Sprawozdanie Metody lista 2

Wojciech Kowalik

Maj 2020

1 Problem 1

Dane jest n serwerów przechowujących dane o cechach populacji. Cech populacji jest m. Niech T_j, dla $j = 1, \ldots, n$, będzie czasem potrzebnym do odczytania wszystkich potrzebnych danych z serwera j. Niech g_{ij}, dla $i = 1, \ldots, m$, $j = 1, \ldots, n$, będzie binarnym parametrem oznaczającym dostępność danej i, na serwerze j. Problem polegał na znalezieniu takiego podzbioru indeksów $1, \ldots, n$, który znajdowałby wszystkie dane i minimalizował przy tym czas potrzebny na uzyskanie tych danych.

1.1 Opis rozwiązania

Rozwiązanie składa się z x_j, $j = 1, \ldots, n$, binarnych zmiennych decyzyjnych, które przedstawiają, czy z serwera j zostały pobrane dane. W takim wypadku dla każdego $i = 1, \ldots, m$ muszą zostać nałożone następujące ograniczenia:

$$\sum_{j=1}^{n} q_{ij} x_j \geqslant 1$$

Te ograniczenia mówią, że dane cechy zostały pobrane z przynajmniej jednego serwera.

Funkcja celu prezentuje się następująco:

$$\min \sum_{j=1}^{n} \sum_{i=1}^{m} T_j q_{ij}$$

1.2 Przykładowy egzemplarz

Przy pomocy powyższego podejścia został rozwiązany następujący egzemplarz. Dane jest 8 serwerów i 6 cech populacji do pobrania.

$$T = [2, 4, 5, 1, 2, 3, 6, 25]$$

$$q_{ij} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Program wyliczył, że trzeba pobrać dane z serwerów 2, 3 i 5, aby uzyskać wszystkie dane w czasie 11.

2 Problem 2

Drugi problem dotyczy wyboru podprogramów do wyliczania różnych funkcji. W tym przypadku P_{ij} oznacza j-ty podprogram do obliczania funkcji i, $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$. Analogicznie r_{ij} oznacza ilość komórek pamięci potrzebnych do wykonania tego podprogramu, a t_{ij} oznacza czas potrzebny na jego wykonanie. Celem jest znalezienie takiego zbioru podprogramów P_{ij}, które obliczają wszystkie funkcje ze zbioru $I \subseteq \{1, \ldots, m\}$, tak aby zużyć maksymalnie M komórek pamięci i w jak najmniejszym czasie.

2.1 Opis rozwiązania

W celu wyznaczenia podprogramów, potrzebne są binarne zmienne decyzyjne x_{ij}, $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$. Oznaczają one wybór podprogramu do ostatecznego rozwiązania. Funkcja celu w tym przypadku prezentuje się następująco:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} t_{ij}$$

W zależności od tego czy $i \in I$, muszą zostać spełnione następujące ograniczenia:

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

$$\forall_{i \notin I} \sum_{j=1}^{n} x_{ij} = 0$$

Ograniczenie dotyczące pamięci prezentuje się następująco:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} r_{ij} \leqslant M$$

2.2 Przykładowy egzemplarz

Następujący egzemplarz problemu został rozważony. $n = 6$, $m = 5$, $M = 17$, $I = \{1, 2, 4, 6\}$

$$t_{ij} = \begin{pmatrix} 1 & 1 & 1 & 1 & 2 \\ 2 & 5 & 3 & 2 & 1 \\ 3 & 9 & 1 & 1 & 2 \\ 3 & 4 & 6 & 5 & 8 \\ 3 & 3 & 3 & 3 & 3 \\ 6 & 6 & 7 & 6 & 9 \end{pmatrix}$$

$$r_{ij} = \begin{pmatrix} 9 & 9 & 8 & 9 & 7 \\ 3 & 1 & 5 & 8 & 7 \\ 5 & 1 & 8 & 7 & 8 \\ 4 & 4 & 4 & 3 & 1 \\ 7 & 7 & 9 & 6 & 8 \\ 2 & 1 & 2 & 3 & 1 \end{pmatrix}$$

Dla powyższych danych, czas potrzebny na wyliczenie funkcji z I wyniósł 12, z wykorzystaniem 16 komórek pamięci z wykorzystaniem P_{13}, P_{21}, P_{41}, P_{62}.

3 Problem 3

Trzeci problem dotyczy znalezienia kolejności wykonywania zadań na trzech procesorach, która trwa najkrócej. Zadań jest m. Dla każdego zadania i podany jest czas t_{ij} potrzebny na wykonanie go na procesorze j. Każdy procesor może wykonywać na raz tylko jedno zadanie. Każde zadanie musi zostać najpierw wykonane na procesorze o indeksie poprzedzającym. Kolejność wykonywania zadań jest identyczna na wszystkich procesorach.

3.1 Opis rozwiązania

Poniższe rozwiązanie działa $n \geqslant 1$ liczby procesorów.

Niech $s_{ij} \geqslant 0$ będzie naturalną zmienną decyzyjną, oznaczającą moment rozpoczęcia zadania $i \in \{1, \ldots, m\}$ na procesorze $j \in \{1, \ldots, n\}$. W takim przypadku rozwiązanie dopuszczalne musi spełniać następujące ograniczenia, dotyczące kolejności wykonywania zadań na procesorach:

$$\forall_{i \in \{1,\ldots,m\}, j \in \{1,\ldots,n-1\}} s_{ij} t_{ij} \leqslant s_{ij+1}$$

Niech B oznacza bardzo dużą liczbę. Niech y_{ikj}, będzie pomocniczą, binarną zmienną decyzyjną. Wtedy następujące ograniczenia dotyczące możliwości wykonywania tylko jednego zadania w danym czasie muszą być spełnione:

$$\forall_{j \in \{1,\ldots,n\}, k \in \{1,\ldots,m\}, i \in \{1,\ldots,k-1\}} s_{ij} - s_{kj} + y_{ikj} B \geqslant t_{kj}$$

$$\forall_{j \in \{1,\ldots,n\}, k \in \{1,\ldots,m\}, i \in \{1,\ldots,k-1\}} s_{kj} - s_{ij} + (1 - y_{ikj}) B \geqslant t_{ij}$$

Kolejne ograniczenia dotyczą identycznej kolejności wykonywania zadań na procesorach. Niech x_{ikj}, będzie pomocniczą, binarną zmienną decyzyjną. Wtedy ograniczenia wyglądają następująco:

$$\forall_{j\in\{1,\ldots,n-1\},k\in\{1,\ldots,m\},i\in\{1,\ldots,k-1\}} s_{ij} - s_{kj} + x_{ikj}B \geqslant 0$$
$$\forall_{j\in\{1,\ldots,n-1\},k\in\{1,\ldots,m\},i\in\{1,\ldots,k-1\}} s_{ij+1} - s_{kj+1} + x_{ikj}B \geqslant 0$$
$$\forall_{j\in\{1,\ldots,n-1\},k\in\{1,\ldots,m\},i\in\{1,\ldots,k-1\}} s_{kj} - s_{ij} + (1 - x_{ikj})B \geqslant 0$$
$$\forall_{j\in\{1,\ldots,n-1\},k\in\{1,\ldots,m\},i\in\{1,\ldots,k-1\}} s_{kj+1} - s_{ij+1} + (1 - x_{ikj})B \geqslant 0$$

Binarne zmienne x i y mają zapewnić, że tylko jedno z ograniczeń jest spełnione, podczas, gdy drugie potrzebuje dodania B, aby było prawdziwe.

Mając powyższe ograniczenia, wciąż potrzeba dodania jeszcze jednego. Niech $e \geqslant 0$ będzie naturalną zmienną decyzyjną oznaczającą czas zakończenia ostatniego zadania. Musi ona spełniać:

$$\forall_{i\in\{1,\ldots,m\}} s_{in} + t_{in} \leqslant e$$

Wtedy funkcja celu to po prostu:

$$\min e$$

3.2 Przykładowy egzemplarz

Niech zadań będzie 6 na 3 procesorach. Niech macierz t wygląda następująco:

$$t_{ij} = \begin{pmatrix} 1 & 2 & 1 \\ 5 & 6 & 4 \\ 1 & 4 & 4 \\ 3 & 4 & 4 \\ 6 & 7 & 4 \\ 3 & 7 & 3 \end{pmatrix}$$

Wtedy solver GLPK zwraca następującą permutacje $\pi = (3, 4, 2, 5, 6, 1)$, która zajmuje 33 jednostki czasu do wykonania na 3 procesorach.

4 Problem 4

W problemie 4 należy znaleźć optymalny harmonogram wykonywania zadań. Zadań jest n. Każde zadanie ma określony czas wykonywania t_j. Do wykonywania zadań potrzebne są odnawialne zasoby R_i, $i = 1, \ldots, p$. Na każdy zasób nałożone są limity N_i, które odnawiają się w każdej jednostce czasu. Każde zadanie ma wartość $r_i j$ przechowującą zapotrzebowanie na dany zasób. Dodatkowo, niektóre zadania mogą być wykonane, dopiero po zakończeniu innych zadań. Dla każdego zadania j, $j = 1, \ldots, n$ te informacje przechowywane są w zbiorze $P_j \subset \{1, \ldots, n\}$.

4.1 Opis rozwiązania

Do zapisania rozwiązania końcowego służy n binarnych zmiennych decyzyjnych s_j, które przechowują moment rozpoczęcia danego zadania. Zmienna e oznacza moment zakończenia ostatniego z zadań. Jest ona wykorzystywana w funkcji celu:

$$\min e$$

Gdzie e jest dodatkowo ograniczona przez:

$$\forall_{i=1}^{n} s_i t_i \leqslant e$$

Poprzedzanie zadań zostało rozwiązane poprzez następujące ograniczenia:

$$\forall_{i=1}^{n} \forall_{j \in P_i} s_i \geqslant s_j + t_j$$

Następnie ograniczenia zasobowe zostały rozwiązane następująco. Czas został podzielony na pojedyncze jednostki czasowe od 1 do $T_m = \sum_{i=1}^{m} t_i$. Zostały wprowadzone trzy grupy binarnych zmiennych decyzyjnych, x_{it}, y_{it}, z_{it}, $i = 1, \ldots, n$, $t = 1, \ldots, T_m$. Zmienne x oznaczają przypadek kiedy w momencie t dane zadanie już się rozpoczęło. Zmienne y oznaczają przypadek, w którym dane zadanie się jeszcze nie rozpoczęło. Zmienne z przedstawiają się następująco: $z_{it} = x_{it} \wedge y_{it}$. Zapisane w ramach programowania liniowego prezentuje się to następująco:

$$\forall_{t=1}^{T_m} \forall_{i=1}^{n} s_i - t + x_{it} * T_m \geqslant 0$$
$$\forall_{t=1}^{T_m} \forall_{i=1}^{n} s_i + t_i - t - y_{it} * T_m \leqslant -1$$
$$\forall_{t=1}^{T_m} \forall_{i=1}^{n} y_{it} + x_{it} - z_{it} \leqslant 1$$

Mając te pomocnicze zmienne można zapisać warunki, aby nie można było przekraczać ustalonych limitów zasobów w każdej jednostce czasu:

$$\forall_{t=1}^{T_m} \forall_{r=1}^{p} \sum_{i=1}^{n} z_{it} r_{ir} \leqslant N_r$$

4.2 Przykładowy egzemplarz

Przy tym problemie został przedstawiony przykładowy egzemplarz, który został rozwiązany powyższą metodą. Następujący harmonogram został wybrano jako optymalny:

t	1	51	53	97	108	144	150	165	176	238
task	1	4	3	2	2	5	5	5	8	-
task	-	-	4	3	6	6	6	-	-	-
task	-	-	-	-	-	-	7	-	-	-
R_1	9	4	15	28	24	20	27	13	17	0

Tutaj t oznacza moment, w którym nastąpiła zmiana w aktualnie wykonywanych zadaniach. Wiersze oznaczone jako "taskóznaczą obecnie wykonywane zadania. R_1 oznacza ile w danym momencie jest wykorzystywanych zasobów nr 1.