Wrocław, 4 października 2018

Projektowanie **E**fektywnych **A**lgorytmów - Projekt
prowadzący: *mgr inż. Radosław Idzikowski*

1 Założenia projektu

W ramach zajęć należy zaimplementować oraz dokonać analizy efektywności algorytmów dla wybranego problemu optymalizacyjnego. W celu zaliczenia trzeba zrealizować projekty nr 1-3, aby móc ubiegać się o ocenę 5.5 z poprzednich projektów trzeba uzyskać średnią conajmniej $4,66$.

Tematy:

1. *Algorytm dokładny,* **16.11.2018**
2. *Algorytm poszukiwania lokalnego,* **14.12.2018**
3. *Algorytm populacyjny,* **18.01.2019**
4. Ustalany indywidualnie **18.01.2019**

Problemy:

- Problem Komiwojażera *(**T**ravelling **S**alesman **P**roblem)*,
- Problem Przepływowy *(**F**low **S**hop **P**roblem)*,
- Ważona Suma Opóźnień dla jednomaszynowego problemu szeregowania zadań *(**W**eighted **S**um **T**ardiness for task scheduling)*.

Podczas realizacji zadania należy przyjąć następujące założenia:

1. programy oraz sprawozdanie trzeba oddać w terminie, za każdy tydzień spóźnienia obniżana jest ocena o 0.5,
2. implementacji algorytmu należy dokonać zgodnie z obiektowym paradygmatem programowania,
3. program ma umożliwić weryfikację poprawności działania algorytmu. W tym celu trzeba zapewnić możliwość wczytania danych wejściowych z pliku tekstowego,
4. po zaimplementowaniu i sprawdzeniu poprawności działania algorytmu należy dokonać pomiaru czasu jego działania w zależności od rozmiaru problemu dla załączonych danych wejściowych,
5. używanie „okienek" nie jest konieczne i nie wpływa na ocenę,

Sprawozdanie powinno zawierać:

- opis problemu,
- opis algorytmu,
- szczegóły odnośnie zaimplementowanego algorytmu (np. użyty operator krzyżowania, lista tabu, parametry algorytmu),
- wyniki,
- wnioski.

oraz jeśli dotyczy:

- jakość dostarczanych rozwiązań,
- wpływ najważniejszych parametrów (jak liczba iteracji, liczba osobników, długość listy tabu) na jakość wyników.

UWAGA! Aby uzyskać ocenę bardzo dobrą (5.0) sprawozdanie musi być wykonane w technologii LaTeX

2 Opisy problemów

2.1 TSP

Problem Komiwojażera *(**T**ravelling **S**alesman **P**roblem)*, W problemie mamy n miast oraz macierz nxn przejść pomiędzy miastami. Za zadanie mamy znaleźć jak najkrótszą ścieżkę przechodzącą przez wszystkie miasta wliczając w to powrót do miasta początkowego.

2.2 FSP

Problem Przepływowy *(**F**low **S**hop **P**roblem)*, W problemie mamy zbiór zadań o wielkości n:

$$J = \{J_1, J_2, ...J_j, ..., J_n\}, \tag{1}$$

tym razem mamy zbiór maszyn o wielkości m:

$$M = \{M_1, M_2, ...M_i, ..., M_m\}, \tag{2}$$

każde zadanie jest podzielone na m operacji, które wykonują się na maszynach:

$$J_j = \{O_{1,j}, O_{2,j}, ...O_{i,j}, ..., O_{1,j}\}, \tag{3}$$

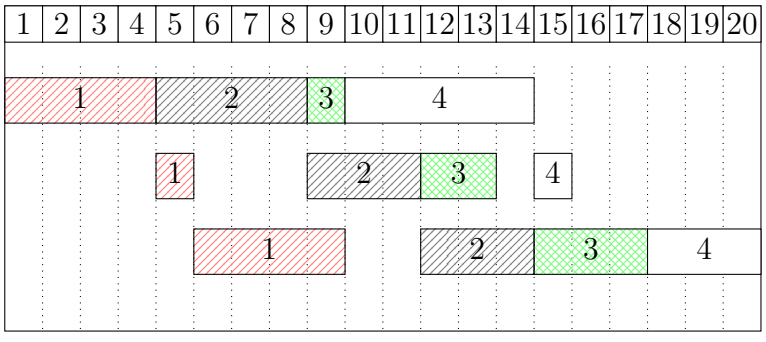
Warunki:

- zadania wykonują się nieprzerwanie,
- na wszystkich maszynach jest ta sama kolejność wykonywania zadań,
- w ramach każdego zadania musi być zachowany porządek technologiczny:

$$S_{i+1,\pi(j)} \geqslant C_{i,\pi(j)} \tag{4}$$

Kryterium optymalizacyjnym jest czas zakończenia wszystkich zadań, co jest równe czasowi zakończenia ostatniego zadania:

$$C_{max} = C_{m,\pi(n)} \tag{5}$$



zadanie	1	2	3	4
maszyna 1	4	4	1	5
maszyna 2	1	3	2	1
maszyna 3	4	3	3	3

2.3 WST

Kryterium Ważonej Sumy Opóźnień dla jednomaszynowego problemu szeregowania zadań*(**W**eighted **S**um **T**ardiness for task scheduling)*. W problemie mamy zbiór n zadań wykonywanych na jednej maszynie.

$$J = \{J_1, J_2, ...J_i, ..., J_n\}, \tag{6}$$

każde i-te zadanie składa się z trzech parametrów:

- p_i - czas wykonania,
- w_i - współczynnik kary,
- d_i - żądany termin zakończenia.

Każde zadanie musi być wykonywane nieprzerwanie przez p_i oraz powinno być ukończone przed terminem d_i, w przeciwnym wypadku zostanie naliczona kara. Naraz może być wykonywane tylko jedynie jedno zadanie. Przez π będziemy oznaczać kolejność wykonywania zadań. Dla każdego zadania należy obliczyć jego spóźnienie T_i:

$$T_i = max(C_i - d_i, 0) \tag{7}$$

gdzie C_i jest to moment zakończenia i-tego zadania, a wcześniejsze zakończenie nie jest dodatkowo premiowane. Kryterium optymalizacyjnym jest suma wiTi:

$$\sum_{i=1}^{n} w_i T_i \tag{8}$$

3 Zadania możliwe do zrealizowania

3.1 Algorytm dokładny

- przegląd zupełny (***B**rute **F**orce*),
- algorytm podziału i ograniczeń (***B**ranch and **B**ound*),
- programowanie dynamiczne (***D**ynamic **P**rogramming*).

3.2 Algorytm poszukiwania lokalnego

- poszukiwanie z zakazami (***T**abu **S**earch*),
- symulowane wyżarzanie (***S**imulated **A**nnealing*),
- inny po wcześniejszej konsultacji.

3.3 Algorytm populacyjny

- algorytm genetyczny (***G**enetic **A**lgorithm*),
- algorytm mrówkowy (***A**nt **C**olony **O**ptimization*),
- inny po wcześniejszej konsultacji.