

Implementacja i analiza efektywności algorytmu podziału i ograniczeń lub programowania dynamicznego dla wybranego problemu optymalizacji

Projektowanie efektywnych algorytmów



Autor: Maciej Bronikowski 248838

Grupa projektowa: środa 13:15  
Prowadzący: Antoni Sterna

# Opis problemu

Przedmiotem projektu było zaprojektowanie algorytmów dla jednoprocesorowego problemu szeregowania zadań przy kryterium minimalizacji ważonej sumy opóźnień zadań. Problem przedstawia zadania opisywane przez trzy parametry:

* pi – czas wykonywania zadania
* di – termin, w którym zadanie powinno być wykonane
* wi – dodatnia waga zadania

Zadania należy ułożyć kolejno w taki sposób, aby ich suma kar zapisanych wzorem  
 , gdzie jest jak najmniejsza. Zadania są przetwarzane na jednej maszynie, nie mogą być wykonywane jednocześnie oraz muszą być wykonywane bez przerwy od czasu rozpoczęcia przez czasy wykonywania. Problem ten jest NP – trudny, czyli niedający się rozwiązać w złożoności czasowej wielomianowej. W projekcie zostały zastosowane trzy algorytmy: przeglądu zupełnego, podziału i ograniczeń oraz programowania dynamicznego.

## Przykład obliczania całkowitej kary za opóźnienia

|  |  |  |  |
| --- | --- | --- | --- |
| Nr zadania | 1 | 2 | 3 |
| pi | 4 | 6 | 2 |
| di | 10 | 8 | 10 |
| wi | 3 | 6 | 4 |

Dla podanego przykładu zostanie obliczona sumaryczne ważone opóźnienie dla kolejności zadań {1,2,3}. Najpierw należy wyznaczyć Ti:

|  |  |  |
| --- | --- | --- |
| T1 = max(0, 4 - 10) = 0 | T2 = max(0, 4 + 6 - 8) = 2 | T3 = max(0, 10 + 2 – 10) = 2 |

Całkowity ważony czas opóźnienia dla danej kolejności będzie równy:

# Algorytmy wyznaczania minimalnego rozwiązania

## Brute force

Brute force jest metodą siłową, sprawdzającą wszystkie możliwe permutacje kolejności zadań i oblicza dla nich opóźnienie ważone. Do wygenerowania wszystkich możliwych kolejności indeksów zadań został wykorzystany algorytm Heapa. Algorytm ten używany jest przez klasę WeightedTardiness w metodzie bruteforce.

## Generowanie wszystkich permutacji zbioru zadań

Do algorytmu brute force została wykorzystana wersja rekurencyjna algorytmu Heap’a.



Algorytm dokonuje zamian na odpowiednich pozycjach wektora order, dopóki k != 1, które oznacza koniec generowania danej permutacji. Operacje wykonywane przez algorytm na vector o rozmiarze równym 4 można zobaczyć w tablicy:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |