European Social Fund Prague & EU: We invests in your future.

Combinatorial optimisation
Seminar No. 2
Integer linear programming

Zdeněk Bäumelt, Přemysl Šůcha (baumezde@fel.cvut.cz, suchap@fel.cvut.cz)

February 18, 2013

1 Integer linear programming

A number of problems belonging to the area of combinatorial optimisation can be modeled and solved using *Integer Linear Programming* (ILP).

Definition 1.1 *An Integer Linear Programming task is given by a matrix of constraints* $A \in \mathbb{R}^{m \times n}$ *and vectors* $b \in \mathbb{R}^m$ *and* $c \in \mathbb{R}^n$*. The goal is to find a vector* $x \in \mathbb{Z}^n$ *such that* $Ax \leq b$ *and a criterion* $c^T x$ *is minimal.*

In comparison to *Linear Programming* (LP), ILP variables are limited to integers [1]. Generally, one is not able to solve the ILP tasks using LP continuous variables rounded to the nearest integers due to several reasons. Firstly, we have no guarantee of the optimality of the solution, i.e. the optimal solution can be missed. Moreover, the feasibility of the solution cannot be warranted, i.e. some of the constraints of the problem can violated after this rounding. Therefore, the tasks having integer variables have to be solved by ILP, which is more expensive from the time complexity point of view. The ILP (unlike LP) is a NP-hard problem, i.e. polynomial-time algorithms are not known yet. The ILP tasks can be solved by enumerative methods, the Branch&Bound algorithm or Cutting Planes methods (see more in [2]).

1.1 ILP and totally unimodular matrix

Even though the ILP is NP-hard it can be solved in polynomial time if some conditions are satisfied, e.g. the matrix A has to be *totally unimodular*.

Definition 1.2 *A matrix* A *is totally unimodular if each subdeterminant of* A *is equal to* $+1$*,* 0 *or* -1*.*

Theorem 1.1 *If the matrix of constraints* A *is totally unimodular and* $b \in \mathbb{Z}^m$*, the ILP task can be solved by modified simplex method to obtain a solution* x *such that* $x \in \mathbb{Z}^n$*.*

Theorem 1.2 *The ILP task having a totally unimodular matrix of constraints* A *and* $b \in \mathbb{Z}^m$ *is solvable in polynomial time.*

Theorem 1.3 *Let* A *be an* $m \times n$ *matrix such that*

1. *each entry* $a_{ij} \in \{0, 1, -1\}$*,* $i = 1, \ldots, m$*,* $j = 1, \ldots, n$
2. *each column of* A *contains either at most one nonzero entry or just two nonzero entries equal to* $+1$ *and* -1

Then the matrix A *is totally unimodular.*

Pay attention to the structure and the content of the matrix A. When this matrix does not satisfy Theorem 1.3, we are not able to claim that this matrix is not totally unimodular. However, the equation $Ax < b$ can be transformed by the Gauss Elimination Method (GEM) to an equation $A'x < b'$. If $b' \in \mathbb{Z}^n$ is preserved by the GEM transormation and the matrix A' fulfills Theorem 1.3, then A' is totally unimodular. Futhermore, in this case the original matrix A is totally unimodular, i.e. one is able to solve this problem with respect to Theorems 1.2 and 1.3.

2 Call center scheduling problem

2.1 Problem Description

A call center needs to create a cyclic (i.e. numbers of the shifts at particular hours are same on every day) daily schedule of the *shifts* for its employees [3]. Let b be a vector of a *personnel demand* in each hour during the day, i.e. b_i is defined $\forall i = 1, 2, \ldots, 24$. The personnel demand b_i determines the minimal number of shifts (i.e. employees who will be assigned to these shifts) at hour i, e.g. b_{10} expresses the minimal number of shifts running between 9 a.m. and 10 a.m. The objective of this problem is to obtain the cyclic daily schedule of the shifts such that the total number of the shifts used to cover the personnel demand in a day is minimized. The shift may start at arbitrary full hour, its length is set to 8 hours.

The ILP model of this problem is based on a variable x such that x_i represents a number of the shifts starting at hour $(i-1)$. The constraints presented in the model (1) express that hour i is covered by shifts starting from the hour $(i-7)$ to the hour i.

$$\begin{aligned} & \min \sum_{i=1}^{24} x_i \\ \text{subject to} \quad & \\ & x_{i+17} + \cdots + x_{24} + x_1 + \cdots + x_i \geq b_i, \qquad \forall i = 1 \ldots 7 \\ & x_{i-7} + x_{i-6} + \cdots + x_i \geq b_i, \qquad \forall i = 8 \ldots 24 \\ & x_i \geq 0 \qquad \forall i \end{aligned} \tag{1}$$

2.2 ILP in TORSCHE – an example

The ILP task can be solved in TORSCHE by the function `ilinprog` (see more by typing `help ilinprog`). How to use this function is illustrated in the following example.

```
>> sense=1;               %sense of optimization: 1=minimization, -1=maximization
>> b = [ 2  1]';          %vector b
>> A = [ 1 -1; ...        %matrix A
>>       0  1]
>> c = [ 1  1]';          %vector c
>> ctype = ['L','E']';    %constraint type: 'E'="=", 'L'="<=", 'G'=">="
>> lb = [0,0]';           %lower bound of the variables
>> ub = [2,2]';           %upper bound of the variables
>> vartype = ['C','C']';  %variable type: 'C'=continuous, 'I'=integer

%optimization parameters
>> schoptions=schoptionsset('ilpSolver','glpk','solverVerbosity',2);

%call command for ILP
>> [xmin,fmin,status,extra] = ilinprog(schoptions,sense,c,A,b,ctype,lb,ub,vartype);

%show the solution
>> if(status==1)
>>    disp('Solution: '); disp(xmin)
>>    disp('Objective function: '); disp(fmin)
>> else
>>    disp('No feasible solution found!');
>> end;
```

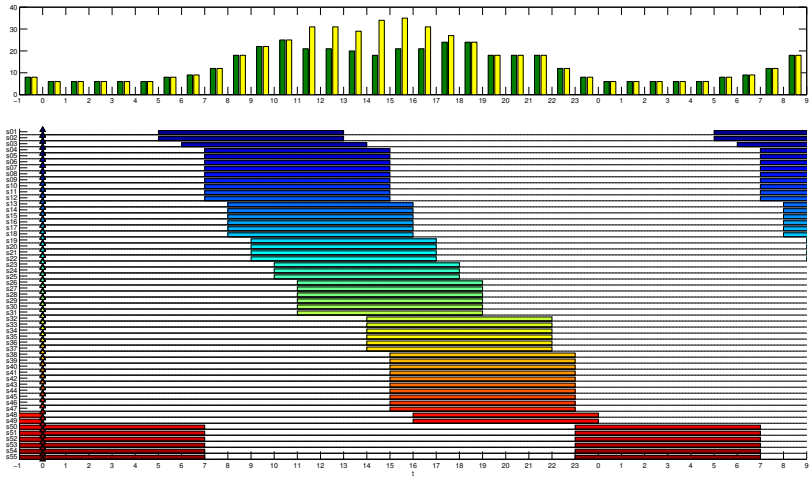

Figure 1: The coverage of the personnel demand b and the final daily schedule of the shifts

A seminar assignment: Create the ILP model for this problem. Namely, write the matrix A and the vectors b and c on the paper. The personnel demand b is given as follows:

```
>> b = [6 6 6 6 6 8 9 12 18 22 25 21 21 20 18 21 21 24 24 18 18 18 12 8]';
```

The cyclic daily schedule of the used shifts is illustrated at the bottom of Fig. 1. You can notice the large surplus of the shifts in comparison to the personnel demand between 12 a.m. and 6 p.m. How to minimize this difference between the personnel demand and its coverage is handled in Sec. 3.

3 A homework assignment

The problem described in Sec. 2.1 has to be modified when the lack of the employees occurs and we are not able to find the feasible solution of this problem. One way is to accept the lack of employees in particular intervals. However, our objective is to find a daily schedule of shifts such that the difference of the personnel demand and its coverage is minimized, i.e. we try to cover the personnel demand as precisely as possible. Therefore, the objective function in the model should be modified as follows.

$$\min \sum_{i=1}^{24} \left| \sum_{j=i-7}^{i} x_{((j-1) \bmod 24)+1} - b_i \right| \tag{2}$$

Unfortunately, the absolute value function (Eq. 2) cannot be used in the ILP model directly. It is necessary to substitute it by an auxiliary variable z that will be bound by the constraints representing the personnel demand coverage. A general transformation is defined by Eq. 3, where v, z are vectors of variables and q is a vector of constants.

$$\begin{array}{lcccc} & \min \sum \left| v - q \right| & \longrightarrow & \min \sum z & \longrightarrow & \min \sum z \\ \text{subject to} & & & & & \\ & \vdots & & \vdots & & \vdots \\ & & & \left| v - q \right| = z & & v - q \leq z \\ & & \longrightarrow & & \longrightarrow & q - v \leq z \\ & & & z \geq 0 & & z \geq 0 \end{array} \tag{3}$$

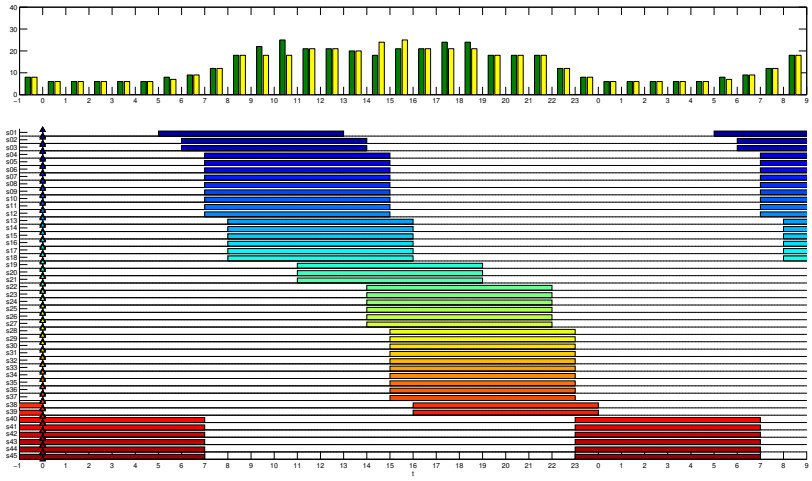

Figure 2: The coverage of the personnel demand b and the final daily schedule of the shifts

A homework assignment: Consider the same vector of the personnel demand b as in the seminar assignment. Use the function `ilinprog` (see the example in Sec. 2.2) from the TORSCHE toolbox. The matrix A must be generated by the code. Firstly, solve the problem described by the ILP model Eq. 1. Secondly, modify the ILP model of the problem with respect to the Sec. 3. Show both solutions, i.e. display the number of shifts (yellow colour) together with the personnel demand b (green colour) similarly like at the top of Fig. 1 and 2, where one of the optimal solutions for each ILP model are depicted. A Matlab standard function `bar` with the default colouring of the bars can be used to show these figures.

References

[1] R. Vanderbei, *Linear Programming : Foundations and Extensions*. http://www.princeton.edu/~rvdb/LPbook: Princeton University, second ed., 2001.

[2] B. H. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*. Springer, third ed., 2006.

[3] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall; United States Ed edition, 1993.

[4] J. Demel, *Grafy a jejich aplikace*. Academia, second ed., 2002.

[5] J. B. Orlin, "Introduction to optimization." MIT OpenCourseWare, 2004.