Wrocław, 17 marca 2020

Sterowanie **P**rocesami **D**yskretnymi - Laboratorium
Problem *RPQ* - *Algorytm Schrage*

prowadzący: *mgr inż. Radosław Idzikowski*

1 Wprowadzenie

Celem laboratorium jest zapoznanie się z algorytmem Schrage dla problemu RPQ oraz dla problemu RPQ z przerwaniami.

2 Problem RPQ

Problem $1|r_j, q_j|C_{\max}$ możemy rozwiązać algorytmem *schrage*. W tym celu potrzebujemy wprowadzić dodatkowe elementy:

- $\mathcal{N}$ – zbiór zadań nieuszeregowanych,
- $\mathcal{G}$ – zbiór zadań gotowych do realizacji,
- t – zmienna pomocnicza symbolizującą chwilę czasową.

Sposób działa algorytmu mamy przedstawiony za pomocą pseudokodu 1. W liniach $2-5$ mamy inicjalizacje zmiennych początkowych, gdzie k – numer zadania w permutacji π, które aktualnie chcemy rozpatrzyć, zbiór zadań $\mathcal{G}$ początkowo zostawiamy pusty, ponieważ nie mamy żadnych zadań gotowych do realizacji na maszynie, do zbioru $\mathcal{N}$ przypisujemy wszystkie zdania, początkowa wartość zmiennej czasu t jest równa najmniejszemu terminowi dostępności zadania r_j ze zbioru zadań nieuszeregowanych $\mathcal{N}$.

Główny schemat algorytmu jest zawarty w pętli `while` w liniach $6-21$, która wykonuję się dopóki mamy jeszcze zadania nieuszeregowane w zbiorze $\mathcal{G}$ lub $\mathcal{N}$. Pętla `while` w liniach $7-11$ odpowiada, że przesunięcie przygotowanych zadań w chwili t ze zbioru $\mathcal{N}$ do zbioru $\mathcal{G}$. W instrukcji warunkowej `if` jeśli zbiór zadań gotowych $\mathcal{G}$ nie jest pusty, to należy przyporządkować zadanie o najdłuższym czasie czasie dostarczenia q_j do wykonania na maszynie, pamiętając o tym, że zadanie nie może się zacząć wykonywać wcześniej niż jego termin dostępności i czas zakończenia ostatniego zadania:

$$S_{\pi(j)} = max\{r_{\pi(j)}, C_{\pi(j-1)}\}, \tag{1}$$

jeśli nie mam zadań gotowych do realizacji ($\mathcal{G} = \emptyset$), to należy zmienić chwilę czasową t na termin dostępności najbliższego zadania ze zbioru zadań nieuszeregowanych (o najmniejszym r_j).

Analizując opis oraz pseudokod algorytmu można łatwo zauważyć, że jesteśmy często zmuszeni do przeszukiwania zbioru $\mathcal{N}$ i $\mathcal{G}$, dlatego jednym z celów laboratorium jest napisanie wersji programu z wyszukiwaniem minimum/maximum oraz drugiej z wykorzystaniem struktur samoorganizujących się (np.: kolejka priorytetowa).

Algorithm 1 Pseudokod algorytmu Schrage

procedure SCHRAGE
    $k \leftarrow 1$
    $\mathcal{G} \leftarrow \emptyset$
    $\mathcal{N} \leftarrow \mathcal{J}$
    $t \leftarrow \min_{j \in \mathcal{N}} r_j$
    while $\mathcal{G} \neq \emptyset \vee \mathcal{N} \neq \emptyset$ **do**
        while $\mathcal{N} \neq \emptyset \wedge \min_{j \in \mathcal{N}} r_j \leqslant t$ **do**
            $j^* \leftarrow \arg\min_{j \in \mathcal{N}} r_j$
            $\mathcal{G} \leftarrow \mathcal{G} \cup \{j^*\}$
            $\mathcal{N} \leftarrow \mathcal{N} \setminus \{j^*\}$
        end while
        if $\mathcal{G} \neq \emptyset$ **then**
            $j^* \leftarrow \arg\max_{j \in \mathcal{G}} q_j$
            $\mathcal{G} \leftarrow \mathcal{G} \setminus \{j^*\}$
            $\pi(k) \leftarrow j^*$
            $t \leftarrow t + p_{j^*}$
            $k \leftarrow k + 1$
        else
            $t \leftarrow \min_{j \in \mathcal{N}} r_j$
        end if
    end while
    return π
end procedure

3 Problem *RPQ* z przerwaniami

Problem $1|r_j, q_j, \mathrm{pmtn}|C_{\max}$ jest zbliżony do omówionego problemu $1|r_j, q_j|C_{\max}$, ale proszę pamiętać, że jest to całkiem inny problem, ponieważ dopuszczamy możliwość przerwania zadania. Dopuszcza się wielokrotne przerywanie jednego zadania. Zadanie na maszynie możemy przerwać w momencie $t = r_j$, kiedy pojawi się nam dostępne zadanie o większym czasie q_j niż aktualne wykonywane. Zadanie przerwane uważa się za częściowo wykonane należy odłożyć do zbioru $\mathcal{G}$ z pomniejszym czasem wykonywania. Zmodyfikowany algorytm Schrage rozwiązuje problem $1|r_j, q_j, \mathrm{pmtn}|C_{\max}$ optymalnie.

4 Zadanie

1. algorytm Schrage dla problemu $1|r_j, q_j|C_{\max}$ bez wykorzystania kolejki,
2. algorytm Schrage dla problemu $1|r_j, q_j|C_{\max}$ z wykorzystaniem kolejki,
3. algorytm Schrage dla problemu $1|r_j, q_j, \mathrm{pmtn}|C_{\max}$ bez wykorzystania kolejki,
4. algorytm Schrage dla problemu $1|r_j, q_j, \mathrm{pmtn}|C_{\max}$ z wykorzystaniem kolejki,

5 UWAGA!

Aby uzyskać ocenę *3.0* za laboratorium $2-4$ – problem *RPQ* należy napisać wszystkie programy z laboratorium nr 1 oraz algorytm Schrage dla problemu $1|r_j, q_j|C_{\max}$ bez wykorzystania kolejki lub z wykorzystaniem kolejki. Kolejne wersje algorytmów będą zwiększać ocenę o około 0.25. Więcej szczegółów w instrukcji do laboratorium nr 3.

6 Wyniki

Wersje algorytmów z lub bez wykorzystania kolejki zwracają ten sam wynik, różnią się za to złożonością obliczeniową.

Tabela 1: Wyniki dla algorytmu Schrage dla problemu *RPQ*

name	data10	data20	data50	data100	data200	data 500
Schrage	687	1309	1514	3076	6416	14786

opracował: *Radosław Idzikowski*