Metody Optymalizacji
Lista2

Jakub Gogola
236412

15 maja 2020r.

1 Wstęp

Celem niniejszej listy było rozwiązanie problemów optymalizacyjnych za pomocą biblioteki `JuMP` dla języka `Julia`. Poniżej zostały przedstawione badane problemy w formie krótkiego wstępu teoretycznego, prezentacji wyników wraz z obserwacjami oraz płynących z nich wniosków.

2 Odczyt danych rozproszonych

W **pierwszym** zadaniu z listy należało rozpatrzeć problem odczytu danych dotyczących cech pewnej populacji, które zostały zamieszczone w *chmurze*, na różnych serwerach. Należało zminimalizować czas niezbędny do odczytu wszystkich informacji.

2.1 Model formalny

Przyjmujemy, że dana populacja jest reprezentowana przez zbiór m cech, które są rozmieszczone na n różnych serwerach. Przeszukanie każdego serwera zajmuje określony czas. Niech:

- wektor $\mathbf{T}_n$ zawiera informację o czasie potrzebnym do przeszukania każdego z n serwerów,
- macierz $\mathbf{Q}_{m\times n} = \{q_{ij} \in \{0,1\}\}$ zawiera informacje o rozmieszczeniu informacji o m cechach populacji na n serwerach. Przyjmijmy, że jest to macierz wartości binarnych, gdzie wartość $q_{ij} = 1$ oznacz, że informacja o i-tej cesze jest zapisana na j-tym serwerze, a wartość 0, że serwer nie przechowuje tych danych.

Ponadto, dla uproszczenia zapisu, przyjmijmy, że $M = \{1,\dots,m\}$ oraz $N = \{1,\dots,n\}$.

Zmienna decyzyjna to wektor $\mathbf{S}_n = \{s_i \in \{0,1\}\}$, który zawiera informacje, które serwery należy przeszukać (zapisane binarnie). $s_i = 1$ oznacza, że dany serwer powinien zostać przeszukany, a wartość 0, że dane miejsce nie jest przeszukiwane.

Funkcja celu dla tego problemu prezentuje się następująco:

$$\min \sum_{j\in N} T_j \cdot S_j \tag{1}$$

Celem jest takie wybranie serwerów do przeszukania, aby sumaryczny czas poszukiwań był jak najmniejszy.

Ponadto nałożone zostało ograniczenie, które ma zapewnić, że do odczytania każdej cechy użyto dokładnie jednego serwera:

$$(\forall i \in M)((\sum_{j\in N} Q_{ij} \cdot S_j) = 1) \tag{2}$$

2.2 Wyniki

W celu przetestowania przedstawionego powyżej modelu przeszukiwano informacje o $m = 3$ cechach na $n = 3$ serwerach. Czasy przeszukiwania i miejsca zapisu informacji przedstawiono poniżej:

Serwer	Czas
1	6
2	3
3	2

Cecha / Serwer	**1**	**2**	**3**
1	1	0	1
2	1	0	1
3	0	1	0

Do znalezienia optymalnego rozwiązania użyto solwera `GLPK`. W wyniku działania programu otrzymano następujące wyniki:

Serwer	Czy przeszukać?
1	Nie
2	Tak
3	Tak

Sumaryczny czas przeszukiwania wyniósł 5.

3 Obliczanie zbioru funkcji za pomocą podprogramów

Celem **drugiego** zadania było takie dobranie podprogramów do obliczenia zadanego zbioru funkcji, aby sumaryczny czas ich obliczania był jak najmniejszy.

3.1 Model formalny

Niech P_{ij} będzie j-tym podprogramem, który oblicza i-tą funkcję, należącym do biblioteki podprogramów ($i \in \{1, \ldots, m\}, j \in \{1, \ldots n\}$). Wykonanie każdego programu Pij wymaga zajęcia określonej liczby komórek pamięci oraz każdy podprogram wykonuje się przez pewien, określony czas. Niech:

- $\mathbf{R}_{m\times n}$ będzie macierzą zawierającą informacje o liczbie komórek pamięci alokowanych przez każdy podprogram do obliczenia każdej funkcji,
- $\mathbf{T_{m\times n}}$ będzie macierzą zawierającą informacje o czasie potrzebnym dla każdego podprogramu do obliczenia każdej z funkcji,
- M będzie górnym ograniczeniem na liczbę możliwych do zaalokowania komórek pamięci przez program P.

Zmienna decyzyjna dla tego problemu to macierz $\mathbf{P}_{m\times n}$, która (binarnie) przechowuje dane, który podprogram powinien zostać użyty do użycia której funkcji. Wartość $P_{ij} = 1$ oznacza, że j-ty podprogram, został użyty do obliczenia i-tej funkcji. Wartość 0 oznacza, że do obliczenia i-tej funkcji nie wykorzystano tego podprogramu.

Dla dalszego uproszczeni zapisu przyjmijmy, że $F = \{1, \ldots, m\}$ oraz, że $N = j \in \{1, \ldots n\}$.

Funkcja celu ma za zadanie zminimalizować sumaryczny czas wykonywania programu (złożonego z podprogramów P_{ij}) przy zadanym limicie na maksymalną liczbę zaalokowanych komórek:

$$\min \sum_{i\in F, j in N} T_{ij} \cdot P_{ij} \tag{3}$$

Dla modelu zostały zdefiniowane również ograniczenia. Pierwsze dotyczy faktu, że każda i-ta funkcja powinna zostać obliczona jedynie przez jeden podprogram (ale jeden podprogram może zostać wykorzystany do obliczenia kilku funkcji):

$$(\forall i \in F)(\sum_{j\in N} P_{ij} = 1) \tag{4}$$

Dodatkowo, nie może zostać zaalokowane więcej komórek pamięci niż przewiduje zadany limit M:

$$\sum_{i\in F, j\in N} R_{ij} \cdot P_{ij} \leqslant M \tag{5}$$

3.2 Wyniki

Działanie modelu oraz generowane przezeń rozwiązania zostały przetestowane dla poniżej prezentowanych danych. Limit M na komórki pamięci wyniósł 30 i założono, że do obliczenia $m = 3$ funkcji można wykorzystać $n = 3$ różnych podprogramów o odpowiednio zdefiniowanych wymaganiach pamięciowych oraz czasowych, co zaprezentowano w tabelach.

Funkcja / Podprogram	**1**	**2**	**3**
1	2	30	35
2	10	2	40
3	15	35	45

Wymagania **pamięciowe** dla poszczególnych podprogramów

Funkcja / Podprogram	**1**	**2**	**3**
1	2	8	14
2	4	1	16
3	6	12	18

Wymagania **czasowe** dla poszczególnych podprogramów

W wyniku uruchomienia modelu z solwerem `GLPK` otrzymano wyniki:

Funkcja / Podprogram	**1**	**2**	**3**
1	1	0	0
2	0	1	0
3	1	0	0

Oznacza to, że do obliczenia funkcji 1 i 3 został wykorzystany podprogram 1, a do obliczenia funkcji 2 - podprogram 2. Sumaryczny czas działania wszystkich podprogramów wyniósł 9.

4 Szeregowanie zadań na procesorach

Trzecie zadanie dotyczyło problemu uszeregowania zadań ze zbioru Z na procesorach z zachowaniem odpowiedniej kolejności ich wykonywania.

4.1 Model formalny

Dla problemu należy przyjąć na wstępie następujące założenia:

1. każdy procesor w danym momencie może wykonywać tylko jedno zadanie,
2. każde zadanie musi zostać najpierw wykonane na procesorze P_1, następnie na P_2 i na końcu na P_3,
3. kolejność wykonywania zadań na każdym z procesorów jest taka sama.

Należy wyznaczyć taką permutację π zadań, aby zminimalizować czas ich wykonania, tj. czas wykonania ostatniego zadania z wyznaczonej permutacji na ostatnim procesorze powinien być jak najmniejszy (najwcześniejszy). Niech $\mathbf{T}_{m \times n}$ oznacza macierz zawierającą czasy wykonania zadań ze zbioru Z na poszczególnych procesorach, gdzie m oznacza liczbę zadań do wykonania, a n - liczbę procesorów (w tym przypadku rozważamy $n = 3$). Dla zadania zdefiniowano następujące zmienne decyzyjne:

- $\mathbf{S}_{mn}$ macierz zawierająca informację o czasie rozpoczęcia każdego z zadań na każdym z procesorów,
- $\mathbf{O}_{jik}$ macierz (zmienna pomocnicza) przechowująca dane o kolejności wykonywania zadań,
- L - suma czasów wykonania wszystkich zadań na każdym z procesorów, górne ograniczenia (zmienna pomocnicza),
- c_{max} - czas zakończenia wszystkich zadań.

Dla dalszego uproszczenia zapisu przyjmijmy, że $M = \{1, \ldots, m\}$ oraz $N = \{1, \ldots n\}$.

Dla problemu została zadana następująca funkcja celu, której zadaniem jest zminimlizowanie czasu zakończenia wszystkich zadań:

$$c_{max} = c_{\pi(m)} \to \min \tag{6}$$

Ponadto zadane zostały następujące ograniczenia:

- i-te zadanie może być przetwarzane na $j+1$ procesorze dopiero, gdy zostanie przetworzone przez procesor j-ty
$$(\forall i \in M, j \in N, j < n)(S_{i,j+1} \geqslant S_{ij} + T_{ij}) \tag{7}$$
- tylko jedno zadanie może byc równocześnie wykonywane na danym procesorze:
$$(\forall (j,i,k) \in O)(S_{ij} - S_{kj} + L \cdot O_{jik} \leqslant T_{kj}) \tag{8}$$
$$(\forall (j,i,k) \in O)(S_{kj} - S_{ij} + L \cdot (1 - O_{jik}) \leqslant T_{ij}) \tag{9}$$
- c_{max} to czas zakończenia wykonywania wszystkich zadań na ostatnim z procesorów:
$$(\forall i \in M)(S_{in} + T_{in}) \leqslant c_{max} \tag{10}$$

4.2 Wyniki

Model przetestowano z solwerem `GLPK` dla następujących danych:

Zadanie / Procesor	**1**	**2**	**3**
1	4	2	2
2	4	3	1
3	2	4	2
4	1	3	2

Czasy wykonania poszczególnych zadań na procesorach

Dla powyższych danych uzyskano poniższy rozkład zadań zaprezentowany na wykresie Gantta wygenerowanym w konsoli tekstowej:

```
4  3  3  1  1  1  1  2  2  2  2  =  =  =  =
=  4  4  4  3  3  3  3  1  1  =  2  2  2  =
=  =  =  =  =  =  =  =  4  4  3  3  1  1  2
```

gdzie cyfra oznacza numer zadania i liczba jej powtórzeń oznacza ile dane zadanie zajęło jednostek czasu na danym procesorze. Symbol = okres bezczynności. Sumarycznie, wykonanie wszystkich zadań zajęło 15 jednostek czasu.