Technische Universität Berlin
WiSe 2012

Prof. Andreas Bley

Computational mixed-integer programming
1. Set of programming exercises: ILP modeling

Preliminaries

Necessary software

The latest version of the ZIBOPTSUITE (including the programs and programming libraries ZIMPL and SCIP) and online documentations of ZIMPL and SCIP can be found at `http://zibopt.zib.de`.

In the computer pool the ZIBOPTSUITE is installed locally in the directory `/usr/site-local/ziboptsuite-2.0.1`. In later exercises we will switch to version 3.0.0 of the suite, which then can be found in the directory `/usr/site-local/ziboptsuite-3.0.0`.

When working in the computer pool, the most recent versions of the programs `zimpl` and `scip` can be executed by just calling `zimpl` or `scip` in the command line. On your own computer, you may add the directories`/usr/local/ziboptsuite-XXX/zimpl-XXX/bin` and `/usr/local/ziboptsuite-XXX/scip-XXX/bin` to your PATH environment variable (or add softlinks to the binary files in these directories) to execute the binaries more easily (where XXX needs to be replaced by the apropriate version).

Documentation

A documentation of the modeling language ZIMPL can be found at `/usr/size-local/ziboptsuite-2.0.1/zimpl-3.1.0/doc/zimpl.pdf`. It is very helpful to read this documentation. A documentation of `scip` (including tutorials for many application and programming scenarios) can be found at te scip website `http://scip.zib.de`.

Exercises

All exercises will be available at the course website at the start of the programming exercise session. The course website can be found via `http://www.coga.tu-berlin.de/v-menue/lehre/ws12/vl_computational_integer_programming/`.

The package `exercise01.tgz` contains the 1st exercise sheet, the template `zpl`-files for the exercise, and benchmark instances to test and compare your models.

Exercise 1 – Bin packing problem

In this exercise, we will consider the *bin packing problem*. We will do parts of this exercise together.

Task: Write a ZIMPL Model for each of the following two formulations. Then compare how the different formulations perform by solving them using SCIP (for many test instances!).

The data parsing part is already implemented in the zpl-files. This part reads the original problem data files and creates the following sets and parameters:

- `capacity` this parameter stores the bin capacity c which is the same for all bins
- `nitems` gives you the number of items to pack
- `I` is an index set for the items, i.e. `I` $= \{1, \ldots, $`nitems`$\}$

- `sizes` is a map which maps the item index to the size s_i of item i

Instances to test your models can be found in the `bin-zimpl/Data` folder.

Note that, in contrast to the model presented in the lecture, items have no multiplicities, i.e., each item must be packed exactly once. Items of the same size occur as multiple individual items in the input data.

a) Open the file `bin-zimpl/assignment.zpl` and model the bin packing problem using the assignment formulation. This formulation is based on the following decision variables:
   - "Is item i assigned to bin b?" ($y_{i,b}$)
   - "Is bin b in use?" (x_b)

$$
\begin{aligned}
\min \quad & \sum_{b \in B} x_b && && && \text{(ASP)} \\
\text{s.t.} \quad & \sum_{b \in B} y_{i,b} && \geq 1 && \forall\, i \in I && \\
& \sum_{i \in I} s_i \cdot y_{i,b} && \leq c \cdot x_b && \forall\, b \in B && \\
& y_{i,b} && \leq x_b && \forall\, i \in I, b \in B && (*) \\
& \mathbf{x} \in \{0,1\}^B && && && \\
& \mathbf{y} \in \{0,1\}^{I \times B} && && &&
\end{aligned}
$$

Study the impact of adding or removing inequality (*) to/from the model on the value of the linear programming relaxation, on the problem size, on the time necessary to solve the linear relaxation, and on the total solution time of the integer program.

b) Open the file `bin-zimpl/setcovering.zpl` and model the bin packing problem as an set-covering formulation. This formulation has a decision variable for every feasible packing.

1. First, you should define the set of feasible packings P.
   Hint: You can (for example)
   i. use the ZIMPL function `subset` or `powerset` to create all subsets of items,
   ii. then define an `indexset` to iterate over all these subsets, and
   iii. finally define a subset of the indexset that contains only those subsets that define a feasible packing (i.e., fit into a bin).
2. Define the required decision variables x_p for each feasible packing p.
3. Now you can model the constraints, so that each item has to be part of at least one packing.

$$
\begin{aligned}
\min \quad & \sum_{p \in P} x_p && && \text{(SCP)} \\
\text{s.t.} \quad & \sum_{p \in P} a_{i,p} \cdot x_p && \geq 1 && \forall\, i \in I \\
& \mathbf{x} \in \{0,1\}^P && &&
\end{aligned}
$$

The $a_{i,p}$ denote whether item i is in the packing p or not, i.e.

$$
a_{i,p} = \begin{cases} 1 & \text{if item } i \text{ is in packing } p \\ 0 & \text{otherwise}\,. \end{cases}
$$

Exercise 2 – Travelling salesman problem

Task: Write a ZIMPL Model for each of the following ATSP formulations. Then compare how the different formulations perform by solving them in SCIP.

Again, the data parsing part in ZIMPL is already given:

- `V` is the set of nodes
- `A` is the set of arcs
- `d` gives you the length of each arc

Instances to test your models can be found in the folder `atsp-zimpl/Data`.
Helpful ZIMPL functions for the subset calculations: `powerset` and `indexset`

a) Open the file `atsp-zimpl/subtour.zpl` and model the ATSP with subtour elimination constraints:

$$
\begin{aligned}
&\min\ c^T x && && && \text{(SUB)}\\
&\text{s.t.}\ && x(\delta^+(v)) && = 1 && \forall\ v \in V\\
& && x(\delta^-(v)) && = 1 && \forall\ v \in V\\
& && x(A(S)) && \leq |S| - 1 && \forall\ \emptyset \neq S \subsetneq V\\
& && 0 \leq x_a \leq 1 && && \forall\ a \in A\\
& && \mathbf{x} \in \mathbb{Z}^A
\end{aligned}
$$

b) Open the file `atsp-zimpl/cut.zpl` and model the ATSP with cut constraints:

$$
\begin{aligned}
&\min\ c^T x && && && \text{(CUT)}\\
&\text{s.t.}\ && x(\delta^+(v)) && = 1 && \forall\ v \in V\\
& && x(\delta^-(v)) && = 1 && \forall\ v \in V\\
& && x(\delta^+(S)) && \geq 1 && \forall\ \emptyset \neq S \subsetneq V\\
& && 0 \leq x_a \leq 1 && && \forall\ a \in A\\
& && \mathbf{x} \in \mathbb{Z}^A
\end{aligned}
$$

c) Open the file `atsp-zimpl/mtz.zpl` and model the formulation introduced by Miller, Tucker and Zemlin:

$$
\begin{aligned}
&\min\ c^T x && && && \text{(MTZ)}\\
&\text{s.t.}\ && x(\delta^+(v)) && = 1 && \forall\ v \in V\\
& && x(\delta^-(v)) && = 1 && \forall\ v \in V\\
& && u_v - u_w + (n-1)\, x_{(v,w)} && \leq n - 2 && \forall\ (v,w) \in A, w \neq 1\\
& && 0 \leq x_a \leq 1 && && \forall\ a \in A\\
& && 1 \leq u_v \leq n - 1 && && \forall\ v \in V \setminus \{1\}\\
& && u_1 = 0\\
& && \mathbf{x} \in \mathbb{Z}^A\\
& && \mathbf{u} \in \mathbb{Z}^V
\end{aligned}
$$

d) For the last formulation edit the file `atsp-zimpl/vvw.zpl` and model the van Vyve-Wolsey extended formulation. Test your model for different neighborhood sizes k.

$$
\begin{aligned}
&\min\ c^T x && && && (\mathrm{VVW}_k)\\
&\text{s.t.}\ && x(\delta^+(v)) && = 1 && \forall\ v \in V\\
& && x(\delta^-(v)) && = 1 && \forall\ v \in V\\
& && u_v - u_w + (n-1)\, x_{(v,w)} && \leq n-2 && \forall\ (v,w) \in A, w \neq 1\\
& && f^l(\delta^+(v)) - f^l(\delta^-(v)) && = \delta_{vl} && \forall\ l, v \in V_l\\
& && f^l_a \leq x_a && && \forall\ l, a \in A_l\\
& && 0 \leq x_a \leq 1 && && \forall\ a \in A\\
& && 0 \leq f^l_a \leq 1 && && \forall\ l, a \in A\\
& && 1 \leq u_v \leq n-1 && && \forall\ v \in V \setminus \{1\}\\
& && u_1 = 0 && &&\\
& && \mathbf{x} \in \mathbb{Z}^A && &&\\
& && \mathbf{u} \in \mathbb{Z}^V && &&
\end{aligned}
$$

For every node l the set V_l contains the k nodes closest to l, with respect to the following distance function: $dist(l, v) := c_{l,v} + c_{v,l}$.
The arc set A_l contains all arcs incident to V_l, i.e.

$$A_l = A(V_l) \cup \delta^-(V_l) \cup \delta^+(V_l).$$