AQuA: Solver Description

Paolo Marin

Albert-Ludwigs-Universität Freiburg, Germany
{marin}@informatik.uni-freiburg.de

AQuA is a search-based quantified Boolean formulas (QBF) solver. Input formulas are those in prenex conjunctive normal form (PCNF) stored in `QDimacs 1.1` format [7]. AQuA was implemented from scratch, and uses natively 64 bit data structures. In AQuA, literal propagation is performed through unit, pure, and don't literal detection using lazy data structures [1]. In particular, Boolean constraint propagation (BCP) makes use of either two or three literals watching strategies, which can be chosen on the command line, and pure/don't care make use of a constant time access data structure. For backtracking, a conflict and solution driven constraint learning (CSDCL) [2,6] approach is used. When backtracking from a solution, which is obtained by setting a complete assignment to the variables, an initial pseudo-minimal prime implicant is built. The conflict/solution analysis terminates and computes a new constraint to learn when in the implication graph a cut-point according to a specific criterion is found. In AQuA, two criteria are implemented, namely the common first UIP [11] (`F-UIP`) and a so-called first semantic UIP (`S-UIP`). None of the two implementations incur into the worst case exponential learning analyzed in [9].

Additionally, two decision heuristics (`OCCS` and a `VSIDS`-like one) [10] are implemented. Finally, a restart strategy [5] and phase saving [8] are also implemented.

The solver is loosely coupled with the QBF preprocessor sQueezeBF [3], which is given a timeout of 100 seconds and is run as separate thread using a `popen()` POSIX call: during the preprocessing phase, AQuA does nothing, hence there is no parallel work.

Three variants of AQuA have been submitted, which are listed below.

Submitted Files

The submitted file (`AQuA.zip`) contains the following files:

- `qube` this is QuBE7.2 [4] binary, which was officially released and available for download. In it the preprocessor sQueezeBF is tightly coupled, and is called internally by AQuA with a command line of the form `''ulimit -t 100; ./qube <qbf-file-name> -all''`, and its output is read through a pipe. This file must lay in the same directory AQuA does;
- `AQuA-F3V` with `F-UIP` learning, 3 literals watching, and `VSIDS` decision heuristic;
- `AQuA-S2V` with `S-UIP` learning, 2 literals watching, and `VSIDS` decision heuristic;
- `AQuA-S3O` with `S-UIP` learning, 3 literals watching, and `OCCS` decision heuristic.

References

1. Gent, I., Giunchiglia, E., Narizzano, M., Rowley, A., Tacchella, A.: Watched data structures for QBF solvers. In: Giunchiglia, E., Tacchella, A. (eds.) Theory and Applications of Satisfiability Testing, 6th International Conference, (SAT). LNCS, vol. 2919, pp. 25–36. Springer (2004)
2. Giunchiglia, E., Narizzano, M., Tacchella, A.: Clause/Term Resolution and Learning in the Evaluation of Quantified Boolean Formulas. Journal of Artificial Intelligence Research. 26, 371–416 (2006)
3. Giunchiglia, E., Marin, P., Narizzano, M.: sQueezeBF: An Effective Preprocessor for QBFs Based on Equivalence Reasoning. In: Strichman, O., Szeider, S. (eds.) Proceedings of the International Conference on Theory and Applications of Satisfiability Testing. Lecture Notes in Computer Science, vol. 6175. Springer (2010)
4. Giunchiglia, Enrico and Marin Paolo and Narizzano, Massimo: QuBE7.0, System Description. Journal of Satisfiability. 7(8), 83–88 (2010)
5. Gomes, C.P., Selman, B., Kautz, H.: Boosting combinatorial search through randomization. In: Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence. pp. 431–437. AAAI '98/IAAI '98, American Association for Artificial Intelligence, Menlo Park, CA, USA (1998), `http://dl.acm.org/citation.cfm?id=295240.295710`
6. Marin, P., Giunchiglia, E., Narizzano, M.: Conflict and Solution Driven Constraint Learning in QBF. In: Doctoral Program of Constraint Programming Conference (2010)
7. Narizzano, M., Tacchella, A.: QDIMACS prenex CNF standard ver. 1.1 (2005), available on-line from `http://www.qbflib.org/qdimacs.html`
8. Pipatsrisawat, K., Darwiche, A.: A lightweight component caching scheme for satisfiability solvers. In: Marques-Silva, J., Sakallah, K.A. (eds.) Theory and Applications of Satisfiability Testing – SAT 2007: 10th International Conference, Lisbon, Portugal, May 28-31, 2007. Proceedings. pp. 294–299. Springer Berlin Heidelberg, Berlin, Heidelberg (2007), `http://dx.doi.org/10.1007/978-3-540-72788-0_28`
9. Van Gelder, A.: Contributions to the theory of practical quantified boolean formula solving. In: Principles and Practice of Constraint Programming. pp. 647–663. Springer (2012)
10. Zhang, L., Madigan, C.F., Moskewicz, M.W., Malik, S.: Efficient conflict driven learning in a Boolean satisfiability solver. In: International Conference on Computer-Aided Design. pp. 279–285 (Nov 2001), `http://www.ee.princeton.edu/~chaff/iccad2001_final.pdf`
11. Zhang, L., Malik, S.: Conflict Driven Learning in a Quantified Boolean Satisfiability Solver. In: Proceedings of International Conference on Computer Aided Design (2002)