Graph Optimization
Lab session 1 - Introduction

Giuliana Carello

2023/2024

Solving models

How to solve a problem, using LP, MIP or IP formulations?

$\Longrightarrow$ use a SOLVER

Given the problem:

- formulate the model
- write it in a suitable language ($\Longrightarrow$ MODELER)
- solve it with the solver
- analyze the solution (and check the model)
- build a strategy

The solvers

- A *solver* is a software that takes as input an optimization problem description (i.e., model+data) and gives as output the optimal solution (if one exists)
- it is important
  - to distinguish *models* and *data* since the same model can be used with many different data
  - to write the model with a user friendly *modeling language*

$\Longrightarrow$ we use *algebraic modeling generators*

Algebraic modeling generators: goals

- To describe complex problems with a simple language, which is
  - high level (comprehensible to users)
  - formally structured ("comprehensible to" softwares)
- To use a tool independent of the solver
- To distinguish between the logic structure of the problem (i.e., the model) and the numerical data (i.e., the instances of the same problem)

Software

- Solver
  - NLP $\Longrightarrow$ MINOS
  - LP and ILP $\Longrightarrow$ CPLEX, GUROBI, XPRESS, GLPK
- Modeling language $\Longrightarrow$ AMPL (www.ampl.com)
  - file .mod $\Longrightarrow$ model
  - file .dat $\Longrightarrow$ problem data
  - file .run $\Longrightarrow$ scripting commands

The knapsack problem

A set of items I is given. Each item $i \in I$ has a profit p_i and a weight w_i. A knaosack is given, as well, whose capacity is B. A feasible subset of items is such that the sum of weights of the items in the subset does not exceed the knapsack capacity. We have to select the maximum profit feasible subset.

$$\begin{aligned}
\max \quad & \sum_{i \in I} p_i x_i \\
& \sum_{i \in I} w_i x_i \leq B \\
& x_i \in \{0, 1\} \qquad \forall i \in I
\end{aligned}$$

Knapsack problem: instance

$$
\begin{aligned}
\max \quad & 10x_1 + 12x_2 + 5x_3 + 7x_4 + 9x_5 \\
\text{s.t.} \quad & 5x_1 + 8x_2 + 6x_3 + 2x_4 + 7x_5 \leq 14 \\
& x_1, \ldots, x_5 \in \{0, 1\}
\end{aligned}
$$

AMPL syntax: variables

Model	AMPL syntax
$x_1 \in \{0, 1\}$	`var x_1, binary;`
$x_2 \in \{0, 1\}$	`var x_2, binary;`
$x_3 \in \{0, 1\}$	`var x_3, binary;`
$x_4 \in \{0, 1\}$	`var x_4, binary;`
$x_5 \in \{0, 1\}$	`var x_5, binary;`
$\max 10x_1 + 12x_2 + 5x_3 + 7x_4 + 9x_5$	
$5x_1 + 8x_2 + 6x_3 + 2x_4 + 7x_5 \leq 14$	

- variables are introduced by `var`
  The domain can be:
  - `binary`
  - `integer`
  - `>=0`

Each statement must end with ";"

AMPL syntax: objective function

Model	AMPL syntax
$x_1 \in \{0,1\}$	`var x_1, binary;`
$x_2 \in \{0,1\}$	`var x_2, binary;`
$x_3 \in \{0,1\}$	`var x_3, binary;`
$x_4 \in \{0,1\}$	`var x_4, binary;`
$x_5 \in \{0,1\}$	`var x_5, binary;`
$\max 10x_1 + 12x_2 + 5x_3 + 7x_4 + 9x_5$	`maximize profit:`
	`10*x_1 + 12*x_2 + 5*x_3`
	`+ 7*x_4 + 9*x_5;`
$5x_1 + 8x_2 + 6x_3 + 2x_4 + 7x_5 \leq 14$	

- Objective function is introduced by `maximize` (or `minimize`)
- Objective function must have a label
- Product is denoted with "$*$ "

Each statement must end with ";"

AMPL syntax: constraints

Model	AMPL syntax
$x_1 \in \{0,1\}$	`var x_1, binary;`
$x_2 \in \{0,1\}$	`var x_2, binary;`
$x_3 \in \{0,1\}$	`var x_3, binary;`
$x_4 \in \{0,1\}$	`var x_4, binary;`
$x_5 \in \{0,1\}$	`var x_5, binary;`
$\max 10x_1 + 12x_2 + 5x_3 + 7x_4 + 9x_5$	`maximize profit:`
	`10*x_1 + 12*x_2 + 5*x_3`
	`+ 7*x_4 + 9*x_5;`
$5x_1 + 8x_2 + 6x_3 + 2x_4 + 7x_5 \leq 14$	`s.t.  capacity:5*x_1+8*x_2`
	`+6*x_3+2*x_4+7*x_5 <= 6;`

- A constraint is introduced by `s.t.`
- A constraint must have a label
- $\leq$ is denoted as "<= "
- $\geq$ is denoted as ">= "

Each statement must end with ";"

AMPL syntax: solving the problem

command	AMPL syntax
select a solver	`option solver gurobi;`
start the solution process	`solve;`
show the solution	`display ;`

Each command must end with ";"

Output:

```
Gurobi 10.0.1:  Gurobi 10.0.1:  optimal solution;
objective 26
0 simplex iterations
;
```

AMPL syntax: file `.mod` and `.run`

Beside using the command line, we can use a `.mod` file and `.run` file:

`.mod` file contains the description of the model

`.dat` file contains the data of the problem

`.run` file includes all the commands

In the `.run` file:

command	AMPL syntax
clean data and values	`reset;`
load the model description	`model knapsack.mod;`
load the data	`data knapsack.dat;`
select a solver	`option solver gurobi;`
start the solution process	`solve;`
show the solution	`display x;`

From command line: `include knapsack.run;` to run the solution process

Model (.mod file)

It includes the *declaration* of the parameters:

- sets: `set`
  - elements: `in` (e.g., i `in` S)
  - subsets: `within` (e.g., S_0 `within` S)
- data: `param` (single parameter, arrays and matrices)

Model (.mod file)

and the declaration of the main parts of the model (variables, objective function and constraints)

- variables: `var`
  - `binary`
  - `integer`
  - `>=0`
- the objective function: `minimize`, `maximize`
- the constraints: `subject to`

Data (.dat file)

It includes the *definition* of

- sets and subsets
- data

AMPL commands (.run file)

- `quit;` to exit
- `option solver solvername;` to choose a solver
- `model file.mod;` to load a model
- `data file.dat;` to load the data
- `include file.run;` to run a script
- `solve;`
- `reset;` to reset model and data
- `reset data;` to reset data

AMPL commands (.run file)

- `display`: to see the solution
  - variable or parameter: `display VarName;`
  - dual variable: `display ConstraintName.dual;`
  - slack : `display ConstraintName.slack;`

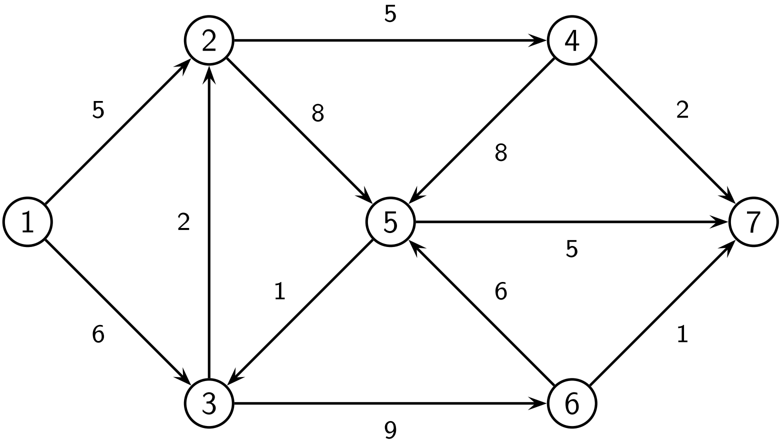
AMPL commands (.run file)

- `:=` : assign a value to a parameter that cannot be changed
- `default` : assign an initial value to a parameter that can be changed
- `let :=` : assign a value to a parameter that can be changed
- `for` : for statement
- `repeat` : repeat statement
- `if` : if statement
- `printf` : print

The AMPL book is available at
https://ampl.com/resources/the-ampl-book/

Maximum flow

Solve the following max flow problem, where, for each arc, the capacity is reported, with AMPL and Gurobi:



Maximum flow

Model

$$
\begin{aligned}
\max v & & & \\
x_{ij} \leq u_{ij}, & \qquad \forall (i,j) \in A & & \text{(capacity constraints)} \\
\sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} & = \begin{cases} v, i = s \\ -v, i = t \\ 0, i \neq s, t \end{cases} , \forall i \in N & & \text{(flow balancing)} \\
x_{ij} \geq 0, & \qquad \forall (i,j) \in A & & \\
v \geq 0 & & &
\end{aligned}
$$

Problem description

A graph $G = (N, A)$ is given (where $n = |N|$). An origin node $s \in N$ is given. A cost c_{ij} is given for each arc $(i, j) \in A$. We have to find the minimum cost paths from node s to any other node of the graph.

Decision variables

$x_{ij} \geq 0$, $\forall (i, j) \in A$: amount of flow on arc (i, j) representing the number of paths using arc (i, j).

Shortest Paths Tree (SPT) - formulation

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

$$\sum_{k|(i,k)\in A} x_{ik} - \sum_{j|(j,i)\in A} x_{ji} = \begin{cases} n-1, & i = s \\ -1, & i \neq s \end{cases}, \quad \forall i \in N$$

$$x_{ij} \geq 0, \quad \forall (i,j) \in A$$

Shortest path tree

Solve the following shortest path tree problem and display the paths: