A Report on the BPrologCSP Solver

Neng-Fa Zhou
CUNY Brooklyn College

This note attempts to give a quick analysis of the results of the BPrologCSP solver in the second international solver competition. The constraint propagators used in the solver are implemented in AR (action rules), a language available in B-Prolog, and the search part is implemented using `labeling_mix`, a built-in in B-Prolog, that allows for the use of mixed strategies and time limits in labeling variables. The reader is referred to the papers on AR [2, 4] for the implementation of the propagators and the B-Prolog users' manual for the use of `labeling_mix`.

The results of the BPrologCSP solver are mixed: One the one hand, it was unexpectedly ranked top in two of the categories (global and n-ary intentional), and on the other hand, it was placed only 13th in the binary intentional category. The propagators from last year's solver [3] were used for extensionally defined constraints. Since the procedures on tables were implemented in Prolog, the poor performance on extensionally defined constraints was expected.

B-Prolog's finite-domain solver has the reputation for high performance. As described in [2], the high performance is partially attributed to the efficient event-handling architecture. This high performance is normally revealed on not only n-ary constraints but also binary constraints. The implementation of the `all_different` constraint is based on a weak version of the hall-set finding algorithm [4], which is weaker in terms of prunning power than Regin's filtering algorithm [1]. It is unclear if Regin's algorithm is used in any other participating solvers. If so, it would be worthwhile to investigate why the BPrologCSP solver outperformed them.

The BPrologCSP solver was disappointedly ranked only 13th among 16 participating solvers in the binary intentional category. The following table shows the instances that the BPrologCSP solver failed to solve within the time limit:

Problem class	# failed instances
fapp	245
taillard	148
haystack	48
rlfap	30
queensKnight	18
knights	15
pigeons	12
os-qp	10

A closer look reveals the reason: Almost all of the failed instances contain non-linear (e.g., $X * Y = C$, $abs(X - Y) = C$, and $X \; mod \; Y = C$) and disjunctive constraints which were not efficiently implemented in the submitted version of the solver.

Future improvements include: (1) refining the propagators for non-linear and disjunctive constraints; (2) introducing certain constraint reasoning ability to the solver; (3) and tuning the labeling strategies.

Acknowledgement

The competition would be impossible without the huge amount of time and energy put into it by the organizers. So thanks go to the organizers. Special thanks go to Olivier Roussel for obtaining all the results and feedbacks.

References

[1] J.C. Regin. A filtering algorithm for constraints of difference in CSPs. In *Proceedings of the National Conference on Artificial Intelligence(AAAI-94)*, pages 362–367. AAAI Press, 1994.

[2] Neng-Fa Zhou. Programming finite-domain constraint propagators in action rules. *Theory and Practice of Logic Programming (TPLP)*, 6(5):483–508, 2006.

[3] Neng-Fa Zhou and Mark Wallace. A simple constraint solver in action rules for the cp'05 solver competition. In *Proceedings of the CP workshop on Constraint Propagation and Implementation*, page 6 pages, 2005.

[4] Neng-Fa Zhou, Mark Wallace, and Peter J. Stuckey. The `dom` event and its use in implementing constraint propagators. Technical report TR-2006013, CUNY Compute Science, 2006.