Learning Polarity from Structure in SAT

Bryan Silverthorn and Risto Miikkulainen

Department of Computer Science
The University of Texas at Austin
{bsilvert,risto}@cs.utexas.edu

1 Introduction

Few instances of a computational problem are *sui generis*; most instead belong to some distribution of related instances, and information gained from solving past instances from the distribution may be leveraged to solve future instances more efficiently. Algorithm portfolio methods and algorithm synthesis systems are two examples of this idea. This paper proposes and demonstrates a third approach. It shows that, for related instances of satisfiability (SAT), variables' values in satisfying assignments can be correlated with structural features of their appearances in those instances, such as the mean polarities of their literals and the statistics of their constraint graph neighborhoods. Experiments on widely-used benchmark collections show that these features can be used by a standard classifier to generate better initial assignments and substantially improve the average performance of a modern solver.

2 Polarity Prediction as Supervised Learning

SAT variables that are semantically related may also share structurally similar positions. Consider two variables: do both appear exclusively in short clauses? Are both seen mostly in negated literals? Do both appear infrequently in their expressions? If a variable in an unsolved instance appears similar to variables in previously-solved instances, most of which take on a particular value in known satisfying assignments, then that information may inform the search process.

This paper transforms variable initialization into the supervised learning problem of predicting the satisfying value of each variable given a vector of real-valued features, including the mean and standard deviation of the lengths of the clauses in which the variable appears; the mean and standard deviation of the ratios of positive literals in such clauses; the number of such clauses; the number of times that the variable appears in a Horn clause; the number of times that the variable appears as a consequent; the ratio of positive literals involving that variable; and the degree of the corresponding node in the variable graph.

3 Classification for Assignment Initialization

To measure the impact of this learning framework, logistic regression is used to initialize the starting assignment of the `TNM` [4] local search solver for each of eight different collections of benchmarks, drawn from SATLIB [1] and from those used previously in the literature [2, 3]. Each classifier is trained on assignments obtained by solving a fraction of the instances in each benchmark collection. Assignments are then generated either deterministically, by computing the maximum-likelihood assignment according to the classifier, or probabilistically, by sampling according to its class probabilities.

Table 1. The median number of search actions required to solve the instances in each benchmark collection, using deterministic or random classifier-based initialization, given as the fraction of the corresponding result obtained under the solver's default initialization scheme. Lower values are therefore better, and values less than one represent improvements in search speed. These ratios were averaged over 64 random 50% train/test splits, for which both mean (μ) and standard deviation (σ) are reported. Runs were limited to 2,000,000 steps. Both classifier-based initialization schemes reduce search cost, often substantially, for the benchmarks tested.

	Cost Ratio			
	Deterministic		Random	
Collection	μ	σ	μ	σ
`SAPS-newQCP`	0.98	0.14	0.98	0.13
`SAPS-SWGCP`	0.21	0.04	0.97	0.21
`satenstein-cbmc`	0.01	0.00	0.23	0.11
`satlib-bms`	0.97	0.19	0.98	0.18
`satlib-cbs-m403-b10`	0.58	0.08	0.82	0.10
`satlib-cbs-m449-b90`	0.87	0.10	0.96	0.10
`satlib-ii`	0.97	0.32	0.76	0.26
`satlib-rti`	0.76	0.10	0.93	0.12

Table 1 reports the effects of different initialization schemes on the number of required solver steps. These empirical results strongly suggest that classifier-based initialization can reduce the net computational cost of solving large collections of related SAT instances.

References

1. Hoos, H.H., Stützle, T.: SATLIB: An online resource for research on SAT (2000), `http://www.cs.ubc.ca/~hoos/SATLIB/benchm.html`
2. Hutter, F., Hoos, H.H., Leyton-Brown, K., Stützle, T.: ParamILS: An automatic algorithm configuration framework. JAIR (2009)
3. KhudaBukhsh, A., Xu, L., Hoos, H., Leyton-Brown, K.: SATenstein: Automatically building local search sat solvers from components. In: IJCAI (2009)
4. Wei, W., Li, C.M.: Switching between two adaptive noise mechanisms in local search for SAT. SAT 2009 Competitive Events Booklet (2009)