MBT Experiments

January 11, 2021

This text contains a description of how to perform experiments with MBT ILP models on Linux.

1 Structure of the directory `experiments/`

- `data/` - instance files to be solved. The directory `data/existing/` consists of instances obtained from online source that were used in the original paper.
- `logs/` - log files for elapsed time (`timelog.txt`) and objective function value (`objlog.txt`)
- `manual/` - this manual
- `models/` - ILP models coded in AMPL
- `old/` - old version of the experiment scripts (no need to bother about them)

2 Running the experiment

The main script for running the experiments is `mbtrun.sh`. The script is executed by typing

```
./mbtrun.sh <options> [path to directory with input files]
```

or

```
./mbtrun.sh <options> [path to one input file]
```

from the directory `experiments/` (where `mbtrun.sh` is located). There are three optional switches:

`-i` specify whether the ILP models should be solved

`-m` specify whether the matching upper bounding algorithm should be solved

`-t number_of_seconds` specify a limit on the solution time for each method (default 3600)

If both switches `-i` and `-m` are excluded, both models and matching algorithm are solved. The reason for these optional switches is to be able to run either only the matching algorithm, or only the ILP models. So for example

```
./mbtrun -mt 600 data/
```

runs only the matching algorithm with time limit set to 10 minutes per each instance in the directory `data/`. In order to run both ILP model and matching with the default 1 hour time limit, simply type

`./mbtrun path_to_directory_with_input_files`.

The last parameter specifying either a directory containing instance files, or a single instance file is required.

3 Input data

The input data filenames are not expected to follow any specific naming convention. The data files are unformatted in the sense that they contains nothing except the values to be read, and are aimed for the AMPL `read` command. The instances is encoded as

```
cardV cardE cardS
v1 v2
v1 v3
v2 v4
```

```
...
s1 s2 s3...
lb ub
```

where `cardV` is the number of nodes, `cardS` is the number of sources and `cardE` is the number of edges (potential communication links). The first line is followed by a list of edges, and the second last line consists of a list of source nodes. Finally, `lb` and `ub` denote lower and upper bound on the optimal solution, respectively. There must not be any other characters. An example of an input file representing a path on 9 nodes with one source "0" look as follows:

```
9 8 1
0 1
1 2
2 3
3 4
4 5
5 6
6 7
7 8
0
4 12
```

4 How it works

The script iterates over all filenames in the input directory and reads the data from them. We study two models, let's call them *maxInformed* and *minTime*. Both models are written in `models/MBT-combined.mod`. It contains two objective functions, and commands `drop [constraint_name]` and `restore [constraint_name]` specify which constraints are used for which model. Once the data are loaded, the script runs LP relaxation of the model *maxInformed*, then the same model without the relaxation, and then the model *minTime*. Finally, the 'matching' algorithm for upper bound is executed 4 times - always for a specified number of steps it looks ahead (4, 3, 2,

and 1).

When running the model *maxInformed*, the script iterates over increasing deadline. The iterative process terminates once either the objective value is `cardV-cardS`, or elapsed time exceeds a given time limit, or the number of iteration reaches an upper bound.

The file `mbtrun.sh` is very ad-hoc and contains several hard coded parameters (location of the cplex solver, log files, etc.), which should be adjusted according to the machine on which it is executed.

5 Logging

Elapsed time in seconds and objective function value of individual methods are stored in files `logs/timelog.txt` and `logs/objlog.txt`. Each line starts with a filename of the processed instance followed by the respective values in the order mentioned above.

The reported elapsed time for model *maxInformed* and is a sum of elapsed times of all calls of the `solve` command during the iterative procedure.