Wrocław, 2 marca 2021

Sterowanie Procesami Dyskretnymi

Laboratorium 0

Wprowadzenie do teorii szeregowania zadań

prowadzący: *mgr inż. Radosław Idzikowski*

1 Cel laboratorium

Celem laboratorium jest zapoznanie się z podstawami teorii szeregowania zadań oraz sposobami modelowania oraz rozwiązywania podestowych problemów. Obejmuje to odpowiednie zdefiniowane problemu (ograniczeń i funkcji celu), danych wejściowych oraz implementacje prostego algorytmu opartego na sortowaniu, a także interpretację wyników.

2 Przebieg zajęć

Laboratorium obejmuje połowę zajęć nr 1 (około 1 godziny). Praca odbywa się w ramach zespołów dwuosobowych. Każdy zespół otrzymuje do zrealizowania jeden problem w tym wypadku $1|r_j|C_{\max}$ oraz jeden algorytm. Wszystkie zespoły realizują to samo zdanie.

W trakcie zajęć w wybranym języku `python`, `C/C++`, `Java` lun `C#` należy zaimplementować metodę generowania instancji przy użyciu załączonych generatorów dla zadanych parametrów (źródło, rozmiar, zakres). Kolejnym krokiem jest implementacja metody oceniania rozwiązania (liczenia funkcji celu). Ostatnim etapem jest implementacja algorytmu bazującego na sortowaniu (można używać wbudowanych metod).

Zadanie nie jest na ocenę, ale należy je przesłać na koniec zajęć do prowadzącego. Warto przetestować prace z githubem.

3 Problem

W problemie $1|r_j|C_{\max}$ mamy zbiór $\mathcal{J} = \{1, 2, \ldots, n\}$ n zadań. Wszystkie zadania muszą zostać wykonanie/przetworzone na maszynie. Każde j-te zadanie składa się dwóch parametrów:

- r_j - czas przygotowania/termin dostępności,
- p_j - czas wykonania.

Każde zadanie, wykonywane nieprzerwanie przez p_j czasu. Jednak wcześniej musi zostać przygotowane przygotowane przez r_j czasu..Na maszynie na raz może być wykonywane dokładnie jedno zadanie. Przez π będziemy oznaczać kolejność wykonywania zadań na maszynie. Momenty rozpoczęcia wykonywania zadań będziemy opisywać wektorem S, gdzie:

$$r_{\pi(j)} \leqslant S_{\pi(j)}. \tag{1}$$

Momenty zakończenia zadań opiszemy wektorem C, gdzie:

$$C_{\pi(j)} = S_{\pi(j)} + p_{\pi(j)}. \tag{2}$$

Musimy pamiętać, że kolejne zadanie nie może zacząć wcześniej niż czas jego przygotowania i czas zakończenia zadania poprzedniego:

$$S_{\pi(j)} = max\{r_{\pi(j)}, C_{\pi(j-1)}\} \tag{3}$$

Badanym kryterium optymalizacyjnym jest czas zakończenia wszystkich zadań:

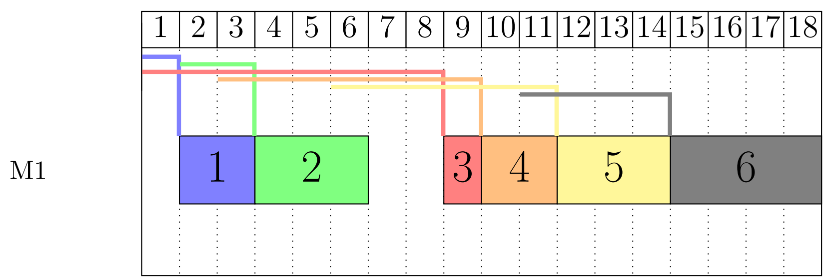
$$C_{\max}(\pi) = \arg\max_{j \in \mathcal{J}}\{C_{\pi(j)}\} \tag{4}$$

Dla badanego problemu czas zakończenia wszystkich zadań jest równoważny z czasem zakończenia ostatniego zadania:

$$C_{\max}(\pi) = C_n \tag{5}$$

4 Przykład

Na Rysunku 1 mamy graficzne rozwiązanie problemu dla danych z Tabeli 1. Każde zadanie dla ułatwienia oznaczono innym kolorem. Czas wykonywania zadań na maszynie jest zaprezentowany w formie bloczku. Czas przygotowania jest oznaczony "*wąsem*". Na przedstawionym schemacie gantt'a dobrze są widoczne przestoje (*przerwy*) na maszynie.



Rysunek 1: Problem RPQ: $n = 3$ dla $\pi = (1, 2, 3)$

Tabela 1: Instancja $n = 6$

j	1	2	3	4	5	6
r_j	1	2	8	7	6	4
p_j	2	3	1	2	3	4

Rozwiązanie możemy również zaprezentować w formie harmonogramu (czasy S_j oraz C_j), co przedstawiono w Tabeli 2. Czas zakończenia oraz dostarczenia wszystkich zdań pogrubiono.

Tabela 2: Tworzenie harmonogramu dla $\pi = (1, 2, 3)$

j	1	2	3	4	5	6
S_j	1	3	8	9	11	14
C_j	3	6	9	11	14	18

5 Sposób generowania instancji

Dla parametru n oraz ziarna Z:

1. init(Z).
2. Dla każdego $j \in \mathcal{J} : p_j \leftarrow$ nextInt$(1, 29)$.
3. $A \leftarrow \sum_{j=1}^{n} p_i$.
4. Dla każdego $j \in \mathcal{J}$:$r_j \leftarrow$ nextInt$(1, A)$.

Tabela 3: Instancja $n = 10$ $Z = 1$

j	1	2	3	4	5	6	7	8	9	10
r_j	52	70	112	5	8	71	90	2	52	9
p_j	1	4	22	14	16	7	2	20	20	28

Wartość $C_{\max}(\pi_{1234}) = 241$ oraz $C_{\max}(\pi_{sort}) = 136$, gdzie przez π_{1234} oznaczono permutacje naturalną, a π_{sort} permutacją otrzymaną po sortowaniu po parametrze r_j.

opracował: *Radosław Idzikowski*