Metody Optymalizacji
Lista 3

Jakub Gogola
236412

30 maja 2020r.

1 Wprowadzenie

Celem niniejszej listy było zaimplementowanie algorytmu *Generalized Assignment Problem* opisanego w książce *Iterative Methods of Combinatorial Optimization* autorstwa L. Ch. Lau, R. Ravi oraz M. Singh (rozdział 3.2).

2 Opis problemu

Generalized Assignment Problem to problem polegający na przydziela zbioru zadań J zbiorowi maszyn M tak, aby zminimalizowany był całkowity czas wykonania wszystkich zadań. Maszyny potrafią pracować równolegle, jednak na każdą z nich nałożone są pewne ograniczenia. W modelu będziemy posługiwać się następującymi zmiennymi:

- M - zbiór dostępnych maszyn: $M = \{1, \ldots, m\}$,
- J - zbiór zadań do wykonania: $J = \{1, \ldots, n\}$,
- T - zbiór ograniczeń dla maszyn; i-ta maszyna może pracować jedynie T_i jednostek czasu - $T = \{T_i : i \in M\}$,
- c - macierz zawierająca koszty dla wykonania j-tego zadania na i-tej maszynie - $c = \{c_{ij} : i \in M, j \in J\}$,
- p - macierz zawierająca czas przetwarzania j-tego zadania przez i-tą mazynę - $p = \{p_{ij} : i \in M, j \in J\}$

Znalezienie rozwiązania dla problemu sprowadza się do stworzenia grafu dwudzielnego G, gdzie jedną grupę wierzchołków stanowią maszyny ze zbioru M, a drugą - zadania pochodzące ze zbioru J. Na początku zadania mamy do czynienia z grafem pełnym, tj. istnieje połączenie pomiędzy każdym wierzchołkiem ze zbioru M, a pomiędzy każdym z J. Zbiór krawędzi tego grafu oznaczamy przez E.

W kolejnych iteracjach algorytmu (przedstawionego w dalszej części sprawozdania) tworzony jest podgraf grafu G, oznaczany dalej jako F, w którym każdemu zadaniu jest przypisana dokładnie jedna maszyna. Krawędzie bowiem oznaczają przydział zadań do poszczególnych maszyn. Ponadto, dla każdej prawędzi nadana jest waga c_{ij} pochodząca ze zbioru c.

Zmienna decyzyjną dla tego problemu jest macierz $\mathbf{x}$, gdzie x_{ij} oznacza, czy j-te zadanie zostało przypisane do wykonania na i-tej maszynie - $\mathbf{x} = \{x_{ij} : i \in M, j \in J\}$.

Minimalizowana jest funkcja celu:

$$\min \sum_{e=(i,j)\in E} c_{ij} \cdot x_{ij} \tag{1}$$

Ponadto zadańsą dla algrytmu następuujące ograniczenia (przy założeniu, że $M' \subseteq M$):

- $(\forall j \in J) \sum_{e\in\delta(j)} x_e = 1$ - każde zadanie przyporządkowane jest tylko jednej maszynie,
- $(\forall j \in J) \sum_{e\in\delta(i)} p_e \cdot x_e \leqslant T_i$ - czas wykonania przyporządkowanych maszynie zadań nie może przekraczać czasu jej dostępności T_i,
- $(\forall e = (i,j) \in E) x_{ij} \geqslant 0$ - każda zmienna decyzyjna jest nieujemna.

3 Algorytm

Wspomniane wcześniej opracowanie podaje iteracyjny algorytm, który znajduje rozwiązanie dla GAP. Zapewnia on, że każda maszyna $i \in M$ jest używana nie więcej niż dwukrotność dozwolonych dla niej jednostek czasu T_i.

Inicjalizacja: $E(F) \leftarrow \emptyset$, $M' \leftarrow M$
while $J \neq \emptyset$ **do**
    Znajdź punkt ekstremalny (optymalny) x dla zagadnienia LP_{ga} i usuń zmienną x_{ij}.
    if $\exists x_{ij} = 1$ **then**
        $F \leftarrow F \cup \{ij\}$
        $J \leftarrow J \setminus \{j\}$,
        $T_i \leftarrow T_i - p_{ij}$
    end if
    if $(\exists i \in M)(d(i) = 1) \vee (\exists i \in M)(d(i) = 2 \wedge \sum_{j \in J} x_{ij} \leqslant 1)$ **then**
        $M' \leftarrow M' \setminus \{i\}$
    end if
end while

W pseudokodzie algorytmu pojawia się pojęcie **punktu ekstremalnego**, czyli rozwiązania bazowego dopuszczalnego. Innymi słowy - jest to wierzchołek zbioru rozwiązań dopuszczalnych.

4 Wyniki

Zaimplementowany algorytm przetestowano na zbiorach danych dostępnych na `http://people.brunel.ac.uk/~mastjjb/jeb/orlib/gapinfo.html` (dla plików `gap1.txt - gap12.txt`). Zmierzono czas znajdowania rozwiązania dla każdego z problemów ze zbioru danych testowych. Zmierzono również liczbę potrzebnych algorytmowi iteracji oraz postęp w budowaniu rozwiązania (postęp w liczbie dopasowań zadanie - maszyna w każdej iteracji). Sprawdzono również czy dla żadnej z maszyn nie został przekroczony limit $2T_i$ (**nie został** przekroczony dla żadnego z problemów), a także obliczono przeciążenie z każdej z maszyn (wykorzystanie ponad przyporządkowaną liczbę jednostek czasu T_i). Do testów użyto solwera `GLPK`.

Szczegółowe wyniki dla wszystkich zagadnień można uzyskać wywołując funkcję `run_all`. Wówczas wydrukowane zostaną m.in. obciążenia dla każdej z maszyn, a także postęp dla każdego z problemów w każdej z iteracji. Ze względu na mnogość danych, w niniejszym sprawozdaniu postanowiłem przedstawić jedynie te, które zobrazują jakoś zwracanych rozwiązań. W tabeli przedstawiono dla każdego pliku (i każdego problemu w tym pliku się zawierającego) zestawienie - liczbę potrzebnych iteracji dla wyznaczenia podgrafu F, czas działania algorytmu dla tego problemu, maksymalne przeciążenie maszyny w podgrafie F dla danego rozwiązania ($T_{max}(F)$), maksymalne dopuszczalne przeciążenie maszyny (T_{max}) oraz stosunek pomiędzy dwiema ostatnimi wymienionymi wielkościami - $\frac{T_{max}(F)}{T_{max}}$.

4.1 Spostrzeżenia

Na podstawie wyników zwracanych przez algorytm udało się zaobserwować, że dla zdecydowanej większości przypadków już w pierwszych dwóch iteracjach wyznaczone zostaje ok 80% par maszyna-zadanie. Ponadto, dla większości przypadków przydziału zadań do maszyn, algorytm powoduje ich przeciążenie, jednak w żadnym przypadku nie zostaje przekroczone ograniczanie, że przeciążenie danej maszyny powinno być mniejsze niż $w \cdot T_i$.

Plik	Problem	Liczba iteracji	Czas działania [ms]	$T_{max}(F)$	T_{max}	$\frac{T_{max}(F)}{T_{max}}$
gap1	c515-1	5	3.32	52	38	1.368
gap1	c515-2	6	4.99	50	48	1.042
gap1	c515-3	4	2.9	54	44	1.227
gap1	c515-4	5	3.13	46	39	1.179
gap1	c515-5	5	3.17	56	40	1.4
gap2	c520-1	7	6.06	62	55	1.127
gap2	c520-2	5	4.07	57	51	1.118
gap2	c520-3	6	4.32	63	51	1.235
gap2	c520-4	5	3.62	66	52	1.269
gap2	c520-5	5	5.04	59	54	1.093
gap3	c525-1	5	5.13	74	64	1.156
gap3	c525-2	7	5.91	64	61	1.049
gap3	c525-3	5	5.53	75	65	1.154
gap3	c525-4	5	4.49	75	69	1.087
gap3	c525-5	6	4.98	77	61	1.262
gap4	c530-1	6	7.41	88	79	1.114
gap4	c530-2	5	4.9	88	80	1.1
gap4	c530-3	5	4.71	95	78	1.218
gap4	c530-4	6	7.23	88	78	1.128
gap4	c530-5	7	6.7	86	74	1.162
gap5	c824-1	6	8.36	52	38	1.368
gap5	c824-2	5	5.93	53	40	1.325
gap5	c824-3	6	7.64	46	40	1.15
gap5	c824-4	5	6.59	47	40	1.175
gap5	c824-5	5	5.74	47	40	1.175
gap6	c832-1	5	8.65	64	52	1.231
gap6	c832-2	5	7.46	63	54	1.167
gap6	c832-3	5	29.12	65	53	1.226
gap6	c832-4	6	10.97	62	53	1.17
gap6	c832-5	6	9.04	73	54	1.352
gap7	c840-1	5	10.27	75	64	1.172
gap7	c840-2	6	11.88	75	64	1.172
gap7	c840-3	6	10.65	74	68	1.088
gap7	c840-4	6	11.41	78	71	1.099
gap7	c840-5	6	11.71	74	67	1.104
gap8	c848-1	5	11.2	62	53	1.17
gap8	c848-2	6	12.58	57	53	1.075
gap8	c848-3	6	12.47	59	54	1.093
gap8	c848-4	7	13.12	55	51	1.078
gap8	c848-5	6	11.43	69	57	1.211
gap9	c1030-1	6	9.05	50	40	1.25
gap9	c1030-2	7	26.77	47	43	1.093
gap9	c1030-3	5	8.51	55	39	1.41
gap9	c1030-4	7	10.45	52	39	1.333
gap9	c1030-5	6	9.13	56	45	1.244
gap10	c1040-1	6	12.11	64	51	1.255
gap10	c1040-2	5	10.31	68	50	1.36
gap10	c1040-3	6	11.22	68	52	1.308
gap10	c1040-4	6	10.67	69	51	1.353
gap10	c1040-5	7	11.97	59	51	1.157
gap11	c1050-1	6	13.32	84	72	1.167
gap11	c1050-2	7	14.83	82	72	1.139
gap11	c1050-3	6	13.28	92	77	1.195
gap11	c1050-4	7	30.26	92	71	1.296
gap11	c1050-5	6	14.57	74	69	1.072
gap12	c1060-1	6	16.64	92	79	1.165
gap12	c1060-2	4	11.5	85	76	1.118
gap12	c1060-3	5	13.83	89	81	1.099
gap12	c1060-4	5	13.52	92	80	1.15
gap12	c1060-5	5	13.48	92	77	1.195