż M komórek pamięci. Ten fakt zapiszemy następująco:

$$\sum_{i \in I} \sum_{j=1}^{n} x_{ij} \cdot r_{ij} \leq M$$

To oznacza, że suma zużywanej pamięci przez wybrane podprogramy, które służą od obliczania funkcji ze zbioru I, nie może przekroczyć zadanego M. Innym ograniczeniem jest to, że interesują nas tylko podprogramy do obliczania funkcji z I i wybieramy dokładnie jeden taki podprogram. To możemy wyrazić następująco:

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall i \in I$$

Warto również dodać ograniczenie, że nie interesują nas podprogramy służące do obliczania funkcji spoza zbioru I (czyli ze zbioru $I' = \{1 \dots m\} \setminus I$):

$$\sum_{j=1}^{n} x_{ij} = 0 \quad \forall i \in I'$$

Powyższe ograniczenie wyraża fakt, że nie bierzemy żadnego podprogamu do obliczania funkcji ze zbioru I'.

Model przetestowałem na następujących danych ($m = 5$, $n = 6$, $M = 100$): Zbiór I:

$$I = \{1, 3, 5\}$$

Macierz z czasami wykonania podprogramów t:

$$t = \begin{bmatrix} 12 & 30 & 20 & 45 & 60 & 10 \\ 10 & 10 & 56 & 78 & 90 & 9 \\ 25 & 45 & 12 & 34 & 67 & 70 \\ 23 & 43 & 12 & 21 & 32 & 45 \\ 23 & 65 & 76 & 68 & 56 & 76 \end{bmatrix}$$

Macierz r wymagań pamięciowych dla poszczególnych podprogramów:

$$r = \begin{bmatrix} 34 & 51 & 21 & 14 & 15 & 17 \\ 20 & 30 & 40 & 10 & 21 & 25 \\ 56 & 12 & 32 & 45 & 61 & 12 \\ 34 & 43 & 23 & 65 & 12 & 9 \\ 33 & 44 & 23 & 65 & 78 & 14 \end{bmatrix}$$

Po uruchomieniu rozwiązywania modelu dla tych danych, otrzymałem następujące rozwiązanie:

$$X = \begin{bmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

Zatem do obliczenia funkcji 1 użyłem podprogramu 6, do obliczenia funkcji 3 podprogramu 3 i do obliczenia funkcji 5 podprogramu 1. Widać, że używam tylko podprogramów do obliczania funkcji z zadanego zbioru I oraz każdy podprogram został wybrany dokładnie raz. Bazując na rozwiązaniu i macierzy r można zauważyć, że koszt pamięciowy P wynosi 82, zatem jest mniejszy niż M. Wartość funkcji celu (czas wykonania P) wynosi 45.0 (bazując na rozwiązaniu i macierzy t można zauważyć, że ta wartość jest poprawna).

Zadanie 3

W zadaniu 3 mamy dane procesory P_1, P_2, P_3 (m - liczba procesorów) oraz zbiór zadań $Z = \{1 \dots n\}$. Każdy procesor w danym momencie może wykonywać tylko jedno zadanie, każde zadanie najpierw musi być wykonane na procesorze P_1 potem na P_2 i później na P_3 oraz kolejność wykonywania zadań na wszystkich procesorach jest taka sama. Dla każdego zadania z $Z = \{1 \dots n\}$ mamy podany czas jego trwania a_i, b_i oraz c_i na odpowiednio procesorze P_1, P_2 i P_3 (te czasy mogą być przechowywane w macierzy $d \in \mathbb{R}^{n \times m}$). Problemem jest znalezienie takiej kolejności zadań (harmonogramu, permutacji $\pi = (\pi(1), \pi(2) \dots \pi(n))$), że czas zakończenia ostatniego zadania na procesorze P_3 jest jak najmniejszy.

Zmienne decyzyjne reprezentuje za pomocą macierzy $S \in \mathbb{R}^{n \times m}$, gdzie wartość S_{ij} oznacza moment rozpoczęcia zadania i na maszynie j. Oczywiście $S_{ij} \geq 0$ dla każdego $i \in \{1 \dots m\}$ oraz $j \in \{1 \dots n\}$. Zakładając, że czasy trwania zadań na poszczególnych maszynach są podane w sekundach można przyjąć, że jednostkami w tym przypadku są również sekundy. Oprócz tego wprowadzam zmienne binarne $p_{jik} \in \{0, 1\}$ (stąd całkowitoliczbowość modelu) dla $j \in \{1 \dots m\}$, $k \in \{1 \dots n\}$ oraz $i < k$, które wyrażają fakt, że zadanie k zostaje wykonane przed zadaniem i na maszynie j. Te zmienne ułatwią nam wyrażenie ograniczenia, że każdy procesor w danej chwili wykonuje tylko jedno zadanie.

W tym przypadku minimalizowaną funkcją celu jest moment zakończenia ostatniego zadania na ostatnim procesorze (P_3). Niech $C_{\pi(k)}$ oznacza moment

zakończenia k-tego zadania na procesorze P_3. Wówczas funkcję celu możemy zapisać następująco:

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

Pierwszym ograniczeniem, które zapiszemy, jest ograniczenie kolejnościowe, czyli każde zadanie jest najpierw wykonywane na P_1 potem na P_2 i później na P_3. Ten fakt możemy wyrazić następująco:

$$\forall i \in \{1 \dots n\} \quad \forall j \in \{1 \dots m-1\} \quad S_{ij} + d_{ij} \leq S_{i(j+1)}$$

Następnie wyrazimy fakt, że każdy procesor w danym momencie może wykonywać tylko jedno zadanie. Jeśli wiemy, że zadanie k jest wykonywane przed zadaniem i na maszynie j, to moment rozpoczęcia zadania i, musi być większy równy niż moment zakończenia zadania k (jeśli by to nie zaszło, to procesor wykonywałby 2 czynności na raz). To ograniczenie możemy wyrazić następująco:

$$S_{ij} - S_{kj} + B \cdot p_{jik} \geq d_{kj}$$

Może być również na odwrót, czyli zadanie i jest wykonywane przed zadaniem k. Wówczas musimy zapisać podobne ograniczenie:

$$S_{kj} - S_{ij} + B \cdot (1 - p_{jik}) \geq d_{ij}$$

Ze względu na to, że mogą się zdarzyć ten przypadek lub poprzedni, muszę w jakiś sposób wyrazić altertnatywę tych warunków. Do tego służy zmienna B (tzw. duża liczba), które zapewnia, że niezależnie od przypadku, wszystkie ograniczenia są spełnione. Wartość zmiennej B w tym przypadku jest równa sumie wszystkich czasów wykonywania wszystkich zadań (suma elementów macierzy d) plus jeden.

Ostatnim ograniczeniem jest fakt, że czas końca całego procesu (czyli moment kiedy P_3 ukończył ostatnie zadanie) nie może być mniejszy niż wartość funkcji celu (m - liczba procesorów):

$$\forall i \in \{1 \dots n\} \quad S_{im} + d_{im} \leq C_{\pi(n)}$$

Model przetestowałem na następujących danych ($m = 3$ oraz $n = 4$):
Macierz czasów wykonania zadań na poszczególnych procesorach:

$$d = \begin{bmatrix} 3 & 3 & 2 \\ 9 & 3 & 8 \\ 9 & 8 & 5 \\ 4 & 8 & 4 \end{bmatrix}$$

Po uruchomieniu rozwiązywania modelu otrzymałem następujące rozwiązanie, zwizualizowane za pomocą diagramu Gantta:

$P1$: 444433333333311122222222##########
$P2$: #####4444444333333333#111222########
$P3$: ############4444####333331122222222

Liczba powtórzeń danego numeru oznacza czas trwania zadania (o tym samym numerze) na danym procesorze. Znak # oznacza, że procesor jest w stanie bezczynności. Widać, że otrzymane rozwiązanie jest dopuszczalne, ponieważ jest zachowana kolejność na każdym procesorze, każdy procesor w danym momencie wykonuje jedno zadanie oraz każde zadanie jest wykonane najpierw na P_1 potem na P_2 a później na P_3. Wartość funkcji celu wynosi 36.0.