Satzilla 0.9

Eugene Nudelman, Kevin Leyton-Brown, Galen Andrew,
Carla Gomes, Jim McFadden, Bart Selman, Yoav Shoham

Our solver takes a novel portfolio approach to the SAT problem. Instead of implementing a SAT algorithm from scratch, Satzilla uses machine learning models to select and run a single algorithm from a set of existing algorithms. This approach is motivated by the idea that different algorithms perform well on different types of problem inputs. For most NP-hard problems, including SAT, there is no single algorithm that outperforms all other algorithms on every instance. This means it is possible for a portfolio that selects from a set of algorithms to outperform all of its constituent algorithms.

Here is how Satzilla uses our portfolio approach to solve SAT instances:

First, it computes about 60 polynomial time features of the instance that are indicative of runtime. These include

- simple features like the number of variables and clauses, and clause to variable ratio
- statistics (mean, stddev, min, max, entropy) of the number of clauses each variable participates in
- statistics and clustering features of constraint graphs that represent the problem, i.e. the clause-graph (a graph whose nodes are clauses with an edges between clauses in which one clause contains a literal and the other clause contains its negation)
- statistics from an LP relaxation of the SAT problem
- probing features that sample data from iterations of local search and DPLL

It then uses linear models of these features (which were learned offline using statistical regression) to predict the run time for each of its constituent algorithms. For Satzilla 0.9, these include: 2clseq, Limmat, JeruSat, OKsolver, Relsat, Sato, Satzrand, and Zchaff. It then runs the algorithms with the lowest predicted run time.

Our Hors Course solver satzilla2 uses two additional complete solvers, Eqsatz and Heerhugo. It also executes the AutoSat algorithm for a little time before starting any other computation, and is thus able to filter out easy satisfiable problems.

Unfortunately, the predictive models that were submitted with the two solvers were trained on a renormalized set of features. Thus, while the algorithm selection performed by Satzilla is not completely random, it is not nearly the same as it would be with correct runtime models.

The techniques for predicting run time of algorithms are described in *Learning the Empirical Hardness of Optimization Problems: The Case of Combinatorial Auctions*, K. Leyton-Brown, E. Nudelman, Y.Shoham. CP-2002. (`http://robotics.stanford.edu/~kevinlb/computation.pdf`)

The portfolio approach is elaborated in our paper *A Portfolio Approach to Algorithm Selection*, K. Leyton-Brown, E. Nudelman, G. Andrew, J. McFadden, Y. Shoham. IJCAI-2003. (`http://robotics.stanford.edu/~kevinlb/boosting-short.pdf`)