

# SATPLAN04: Planning as Satisfiability

Henry Kautz

Department of Computer Science & Engineering  
University Of Washington  
Seattle, WA 98195 USA

SATPLAN04 is a updated version of the planning as satisfiability approach originally proposed in (Kautz & Selman 1992; 1996) using hand-generated translations, and implemented for PDDL input in the blackbox system (Kautz & Selman 1999). Like blackbox, SATPLAN04 accepts the STRIPS subset of PDDL and finds solutions with minimal parallel length: that is, many (non-interfering) actions may occur in parallel at each time step, and the total number of time steps in guaranteed to be as small as possible.

Also like blackbox, SATPLAN works by:

1. Constructing a graphplan-style (Blum & Furst 1995) style planning graph up to some length  $k$ ;
2. Translating the constraints implied by the graph into a set of clauses, where each specific instance of an action or fact at a point in time is a proposition;
3. Using a general SAT solver to try to find a satisfying truth assignment for the formula;
4. If the result is unsat or time out, increment  $k$  and repeat;
5. Otherwise, translate the solution to the SAT problem to a solution to the original planning problem;
6. Postprocess the solution to remove (some of the) unnecessary actions.

The final step is useful because the SAT translation of the planning graph does not guarantee that every action proposition that is true in the solution is actually needed in order to achieve the goals of the original plan.

SATPLAN04 supports four different encoding styles, “action-based”, “graphplan-based”, “skinny action-based”, and “skinny graphplan-based”, based on the classes of clauses included in the encoding. Classes of clauses are:

1. An action implies its preconditions.
2. A fact implies the disjunction of the actions that have it as an effect (including “no op” actions) at the previous time slice.

3. An action implies each of the disjunctions of the actions at the previous time slice that add each of its preconditions.
4. Actions with conflicting preconditions and effects are mutually exclusive.
5. Actions for which mutual exclusion can be inferred using graphplan’s constraint propagation algorithm are mutually exclusive.

“Graphplan-based” encodings use classes (1) and (2), while “action-based” encodings use class (3). “Skinny” encodings include class (5) while non-skinny encodings include both (5) and (6).

In general the action-based skinny encoding gives the most robust performance, simply because as the smallest in terms of both variables and clauses it is least likely to result in a formula that is too large to fit into main memory. (Satisfiability testing and virtual memory are an unhealthy combination.)

The single most important difference between blackbox and SATPLAN04 is the SAT solvers used. Blackbox included the original graphplan (non-translation based) search engine, the local-search SAT solver walksat (Selman, Kautz, & Cohen 1994), the forward-checking DPLL-based solver satz (Li & Anbulagan 1997), and the clause-learning DPLL-based solvers relsat (Bayardo & Schrag 1997) and zChaff (Moskewicz *et al.* 2001).

By contrast, SATPLAN04 uses a single highly optimized DPLL-based solver called “siege”, that was developed by Lawrence Ryan as part of his research at Simon Fraiser University under the direction of Prof. David Mitchell. Linux binaries of siege can be downloaded from <http://www.cs.sfu.ca/loryan/personal/>. We thank Lawrence Ryan for permission to incorporate siege in SATPLAN04.

Siege, like relsat and zChaff, performs clause-learning (that is, inferring new clauses at backtrack points), and like zChaff uses optimized “watched literal” data structures for managing large clause sets efficiently. Beyond that it appears to incorporate a number of other optimizations that make it particularly well-suited for the planning as satisfiability approach. In our initial informal tests siege signifi-

cantly outperformed all the other solvers mentioned above. Later this summer we will post detailed comparisons of the different SAT solvers on planning formulas on our planning as satisfiability web page, <http://www.cs.washington.edu/homes/kautz/blackbox/>.

The PDDL parser in SATPLAN04 is considerably more robust than the one in blackbox, but it does not yet handle any non-STRIPS features other than types, such as derived effects and conditional actions. We plan to extend SATPLAN04 to handle these and other features in time for the 2005 planning competition.

## References

- Bayardo, R. J. J., and Schrag, R. C. 1997. Using CSP look-back techniques to solve real-world SAT instances. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, 203–208.
- Blum, A., and Furst, M. 1995. Fast planning through planning graph analysis. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 95)*, 1636–1642.
- Kautz, H., and Selman, B. 1992. Planning as satisfiability. In *Proceedings of the 10th European Conference on Artificial Intelligence*, 359–363. Wiley.
- Kautz, H., and Selman, B. 1996. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, 1194–1201. AAAI Press. (Best Paper Award).
- Kautz, H., and Selman, B. 1999. Unifying sat-based and graph-based planning. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, 318–325. Morgan Kaufmann.
- Li, C. M., and Anbulagan. 1997. Heuristics based on unit propagation for satisfiability problems. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI 97)*, 366–371.
- Moskewicz, M.; Madigan, C.; Zhao, Y.; Zhang, L.; and Malik, S. 2001. Chaff: Engineering an efficient sat solver. In *39th Design Automation Conference*.
- Selman, B.; Kautz, H.; and Cohen, B. 1994. Noise strategies for improving local search. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, 337–343. AAAI Press.