Politechnika Wrocławska

Wydział Podstawowych Problemów Techniki

Metody optymalizacji

Sprawozdanie – Lista nr 2

Autor: Tomasz Kulik

Wrocław 2017r.

**Zadanie 1.**

1. **Opis problemu**

Dane są zamówienia na deski o podanych szerokościach i zapotrzebowaniu na deski o danej szerokości. Należy znaleźć sposoby podziału standardowych desek na mniejsze, aby zminimalizować straty wynikłe z powstania drewnianych odpadów nie zdatnych do wytworzenia produktu, na który jest zapotrzebowanie. Tartak otrzymał zamówienie na 110 desek o szerokości 7 cali, 120 szerokości 5 cali oraz 80 desek o szerokości 3 cali. Standardowym rozmiarem deski w tartaku to deska 22-calowa.

1. **Rozwiązanie**

Należy znaleźć wszystkie możliwe podziały standardowej deski na niestandardowe szerokości. Służyć temu ma rekurencyjny algorytm wyszukujący podziały desek, w których wytworzony odpad jest mniejszy od najmniejszej zamawianej deski. Dzięki ostatniej własności unika się wprowadzania do modelu wielu zbędnych podziałów, przez co zmniejsza się liczbę zbędnych ograniczeń w modelu. Następnie należy zbudować model dla **Mixed Integer Programming**, który dla każdego podziału standardowej deski określi ile desek należy podzielić danym podziałem.

Niech zmiennymi modelu będą:

Należy zdefiniować również zbiór wektorów:

Ograniczeniami modelu będą:

Funkcja celu:

1. **Wynik**

Optymalny sposób podziału desek przy zerowych odpadach:

* 37 desek należy podzielić sposobem: 2\*7 cali, 1\*5 cali, 1\*3 cale;
* 28 desek należy podzielić sposobem: 1\*7cali, 3\*5cali;
* 9 desek należy podzielić sposobem: 1\*7 cali, 5\*3cale.

**Uruchomienie zadań 2, 3, 4 wymagają zainstalowania pakietu PyCall, środowiska Python oraz bibliotek matplotlib w celu generowania wykresów Gantt’a.**

**Zadanie 2.**

1. **Opis problemu**

Dany jest problem szeregowania **n** zadań na jednej maszynie. Zadania nie mogą wykonywać się równolegle. Każde zadanie ma przypisaną wagę oraz czas wykonywania . Należy znaleźć takie uszeregowanie, które zminimalizuje , gdzie oznacza koniec wykonywania zadania i-tego.

1. **Rozwiązanie**

Należy wprowadzić podstawowe własności potrzebne w dalszych etapach:

W celu wykrycia kolizji można posłużyć się ograniczeniami:

Powyższe stwierdzenie mówi, że jest prawdziwe wtw, gdy początek zadania i-tego mieści się w czasie trwania zadania j-tego. W przeciwnym wypadku jest fałszywe. To w łatwy sposób przy pomocy zmiennych decyzyjnych pozwala kontrolować kolizje w uszeregowaniu zadań. Model całkowitoliczbowy nie pozwala na wprowadzenie logicznego operatora ‘lub’, należy posłużyć się sposobem symulowania takiego zachowania:

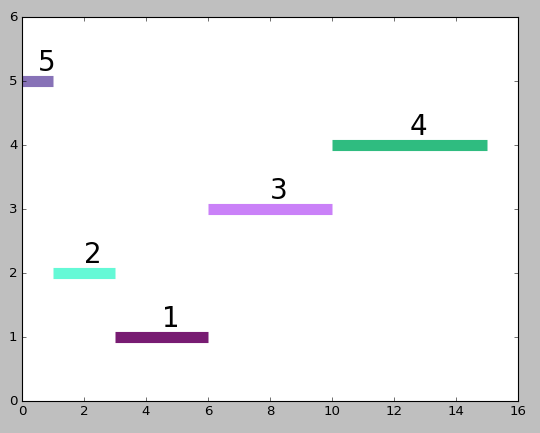
Zmienne (dla każdej pary (i, j) są dobierane osobne zmienne) są zmiennymi pomocniczymi, ich wartości nie mają wpływu na wynik końcowy. Służą do zasymulowania logicznego operatora ‘lub’ w modelu programowania całkowitoliczbowego. Dzięki wprowadzeniu działań ze stałą możliwa jest kontrola nad prawdziwością danych nierówności odpowiednio manipulując zmiennymi decyzyjnymi oraz .

Mając powyższe ograniczenia można wprowadzić zasadę braku kolizji:

Funkcja celu została podana w opisie problemu.

1. **Wynik**

Wynik działania programu dla przykładowych danych:

****

**Zadanie 3.**

1. **Opis problemu**

Dany jest problem szeregowania **n** zadań na **m** maszynach. Zadania nie mogą wykonywać się równolegle na jednej maszynie. Każde zadanie ma przypisany czas wykonywania oraz może mieć wyznaczone zadania, które należy wykonać przed jego rozpoczęciem. Należy znaleźć takie uszeregowanie, które zminimalizuje – czas zakończenia wszystkich zadań, gdzie oznacza koniec wykonywania zadania i-tego.

1. **Rozwiązanie**

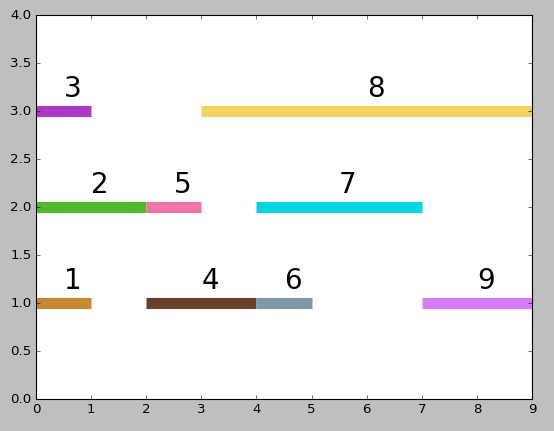
W celu rozwiązania tego problem można posłużyć się modelem wykrywania kolizji z zadania 2. Mając macierz należy zamienić ostatnie ograniczenie na:

Należy wprowadzić również ograniczenia wynikające z relacji poprzedzania między zadaniami:

Funkcją celu będzie zminimalizowanie zmiennej, która jest większa bądź równa zakończeniu każdego z zadań

Przyporządkowanie zadań do poszczególnych maszyn rozwiązano przy pomocy kolejnego modelu **MIP**, który jako dane wejściowe otrzymuje macierz kolizji zadań, która jest efektem ubocznym modelu wykrywania kolizji. Dzięki tej macierzy można w łatwy sposób znaleźć odpowiednie dopasowanie:

1. **Wynik**

****

**Zadanie 4.**

1. **Opis problemu**

Dany jest problem szeregowania **n** zadań. Każde zadanie ma przypisany czas wykonywania oraz może mieć wyznaczone zadania, które należy wykonać przed jego rozpoczęciem. Należy znaleźć takie uszeregowanie, które zminimalizuje – czas zakończenia wszystkich zadań, gdzie oznacza koniec wykonywania zadania i-tego. Każde zadanie pochłania pewną część zasobów odnawialnych.

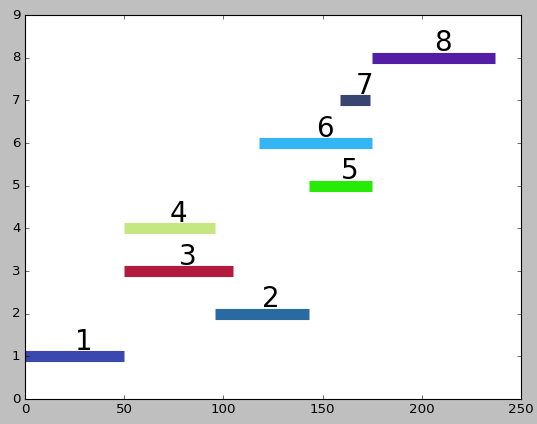
1. **Rozwiązanie**

W celu rozwiązania tego problem można posłużyć się modelem wykrywania kolizji z zadania 2. Mając macierz należy zamienić ostatnie ograniczenie na:

Należy wprowadzić również ograniczenia wynikające z relacji poprzedzania między zadaniami:

Funkcją celu będzie zminimalizowanie zmiennej, która jest większa bądź równa zakończeniu każdego z zadań

1. **Wynik**

****