Uniwersytet Zielonogórski

Instytut Sterowania i Systemów Informatycznych

Laboratorium

Wstęp do programowania całkowitoliczbowego oraz pakietu yalmip

Cele ćwiczenia

Celem ćwiczenia jest zapoznanie się z zagadnieniem programowania całkowitoliczbowego oraz pakietu yalmip do rozwiązywania problemów optymalizacji całkowitoliczbowej.

Programowanie całkowitoliczbowe

Ogólnie programowanie całkowitoliczbowe możemy zapisać następująco:

$$\begin{aligned} &\max c^T x \\ &\quad Ax = b \\ &\quad x \geq 0 \text{ i całkowitoliczbowy} \end{aligned} \tag{1}$$

lub

$$\begin{aligned} &\max \sum_{j=1}^{m} c_j x_j \\ &\quad \sum_{j=1}^{m} a_{i,j} x_j = b_j \quad i \in 1, 2, \ldots, n \\ &\quad x_j \geq 0 \text{ i całkowitoliczbowy}, \quad j \in 1, 2, \ldots, m \end{aligned} \tag{2}$$

Wszystkie elementy w c, A, b z założenia są liczbami całkowitymi. To założenie gwarantuje, ze wszystkie elementy są liczbami wymiernymi, ponieważ przemnożenie funkcji celu przez dowolna całkowita liczbę dodatnia oraz pomnożenie jakiegokolwiek ograniczenia przez liczbę różna od zera nie zmieni charakteru rozpatrywanego zadania. Zadanie programowania całkowitoliczbowego liniowego może być przedstawiane w wielu postaciach. Najczęściej jest przedstawiane w postaci (1) i (2). Wymienione zapisy wykorzystywane są przez oprogramowanie (solver) do rozwiązywania problemu programowanie całkowitoliczbowego. Na potrzeby laboratorium należy pamiętać dwie główne kwestie:

1. Wszystkie elementy problemu programowania całkowitoliczbowego są liczbami całkowitymi

2. Funkcja celu $c^T x$ może być minimalizowana lub maksymalizowana. W celu maksymalizacji funkcji celu należy zmienić znak $-c^T x$.

Instalacja pakietu yalmip

W celu wygodnego i czytelnego rozwiązywania problemów optymalizacji całkowitoliczbowej na potrzeby laboratorium zostanie wykorzystaniem oprogramowania Matlab w wersji 2015a lub nowszej wraz z pakietem yalmip. Pakiet yalmip jest pareserem umożliwiającym wygodny zapis problemów optymalizacyjnych a następnie przekształcenie ich do postaci akceptowalnej przed wybrany solver. Zostanie to pokazane później w przykładach. W celu zainstalowania pakietu yalmip należy go pobrać z oficjalnej strony `https://yalmip.github.io/download/`. Po pobraniu należy wypakować pobrane archiwum i dodać ścieżki do wszystkich folderów w programie Matlab. Aby sprawdzić poprawność instalacji należy wywołać komendę *yalmiptest*. Przykładowy wyniki wywołania może wyglądać następująco:

```
+------------------------------------------------------------------------------------+
|                   Test|   Solution|                                Solver message|
+------------------------------------------------------------------------------------+
|   Core functionalities|        N/A|                    Successfully solved (YALMIP)|
|                     LP|    Correct|             Successfully solved (GUROBI-GUROBI)|
|                     LP|    Correct|             Successfully solved (GUROBI-GUROBI)|
|                     QP|    Correct|             Successfully solved (GUROBI-GUROBI)|
|                     QP|    Correct|             Successfully solved (GUROBI-GUROBI)|
|                   SOCP|    Correct|             Successfully solved (GUROBI-GUROBI)|
|                   SOCP|    Correct|             Successfully solved (GUROBI-GUROBI)|
|                   SOCP|    Correct|             Successfully solved (GUROBI-GUROBI)|
|                    SDP|    Correct|                    Successfully solved (LMILAB)|
|                    SDP|    Correct|                    Successfully solved (LMILAB)|
|                    SDP|    Correct|                    Successfully solved (LMILAB)|
|                    SDP|    Correct|                    Successfully solved (LMILAB)|
|                 MAXDET|    Correct|                    Successfully solved (LMILAB)|
|                 MAXDET|    Correct|                    Successfully solved (LMILAB)|
|           Infeasible LP|        N/A|   Either infeasible or unbounded (GUROBI-GUROBI)|
|           Infeasible QP|        N/A|   Either infeasible or unbounded (GUROBI-GUROBI)|
|          Infeasible SDP|        N/A|                     Infeasible problem (LMILAB)|
|       Moment relaxation|    Correct|                    Successfully solved (LMILAB)|
|          Sum-of-squares|        N/A|                     Infeasible problem (LMILAB)|
|            Bilinear SDP|        N/A|                              No suitable solver|
+------------------------------------------------------------------------------------+
```

Jeśli instalacja przebiegła pomyślnie, możemy teraz przejść do przykładu oraz zadań.

Przykład

Na wydziale obróbki mamy dwie maszyny (M1, M2) oraz dwóch obsługujących je robotników (R1, R2). Znamy wydajność każdego robotnika na poszczególnych stanowiskach według tabeli 3. Wydajność tą określa liczba detali, które dany robotnik może wykonać na danej maszynie w ciągu jednej godziny. Należy ustalić taki przydział robotników do poszczególnych stanowisk, aby łączna wydajność całego zespołu była maksymalna przy czym:

- **Ograniczenie 1:** Na danej maszynie może pracować jeden i tylko jeden pracownik
- **Ograniczenie 2:** Każdy pracownik może pracować na jednej i tylko jednej maszynie

Tablica 1: Wydajność każdego robotnika na poszczególnych stanowiskach

$W_{i,j}$	R1	R2
M1	6	7
M2	12	6

Wyżej wymieniony problem, jest problemem optymalnego przydziału który możemy rozwiązać za pomocą następującego problemu optymalizacji:

Funkcja celu: Maksymalizacja łącznej wydajności całego zespołu:

$$\max \sum_{i=1}^{n} \sum_{j=1}^{m} W_{i,j} x_{i,j} \tag{3}$$

Ograniczenie 1: Na danej maszynie może pracować jeden i tylko jeden pracownik:

$$\sum_{j=1}^{m} x_{i,j} = 1 \quad i \in 1, 2, \ldots, n \tag{4}$$

Ograniczenie 2: Każdy pracownik może pracować na jednej i tylko jednej maszynie:

$$\sum_{i=1}^{n} x_{i,j} = 1 \quad j \in 1, 2, \ldots, m \tag{5}$$

Ograniczenie 3: Ograniczenia co do wartości zmiennych decyzyjnych:

$$x_{i,j} \in \{0, 1\} \quad i \in 1, 2, \ldots n \quad j \in 1, 2 \ldots, m, \tag{6}$$

co oznacza iż

$$x_{i,j} = \begin{cases} 1, & \text{jeżeli } j\text{-ty pracownik został przydzielony do } i\text{-tego stanowiska} \\ 0, & \text{w przeciwnym razie} \end{cases}$$

UWAGA: Zmienna x jest zmienną decyzyjną której wartość (wartości) zostaną ustalone po rozwiązaniu problemu optymalizacji przez solver. Wartości jakie może przyjmować zmienna x to zero lub jeden i to solver decyduje jaka to będzie wartość przy uwzględnieniu ograniczeń oraz charakteru funkcji celu.

Przedstawiony problem optymalizacji może być niejasny na pierwszy rzut oka. W celu wyjaśnienia poszczególnych elementów pomnimy funkcję celu (3) (wrócimy do niej w dalszej części opisu) i przejdźmy do ograniczeń. Według tabeli wydajności, ilość zmiennych decyzyjnych wynosi cztery, odpowiednio:

$$x_{i,j} = [x_{1,1},\, x_{1,2},\, x_{2,1},\, x_{2,2}] \quad i \in 1,2 \quad j \in 1,2. \tag{7}$$

W zapisie tabelarycznym: Pierwsze ograniczenie wprowadza warunek iż na danej maszynie

$W_{i,j}$	R1	R2
M1	6	7
M2	12	6

$\longrightarrow$

$x_{i,j}$	$j=1$	$j=2$
$i=1$	$x_{1,1}$	$x_{1,2}$
$i=2$	$x_{2,1}$	$x_{2,2}$

$i \in 1,2,\ldots,n$ może pracować jedn i tylko jeden pracownik. Tak więc mamy (suma po kolumnach):

- dla $i=1$ $x_{1,1} + x_{1,2} = 1$
- dla $i=2$ $x_{2,1} + x_{2,2} = 1$

Drugie ograniczenie wprowadza warunek iż każdy pracownik $j \in 1,2,\ldots,m$ może pracować na jednej i tylko jednej maszynie. Tak więc mamy (suma po wierszach):

- dla $j=1$ $x_{1,1} + x_{2,1} = 1$
- dla $j=2$ $x_{1,2} + x_{2,2} = 1$

Powyższy opis ograniczeń można zapisać tabelarycznie: Ograniczenie trzecie zostało już

$x_{i,j}$	$j=1$	$j=2$	
$i=1$	$x_{1,1}$	$x_{1,2}$	$\sum_{j=1}^{m} x_{1,j}$
$i=2$	$x_{2,1}$	$x_{2,2}$	$\sum_{j=1}^{m} x_{2,j}$
	$\sum_{i=1}^{n} x_{i,1}$	$\sum_{i=1}^{n} x_{i,2}$	

opisane. Ostatnim elementem jest funkcja celu która jest sumą iloczynów poszczególnych

wydajności oraz zmiennych decyzyjnych która powinna być jak największa (maksymalizujemy wydajność):

$$\max \sum_{i=1}^{n} \sum_{j=1}^{m} W_{i,j} x_{i,j} \longrightarrow \max 6x_{1,1} + 7x_{1,2} + 12x_{2,1} + 6x_{2,2} \tag{8}$$

Podsumowując, przedstawiony wcześniej problem optymalizacji można zapisać następująco:

Funkcja celu: Maksymalizacja łącznej wydajności całego zespołu:

$$\max 6x_{1,1} + 7x_{1,2} + 12x_{2,1} + 6x_{2,2} \tag{9}$$

Ograniczenie 1: Na danej maszynie może pracować jeden i tylko jeden pracownik:

$$x_{1,1} + x_{1,2} = 1 \tag{10}$$

$$x_{2,1} + x_{2,2} = 1 \tag{11}$$

Ograniczenie 2: Każdy pracownik może pracować na jednej i tylko jednej maszynie:

$$x_{1,1} + x_{2,1} = 1 \tag{12}$$

$$x_{1,2} + x_{2,2} = 1 \tag{13}$$

Ograniczenie 3: Ograniczenia co do wartości zmiennych decyzyjnych:

$$x_{i,j} \in \{0, 1\} \quad i \in 1, 2, \ldots n \quad j \in 1, 2 \ldots, m, \tag{14}$$

W wyniku rozwiązania problemu optymalizacji otrzymujemy następujące wyniki: Uzy-

$W_{i,j}$	R1	R2
M1	6	7
M2	12	6

$\longrightarrow$

$x_{i,j}$	$j = 1$	$j = 2$
$i = 1$	0	1
$i = 2$	1	0

$\longrightarrow$ Funkcja celu = 19

skane rozwiązanie spełnia ograniczenia postawione na etapie formułowania problemu optymalizacji. Dodatkowo widać iż w tym przykładzie wyniki jest oczywisty. Została wybrana taka konfiguracja która odpowiada największej wydajności danego robotnika na danej maszynie.

Zadania do wykonania w ramach ćwiczenia:

1. Wykonaj modyfikację kodu z podanego powyżej zadania aby sumowanie odbywało się w pętli `for`. Jeśli to możliwe wykorzystaj funkcję `sum` aby ograniczyć jak najbardziej pętle.

2. Na trzech maszynach M1, M2, M3 mogą być produkowane trzy różne produkty P1, P2, P3. Czasy obróbki w godzinach podano w poniższej tabeli. Chcemy, aby każdy

Tablica 2: Czasy obróbki w godzinach danego produktu

$W_{i,j}$	M1	M2	M3
P1	6	7	3
P2	12	6	2
P3	2	7	4

produkt był produkowany przez dokładnie jedną maszynę i aby każda maszyna produkował dokładnie jeden produkt, przy czym czas obróbki ma być jak najmniejszy.

3. Należy przewieź jak największą ilość przedmiotów o jak największej wartości w trzech skrzyniach. Łączna waga skrzyń nie może być większa jak 100[kg]. W poniższej tabeli podano wagę poszczególnego przedmiotu oraz wartość w zł.

Tablica 3: Masa oraz wartość poszczególnego przedmiotu

waga	15	18	12	10	14	19	15	12	14
wartość	300	10	50	60	70	150	500	80	5

A Kod w programie Matlab dla przykładu pierwszego

```
clc,close all,clear;
%% ************************************************************************
%% Wstęp do programowania całkowitoliczbowego oraz pakietu
%% yalmip
%% ************************************************************************
% Tabela wydajności danego pracownika
W = [6      7;...
     12     6];

% Informacja o ilości maszyn oraz robotników -
% Mamy n stanowisk i m pracowników
n = size(W,1);
m = size(W,2);

%  Zmienne decyzyjne
x = intvar(n,m);

% Funkcja celu ()
Fun = W(1,1)*x(1,1) + W(1,2)*x(1,2) + W(2,1)*x(2,1) + W(2,2)*x(2,2);

% Ograniczenia
Constraints = [];

% 1)  Na danej maszynie może pracować jeden i tylko jeden pracownik
Constraints = [Constraints, x(1,1) + x(1,2) == 1];
Constraints = [Constraints, x(2,1) + x(2,2) == 1];

% 2) Każdy pracownik może pracować na jednej i tylko jednej maszynie
Constraints = [Constraints, x(1,1) + x(2,1) == 1];
Constraints = [Constraints, x(2,1) + x(2,2) == 1];
```

```
% 3) Ograniczenia co do wartości zmiennych decyzyjnych
Constraints = [Constraints, 0 <= x(1,1) <= 1];
Constraints = [Constraints, 0 <= x(1,2) <= 1];
Constraints = [Constraints, 0 <= x(2,1) <= 1];
Constraints = [Constraints, 0 <= x(2,2) <= 1];

% Optymalizacja
ops = sdpsettings('solver','intlinprog','verbose',0);
res = solvesdp(Constraints,-Fun,ops);
double(Fun)
double(x)
```