XCSP3 Competition - Solvers description

Charles Thomas (charles.thomas@uclouvain.be)

June 10, 2017

This document shortly describes the solvers submitted for the First XCSP3 competition [1] and how to launch them using the command line interface. This competition has for aim to assess the performances of solvers for CSP (Constraint Satisfaction Problem) and COP (Constrained Optimization Problem) instances in the XCSP3 format [2].

1 Solvers submitted

The solvers submitted have been developed by members of the beCool team [3] which is a research group of the ICTEAM institute at the Université catholique de Louvain in Belgium active in Constraint Programming and Local Search. The solvers use OscaR [4], a scala toolkit for Operations Research problems including a Constraint Programming engine.

The source code of the solvers is available at `https://bitbucket.org/oscarlib/oscar/wiki/Home`

1.1 Conflict ordering solver

The Conflict ordering solver performs a complete search with a conflict ordering heuristic [5].

1.2 Adaptive large neighbourhood search solver

The Adaptive large neighbourhood search solver uses an adaptive large neighbourhood search (ALNS) approach based on [6] and [7]. It consists in performing a large neighbourhood search using different heuristics in order to adapt the solver during the search by using more often the efficient heuristics. The implementation used in this solver is currently under development in OscaR. As it uses an incomplete approach, it is unable to prove the optimality of the best solution found.

1.3 Hybrid solver

The Hybrid solver combines the approaches of the Conflict ordering and the ALNS solvers. It starts the search with a complete search using the conflict ordering heuristic in order to find and prove the optimum for small instances and to get a good starting solution for larger instances. Then it switches to an ALNS search to improve the current solution. Finally the solver restarts a complete search with the most successful heuristic in the ALNS in order to try to prove the optimality of the best solution found by the ALNS.

1.4 Parallel solver

The parallel solver uses an Embarrassingly Parallel Search approach with a conflict ordering heuristic.

2 Command line usage

All the solvers use the same command line interface. The solvers have been uploaded as executable jar files. To launch a solver with an instance file, use the following command:

```
java -XmxMEMLIMITm -jar SOLVER --randomseed RANDOMSEED --timelimit TIMELIMIT
--memlimit MEMLIMIT --nbcore NBCORE (--tmpdir TMPDIR --dir DIR) BENCHNAME
```

where:

- SOLVER should be replaced by the path to the solver jar file.
- RANDOMSEED should be replaced by the random seed value.
- TIMELIMIT should be replaced by the allocated time in seconds.
- MEMLIMIT should be replaced by the allocated memory in MB.
- NBCORE should be replaced by the number of cores allocated.
- TMPDIR which is optional should be replaced by the temporary directory path.
- DIR which is optional should be replaced by the solver file directory.
- BENCHNAME which is mandatory and must always be the last argument should be replaced by the path to the instance file.

The order of the named arguments does not matter as long as their value directly follows the corresponding flag starting with a double dash and they are placed between the solver file path and the instance path. It is necessary to use the java Xmx command to set the maximal memory limit if the allocated memory is superior to the default value. Note that the values of TMPDIR and DIR will not be used but the arguments are still supported.

References

[1] First international XCSP3 competition, `http://www.xcsp.org/competition`

[2] F. Boussemart, C. Lecoutre, and C. Piette, XCSP3: An integrated format for benchmarking combinatorial constrained problems, Tech. Report arXiv:1611.03398, CoRR, 2016, Available from `https://arxiv.org/abs/1611.03398` and `http://www.xcsp.org/format3.pdf`.

[3] beCool: Belgium Constraints Group, `http://becool.info.ucl.ac.be/`

[4] OscaR: Scala in OR, `https://bitbucket.org/oscarlib/oscar/wiki/Home`

[5] Steven Gay, Renaud Hartert, Christophe Lecoutre, Pierre Schaus, *Conflict Ordering Search for Scheduling Problems*, 2015.

[6] Philippe Laborie, Daniel Godard, *Self-Adapting Large Neighborhood Search: Application to single-mode scheduling problems*, 2007.

[7] Stefan Ropke, David Pisinger, *An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows*, 2006.