macht and minimacht

Djamal Habet and Cyril Terrioux

Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
{djamal.habet, cyril.terrioux}@lis-lab.fr

1 Solver description

macht and minimacht are written in C++ and both implement the algorithm MAC+RST+NG [1].
For the competition, we have made the following choices:

- an heuristic based on the exponential recency weighted average [2,3] is exploited for ordering the variables,
- *lexico* is used as value ordering heuristic,
- generalized arc-consistency is enforced by a propagation-based system exploiting events,
- restarts are performed according to a geometric restart policy based on the number of backtrack with an initial cutoff set to 100 and an increasing factor set to 1.1,

Note that, at now, macht only takes into account the following constraints:

- `intension`,
- `extension`,
- `allDifferent` (the element `<except>` is not supported),
- `allEqual`,
- `ordered`,
- `sum`,
- `maximum` (the variant `<arg_max>` is not supported),
- `minimum` (the variant `<arg_min>` is not supported),
- `element`,
- `channel`,
- `noOverlap` (only the one dimensional form),
- `instantiation`.

2 Command line

macht and minimacht can be launched thanks to the following command line:

```
SOLVER TIMELIMIT BENCHNAME
```

where:

- `SOLVER` is the path to the executable macht or minimacht,
- `TIMELIMIT` is the number of seconds allowed for solving the instance,
- `BENCHNAME` is the name of the XML file representing the instance we want to solve.

Acknowledgments

References

1. Lecoutre, C., Saïs, L., Tabary, S., Vidal, V.: Recording and Minimizing Nogoods from Restarts. JSAT 1(3-4), 147–167 (2007)
2. Liang, J.H., Ganesh, V., Poupart, P., Czarnecki, K.: Exponential recency weighted average branching heuristic for sat solvers. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. pp. 3434–3440. AAAI Press (2016)
3. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, USA, 1st edn. (1998)