Metody Optymalizacji, Lista 3, rozwiązania

Błażej Wróbel, 250070, W4N, 4. rok

Zadanie 1

Omówienie problemu

W zadaniu 1 należało zaimplementować algorytm aproksymacyjny (oparty na programowaniu liniowym) dla uogólnionego zagadnienia przydziału (ang. Generalized Assignment Problem - GAP) i ocenić eksperymentalnie jego jakość na podstawie znacznej części dostępnych danych testowych.

Uogólnione zagadnienie przydziału - sformułowanie

W omawianym problemie mamy zbiór prac J oraz zbiór maszyn (agentów) M. Dla każdej pracy $j \in J$ jest określony jej czas przetwarzania na danej maszynie $i \in M$, który oznaczamy jako p_{ij}. Ponadto wykonanie danej pracy $j \in J$ na wybranej maszynie $i \in M$ wiążę się z kosztem c_{ij}. Każda maszyna i ma czas dostępności równy T_i. W tym problemie chcemy znaleźć takie przyporządkowanie wszystkich prac do maszyn, że na żadnej maszynie nie zostanie przekroczony jej czas dostępności oraz koszt (wynikający z przydziału zadań) będzie minimalny (minimalizowana funkcja celu).

Formalnie minimalizowaną funkcję celu możemy zapisać jako:

$$F = \sum_{j=1}^{m} \sum_{i=1}^{n} c_{ij} \cdot x_{ij} \rightarrow min \tag{1}$$

Gdzie m to liczba maszyn, n to liczba prac a $x_{ij} \in \{0, 1\}$ jest zmienną binarną oznaczającą fakt, czy praca j została przypisana do maszyny i.

Wyraźmy teraz ograniczenia dla tego problemu. Pierwszym naturalnym (i wynikającym ze sformułowania problemu) ograniczeniem jest fakt, że każda praca musi zostać przypisana do dokładnie jednej maszyny. Ten fakt zapisujemy następująco:

$$\forall j \in J \quad \sum_{i \in M} x_{ij} = 1 \tag{2}$$

Gdzie (identycznie jak poprzednio) zmienna $x_{ij} \in \{0, 1\}$ jest zmienną binarną oznaczającą fakt przypisania zadania j do maszyny i. Następnie zapiszmy ograniczenie wynikające z czasu dostępności każdej maszyny T_i:

$$\forall i \in M \quad \sum_{j \in J} p_{ij} \cdot x_{ij} \leq T_i \tag{3}$$

Jak widać w modelu występują zmienne całkowitoliczbowe, zatem mamy do czynienia z programowaniem całkowitoliczbowym, które (jak wiadomo) jest trudne. W takich sytuacjach mogą nam pomóc algorytmy aproksymacyjne.

Algorytm aproksymacyjny

W podanej na liście książce znajduje się szczegółowy opis algorytmu aproksymacyjnego ze współczynnikiem aproksymacji $\epsilon = 2$. Oznacza to, że zwracane przez algorytm rozwiązanie będzie co najwyżej 2 razy „gorsze" od rozwiązania optymalnego (stosunek kosztu rozwiązania zwracanego przez algorytm do kosztu rozwiązania optymalnego). W tym przypadku algorytm zwraca taki przydział zadań, że czas t_i pracy maszyny nie przekroczy dwukrotności czasu dostępności maszyny T_i, czyli $i \in M \quad t_i \leq 2 \cdot T_i$.

Podstawą analizowanego algorytmu jest fakt, że uogólniony problem przydziału można wyrazić za pomocą pełnego (na początku algorytmu, potem krawędzie są usuwane) grafu dwudzielnego G. Graf G ma dwie „strony" gdzie po jednej stronie są wierzchołki odpowiadające pracom $j \in J$ a po drugiej stronie są wierzchołki odpowiadające maszynom $i \in M$. Krawędź między pracą $j \in J$ a maszyną $i \in M$ ma koszt c_{ij}. Dzięki takiej reprezentacji, GAP można zredukować to problemu znajdowania podgrafu F takiego, że $\forall j \in J \quad d_F(j) = 1$ (gdzie $d_F(j)$ oznacza stopień wierzchołka j w grafie F) i drugi koniec krawędzi to maszyna $i \in M$ do której została przypisana praca j (ponadto to odpowiada wymaganiu, że każda praca ma zostać przydzielona do dokładnie jednej maszyny). W tej reprezentacji ograniczenia czasowe można wyrazić w następujący sposób: $\forall i \in M \quad \sum_{e \in \delta(i) \cap F} p_e \leq T_i$. Dodatkowo tutaj „rodzi" się pierwsza relaksacja zastosowana w algorytmie: jeśli dane $p_{ij} > T_i$ dla wybranych i oraz j, to w żadnym optymalnym przydziale praca j nie będzie przydzielona do maszyny i. Ten fakt powoduje, że będzie można usuwać niektóre zmienne z zadania programowania liniowego, więc będzie można zmniejszać rozmiar tego problemu.

Program liniowy, który jest podstawą omawianego algorytmu ma następującą postać:

$$\sum_{e=(i,j)\in E} c_{ij} \cdot x_{ij} \to min \tag{4}$$

$$\forall j \in J \quad \sum_{e \in \delta(j)} x_e = 1 \tag{5}$$

$$\forall i \in M' \quad \sum_{e \in \delta(i)} x_e \cdot p_e \leq T_i \tag{6}$$

$$\forall e \in E \quad x_e \geq 0 \tag{7}$$

Gdzie $\delta(v)$ oznacza zbiór wszystkich sąsiadów dla danego wierzchołka $v \in V$, a e oznacza parę (i, j) (parę praca-maszyna), czyli krawędź w grafie. Równanie (4) wyraża minimalizowaną funkcję celu, równanie (5) wyraża ograniczenie, że każda praca ma zostać przypisana do dokładnie jednej maszyny oraz równanie (6) wyraża ograniczenia czasowe. Zauważmy, że mamy do czynienia z modelem programowania liniowego (a nie całkowitoliczbowego) oraz, że ograniczenia czasowe mają zachodzić dla pewnego podzbioru $M' \subseteq M$ zbioru maszyn M. Zatem w tym punkcie również mamy do czynienia z pewnymi relaksacjami zastosowanymi w algorytmie.

Pseudokod algorytmu znajduje się na rysunku 1.

Analiza eksperymentalna

Algorytm został przetestowany na wszystkich zestawach danych testowych dostępnych na podanej na liście stronie (gap1.txt, gap2.txt, ..., gapc.txt, gapd.txt). W eksperymentach badałem następujące wielkości:

- Czas rozwiązywania poszczególnych instancji problemu.
- Liczba iteracji algorytmu potrzebna do znalezienia rozwiązania.
- Maksymalne przekroczenie czasu (maksymalny stosunek obciążenia maszyny do dopuszczalnego obciążenia).
- Postęp w jednej iteracji (ilość przydzielonych zmiennych w czasie jednej iteracji).
- Średni czas rozwiązywania problemu w zależności od liczby zadań.
- Średnia liczba iteracji algorytmu potrzebna do znalezienia rozwiązania w zależności od liczby zadań.

Oprócz tego w wynikach zamieszczono również wartości funkcji celu dla przydziału zwróconego przez algorytm.

Na wykresach 2 oraz 3 znajdują się odpowiednio wykresy zależności średniego czasu rozwiązywania instancji problemu od liczby zadań n oraz wykresy średniej liczby iteracji algorytmu w zależności od liczby zadań n. W tabelach 1 (pierwsza część) oraz 2 (druga część) znajdują się dane dotyczące liczby iteracji wykonanych przez algorytm, wartości funkcji celu dla zwróconego przydziału, maksymalnego przekroczenia czasu oraz czasu (w sekundach) potrzebnego na rozwiązanie danej instancji problemu. Natomiast w tabelach 3 (pierwsza część) i 4 (druga część) znajdują się dane dotyczące liczby przydzielonych prac (do maszyn) w każdej iteracji algorytmu.

Wnioski

Analizując dane zebrane w tabelach 1 oraz 2 i zamieszczone wykresy widać, że najmniejsza liczba iteracji wynosi 2 (dla instancji $c10200 - 4$) a największa liczba iteracji wynosi 7 (dla instancji $c1050 - 2$). Najczęściej algorytm potrzebował około $4 - 5$ iteracji do znalezienia rozwiązania (miało to miejsce dla znacznej liczby testowych instancji problemu).

Analizując maksymalne wartości przekroczenia czasu w rozwiązaniach dla poszczególnych instancji można zauważyć, że największe takie przekroczenie ma wartość 1.6471, a najmniejsze 1.0199. W żadnym rozwiązaniu maksymalne przekroczenie czasu nie było nawet bliskie maksymalnej dopuszczalnej wartości 2.0 (omawiany algorytm jest 2-aproksymacyjny).

Analizując tabele 3 oraz 4 widać, że najwięcej prac zostaje przydzielonych w pierwszej iteracji algorytmu. Ma to miejsce dla każdej rozwiązanej instancji testowej problemu. Przykładami są instancje $c525-2$, gdzie w pierwszej iteracji przydzielono 80% prac oraz $c10200-4$, gdzie w pierwszej iteracji przydzielono 99% prac.

Z wykresu 2 widać, że średni czas potrzebny na rozwiązanie instancji problemu rośnie wraz z liczbą zadań n. Natomiast wykres 3 pokazuje, że nie ma zależności między średnią liczbą iteracji a liczbą zadań n w danej instancji problemu (chociaż z omawianego wykresu wynika, że największa średnia liczba iteracji była osiągana dla $n = 50$). Może to być spowodowane tym, że na liczbę iteracji algorytmu mają wpływ inne parametry związane z instancją problemu np. liczba maszyn m lub wartości w macierzy kosztów lub macierzy czasów. Szczególnie to widać na przykładzie instancji $c10200-4$ (2 iteracje) oraz $c10200-4$ (5 iteracji) lub $c20200-6$ (6 iteracji) - wyniki dla tych instancji problemu znajdują się w drugiej części tabeli z wynikami 2.

Podsumowując, algorytm działa szybko nawet dla dużych instancji problemu ($n = 200$ oraz $m = 20$), wykonuje niewiele iteracji i zwraca dobre „jakościowo" wyniki (dla danych, na których był testowany). Teoretycznie omawiany algorytm jest 2-aproksymacyjny, jednakże w zwracanych rozwiązaniach maksymalne przekroczenie obciążenia nie było nawet bliskie wartości 2.0. To „zachowanie" algorytmu jest czymś normalnym dla algorytmów aproksymacyjnych, które w teorii mają współczynnik aproksymacji równy ϵ (w tym przypadku $\epsilon = 2$), ale w praktyce zwracają lepsze rozwiązania (mniej się „mylą").

Iterative Generalized Assignment Algorithm

(i) Initialization $E(F) \leftarrow \emptyset$, $M' \leftarrow M$.

(ii) While $J \neq \emptyset$ do

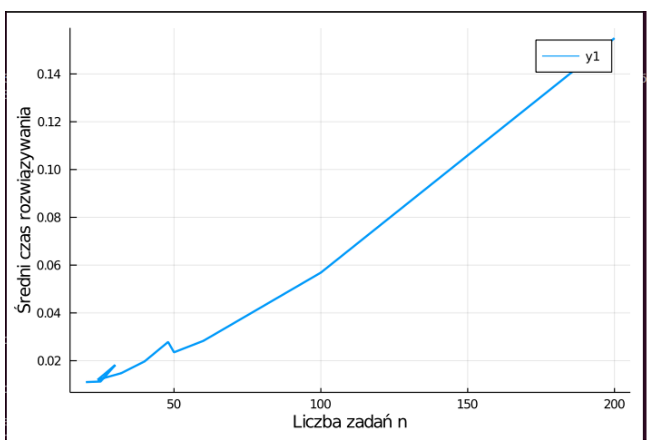
  (a) Find an optimal extreme point solution x to LP_{ga} and remove every variable with $x_{ij} = 0$.

  (b) If there is a variable with $x_{ij} = 1$, then update $F \leftarrow F \cup \{ij\}$, $J \leftarrow J \setminus \{j\}$, $T_i \leftarrow T_i - p_{ij}$.

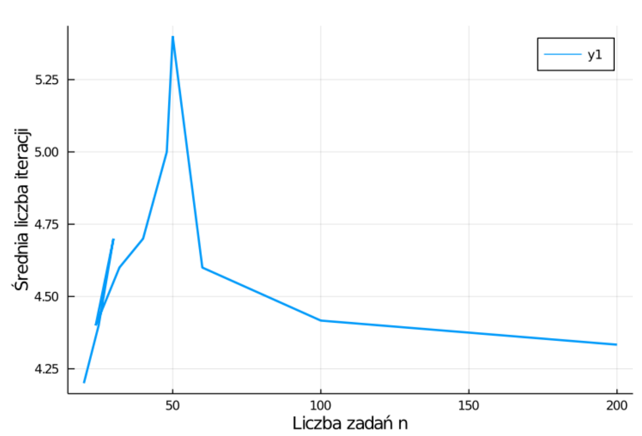
  (c) **(Relaxation)** If there is a machine i with $d(i) = 1$, or a machine i with $d(i) = 2$ and $\sum_{j \in J} x_{ij} \geq 1$, then update $M' \leftarrow M' \setminus \{i\}$.

(iii) Return F.

Rysunek 1: Pseudokod analizowanego algorytmu aproksymacyjnego.



Rysunek 2: Wykres zależności średniego czasu rozwiązywania od liczby zadań n.



Rysunek 3: Wykres zależności średniej liczby iteracji od liczby zadań n.

Instancja	Czas (s)	Iteracje	Funkcja celu	Maksymalne przekroczenie
c515-1	0.009	5	249	1.3684
c515-2	0.0067	4	249	1.2708
c515-3	0.0095	5	245	1.4595
c515-4	0.0078	4	260	1.2778
c515-5	0.0069	4	243	1.6471
c520-1	0.0079	4	259	1.1667
c520-2	0.0089	4	255	1.2391
c520-3	0.0102	5	249	1.3125
c520-4	0.009	4	250	1.2692
c520-5	0.009	4	255	1.3111
c525-1	0.0139	5	428	1.1562
c525-2	0.0113	4	408	1.2083
c525-3	0.0115	4	436	1.1964
c525-4	0.0119	4	421	1.1719
c525-5	0.0169	5	406	1.375
c530-1	0.0148	5	411	1.2222
c530-2	0.0402	4	407	1.2097
c530-3	0.011	4	407	1.2179
c530-4	0.0156	5	380	1.1282
c530-5	0.0156	5	382	1.2
c824-1	0.0117	4	390	1.4857
c824-2	0.0125	4	381	1.325
c824-3	0.0177	6	378	1.2105
c824-4	0.0115	4	375	1.4688
c824-5	0.0123	4	386	1.3939
c832-1	0.0125	4	516	1.3556
c832-2	0.0136	4	513	1.2449
c832-3	0.0137	4	510	1.4444
c832-4	0.0231	6	510	1.5122
c832-5	0.0181	5	513	1.46
c840-1	0.0421	4	633	1.1905
c840-2	0.0167	4	649	1.2143
c840-3	0.019	5	650	1.2982
c840-4	0.0213	5	638	1.2931
c840-5	0.0191	5	639	1.2759
c848-1	0.017	4	786	1.1698
c848-2	0.0203	5	770	1.1458
c848-3	0.0201	5	786	1.2041
c848-4	0.0451	6	782	1.1702
c848-5	0.019	5	778	1.2105
c1030-1	0.0176	5	474	1.2632
c1030-2	0.0192	6	468	1.2368
c1030-3	0.0108	3	481	1.6129

Tablica 1: Pierwsza część tabeli z wynikami.

Instancja	Czas (s)	Iteracje	Funkcja celu	Maksymalne przekroczenie
c1030-4	0.0174	5	482	1.4615
c1030-5	0.0224	5	473	1.4706
c1040-1	0.0145	4	626	1.3061
c1040-2	0.0151	4	628	1.4167
c1040-3	0.034	5	640	1.3636
c1040-4	0.0221	5	627	1.4082
c1040-5	0.0266	6	631	1.2653
c1050-1	0.0231	5	563	1.2787
c1050-2	0.0327	7	574	1.3443
c1050-3	0.0206	5	579	1.1268
c1050-4	0.023	5	561	1.2833
c1050-5	0.0392	5	576	1.2131
c1060-1	0.0298	5	966	1.1795
c1060-2	0.0267	4	944	1.1806
c1060-3	0.0224	4	932	1.2424
c1060-4	0.0296	5	943	1.2462
c1060-5	0.0388	5	934	1.2778
c5100-1	0.0197	3	1696	1.0556
c5200-2	0.0306	3	3234	1.0253
c10100-3	0.0345	4	1358	1.0521
c10200-4	0.0621	2	2623	1.0
c20100-5	0.062	3	1157	1.02
c20200-6	0.1939	3	2337	1.0352
c5100-1	0.018	4	1819	1.0287
c5200-2	0.0444	5	3537	1.0325
c10100-3	0.0635	6	1396	1.1603
c10200-4	0.0984	5	2801	1.0453
c20100-5	0.0897	4	1148	1.2353
c20200-6	0.3126	5	2321	1.1527
c5100-1	0.0211	5	1904	1.0603
c5200-2	0.0365	4	3441	1.0199
c10100-3	0.0571	5	1375	1.1579
c10200-4	0.1074	5	2772	1.075
c20100-5	0.11	5	1200	1.322
c20200-6	0.3006	5	2365	1.1345
c5100-1	0.0198	4	6230	1.0777
c5200-2	0.0635	4	12663	1.0289
c10100-3	0.0498	5	6170	1.1759
c10200-4	0.1228	5	12258	1.077
c20100-5	0.1147	5	5890	1.4444
c20200-6	0.4011	6	11918	1.2038

Tablica 2: Druga część tabeli z wynikami.

Instancja	Liczba przydzielonych prac w iteracji
c515-1	1 => 11 2 => 0 3 => 2 4 => 1 5 => 1
c515-2	1 => 10 2 => 0 3 => 3 4 => 2
c515-3	1 => 10 2 => 0 3 => 3 4 => 1 5 => 1
c515-4	1 => 10 2 => 0 3 => 4 4 => 1
c515-5	1 => 12 2 => 0 3 => 2 4 => 1
c520-1	1 => 15 2 => 0 3 => 4 4 => 1
c520-2	1 => 16 2 => 0 3 => 2 4 => 2
c520-3	1 => 15 2 => 0 3 => 2 4 => 2 5 => 1
c520-4	1 => 15 2 => 0 3 => 3 4 => 2
c520-5	1 => 15 2 => 0 3 => 4 4 => 1
c525-1	1 => 20 2 => 0 3 => 3 4 => 1 5 => 1
c525-2	1 => 20 2 => 0 3 => 4 4 => 1
c525-3	1 => 20 2 => 0 3 => 3 4 => 2
c525-4	1 => 21 2 => 0 3 => 2 4 => 2
c525-5	1 => 20 2 => 0 3 => 3 4 => 1 5 => 1
c530-1	1 => 25 2 => 0 3 => 3 4 => 1 5 => 1
c530-2	1 => 25 2 => 0 3 => 3 4 => 2
c530-3	1 => 26 2 => 0 3 => 3 4 => 1
c530-4	1 => 25 2 => 0 3 => 3 4 => 1 5 => 1
c530-5	1 => 25 2 => 0 3 => 2 4 => 2 5 => 1
c824-1	1 => 18 2 => 0 3 => 5 4 => 1
c824-2	1 => 18 2 => 0 3 => 4 4 => 2
c824-3	1 => 18 2 => 0 3 => 4 4 => 0 5 => 1 6 => 1
c824-4	1 => 16 2 => 0 3 => 5 4 => 3
c824-5	1 => 17 2 => 0 3 => 5 4 => 2
c832-1	1 => 25 2 => 0 3 => 6 4 => 1
c832-2	1 => 26 2 => 0 3 => 4 4 => 2
c832-3	1 => 24 2 => 0 3 => 6 4 => 2
c832-4	1 => 24 2 => 0 3 => 3 4 => 4 5 => 0 6 => 1
c832-5	1 => 25 2 => 0 3 => 5 4 => 1 5 => 1
c840-1	1 => 32 2 => 0 3 => 6 4 => 2
c840-2	1 => 33 2 => 0 3 => 4 4 => 3
c840-3	1 => 32 2 => 0 3 => 3 4 => 4 5 => 1
c840-4	1 => 32 2 => 0 3 => 6 4 => 1 5 => 1
c840-5	1 => 32 2 => 0 3 => 5 4 => 2 5 => 1
c848-1	1 => 41 2 => 0 3 => 6 4 => 1
c848-2	1 => 40 2 => 0 3 => 6 4 => 0 5 => 2
c848-3	1 => 40 2 => 0 3 => 4 4 => 3 5 => 1
c848-4	1 => 41 2 => 0 3 => 3 4 => 3 5 => 0 6 => 1
c848-5	1 => 41 2 => 0 3 => 5 4 => 1 5 => 1
c1030-1	1 => 21 2 => 0 3 => 6 4 => 1 5 => 2
c1030-2	1 => 21 2 => 0 3 => 4 4 => 2 5 => 2 6 => 1
c1030-3	1 => 21 2 => 2 3 => 7

Tablica 3: Tabela z liczbą przydzielonych zmiennych w poszczególnych iteracjach.

Instancja	Liczba przydzielonych prac w iteracji
c1030-4	1 => 22 2 => 0 3 => 7 4 => 0 5 => 1
c1030-5	1 => 20 2 => 0 3 => 7 4 => 2 5 => 1
c1040-1	1 => 31 2 => 0 3 => 7 4 => 2
c1040-2	1 => 32 2 => 0 3 => 5 4 => 3
c1040-3	1 => 32 2 => 0 3 => 5 4 => 2 5 => 1
c1040-4	1 => 31 2 => 0 3 => 7 4 => 1 5 => 1
c1040-5	1 => 30 2 => 0 3 => 6 4 => 2 5 => 1 6 => 1
c1050-1	1 => 43 2 => 0 3 => 4 4 => 1 5 => 2
c1050-2	1 => 40 2 => 0 3 => 6 4 => 1 5 => 2 6 => 0 7 => 1
c1050-3	1 => 41 2 => 0 3 => 7 4 => 1 5 => 1
c1050-4	1 => 40 2 => 0 3 => 6 4 => 2 5 => 2
c1050-5	1 => 42 2 => 0 3 => 6 4 => 1 5 => 1
c1060-1	1 => 52 2 => 0 3 => 6 4 => 1 5 => 1
c1060-2	1 => 51 2 => 0 3 => 6 4 => 3
c1060-3	1 => 51 2 => 0 3 => 8 4 => 1
c1060-4	1 => 50 2 => 0 3 => 7 4 => 2 5 => 1
c1060-5	1 => 50 2 => 0 3 => 5 4 => 2 5 => 3
c5100-1	1 => 99 2 => 0 3 => 1
c5200-2	1 => 199 2 => 0 3 => 1
c10100-3	1 => 98 2 => 0 3 => 1 4 => 1
c10200-4	1 => 198 2 => 2
c20100-5	1 => 97 2 => 2 3 => 1
c20200-6	1 => 195 2 => 1 3 => 4
c5100-1	1 => 95 2 => 0 3 => 3 4 => 2
c5200-2	1 => 195 2 => 0 3 => 3 4 => 1 5 => 1
c10100-3	1 => 91 2 => 0 3 => 5 4 => 2 5 => 1 6 => 1
c10200-4	1 => 190 2 => 0 3 => 5 4 => 4 5 => 1
c20100-5	1 => 85 2 => 0 3 => 13 4 => 2
c20200-6	1 => 184 2 => 0 3 => 12 4 => 2 5 => 2
c5100-1	1 => 95 2 => 0 3 => 3 4 => 1 5 => 1
c5200-2	1 => 195 2 => 0 3 => 3 4 => 2
c10100-3	1 => 91 2 => 0 3 => 6 4 => 2 5 => 1
c10200-4	1 => 190 2 => 0 3 => 6 4 => 3 5 => 1
c20100-5	1 => 81 2 => 0 3 => 14 4 => 4 5 => 1
c20200-6	1 => 184 2 => 0 3 => 13 4 => 2 5 => 1
c5100-1	1 => 95 2 => 0 3 => 3 4 => 2
c5200-2	1 => 195 2 => 0 3 => 4 4 => 1
c10100-3	1 => 90 2 => 0 3 => 7 4 => 2 5 => 1
c10200-4	1 => 190 2 => 0 3 => 6 4 => 2 5 => 2
c20100-5	1 => 81 2 => 0 3 => 15 4 => 3 5 => 1
c20200-6	1 => 180 2 => 0 3 => 11 4 => 5 5 => 2 6 => 2

Tablica 4: Druga część tabeli z liczbą przydzielonych zmiennych w poszczególnych iteracjach.