Sparrow2011

Adrian Balint[1], Andreas Fröhlich[1], Dave A. D. Tompkins[2], and Holger H. Hoos[2]

[1] Institute of Theoretical Computer Science
Ulm University, Germany
{adrian.balint, andreas.froehlich}@uni-ulm.de

[2] Department of Computer Science
University of British Columbia, Canada
{davet,hoos}@cs.ubc.ca

SPARROW is a stochastic local search (SLS) solver for SAT formulae in CNF format [2]. It was named after the mascot of the German city of Ulm. SPARROW2011 is a variant of SPARROW that was developed for the 2011 SAT competition. The core SPARROW2011 algorithm is identical to SPARROW; it has been re-implemented in UBCSAT [9] and a mechanism has been added to automatically select optimized parameter settings based on the maximum clause length of the input instance. These parameter settings were obtained from configurations found using the automated parameter configurator PARAMILS [5,4]. In the following, we describe the basic algorithm underlying SPARROW and the specifics of SPARROW2011.

SPARROW is largely based on the GNOVELTY$^+$ algorithm [7] (version 1.2). The core component of GNOVELTY$^+$ that SPARROW borrows is the combination of the clause penalty-based dynamic local search with a promising variable scheme. In GNOVELTY$^+$ (and in SPARROW), clause penalties (weights) are assigned to each clause and are increased similar to the PAWS algorithm [8] and smoothed probabilistically similar to the SAPS algorithm [6] whenever there are no promising variables (as opposed to being in a local minimum as in PAWS and SAPS). Because the clauses are penalized, a penalized variable score *penScore* is used in lieu of *score*, and variables are considered promising based on their penalized score (see original paper [7] and the 2009 competition source code for more details). We use the notation that a positive *score* corresponds to an increase in the number of satisfied clauses if the variable were to be flipped. The behaviour of SPARROW differs from GNOVELTY$^+$ when there are no (penalized) promising variables. SPARROW replaces the WALKSAT-based component in GNOVELTY$^+$ with a stochastic decision procedure based on a probability distribution.

SPARROW uses two properties we shall call *sparrowScore* and *sparrowAge* that depend on the *penScore* and *age* properties, respectively: The *sparrowScore* property uses the *penScore* property in the following way: if a *penScore* is positive but not promising, then the value of *penScore* is one, otherwise an exponential function is used. The full definition of *sparrowScore* is as follows:

$$\mathit{sparrowScore} = \begin{cases} c_1^{\mathit{penScore}} & \text{if } \mathit{penScore} < 0 \\ 1 & \text{otherwise,} \end{cases}$$

and the *sparrowAge* property is defined as:

$$\mathit{sparrowAge} = 1 + \left(\frac{\mathit{age}}{c_3}\right)^{c_2},$$

where c_1, c_2, c_3 are parameters of SPARROW.

SPARROW uses a a distribution-based decision variable selection mechanism that selects each variable with probability proportional to its evaluation of $\langle \mathit{sparrowScore} \cdot \mathit{sparrowAge} \rangle$. In other words, the probability of selecting a variable is equal to:

$$\frac{\mathit{sparrowScore} \cdot \mathit{sparrowAge}}{\sum \mathit{sparrowScore} \cdot \mathit{sparrowAge}},$$

where the sum ranges over all variables in the given clause.

The full parameter space for SPARROW is comprised of the three constants and the smoothing probability (ps) inherited from GNOVELTY$^+$. In [2] the authors proposed the following values $(c_1, c_2, c_3, ps) = (2, 4, 10^5, 0.347)$ as good settings for 3-SAT instances, which was found by manual tuning on a subset of the SAT competition 2009 benchmark instances. No configurations have been previously identified that would perform particularly well on 5- and 7-SAT instances.

While the original software implementation of SPARROW [2] was a modification of the GNOVELTY$^+$ 2009 competition source code, SPARROW2011 is a re-implementation of the original in the UBCSAT framework [9]. SPARROW2011 is based on a (preliminary) beta version of UBCSAT 1.2 (build 1.2b10). The UBCSAT implementation is semantically equivalent to the original SPARROW solver, but is more efficient with respect to CPU Time and provides better reports and statistics for empirical analysis.

To find good configurations for SPARROW's parameters, (c_1, c_2, c_3, ps), we used the automated configurator PARAMILS [5,4]. As a basis for the automated configuration process, we generated 100 random 3-,5- and 7-SAT satisfiable instances with characteristics similar to those used in the 2009 competition:

3-SAT: 10 000 variables, 4.2 clause/variable ratio
5-SAT: 600 variables, 20 clause/variable ratio
7-SAT: 100 variables, 85 clause/variable ratio

For each training set, we performed 24 independent runs of PARAMILS for 4 (CPU) days each. The parameter configurations found by PARAMILS for each instance set were all evaluated on subsets of the instances from the SAT 2009 competition to find the best configuration. For the evaluation we have used the EDACC framework [1]. For 3-SAT, the best configuration found by PARAMILS did not perform as well as the manually tuned configuration mentioned previously, so we used the latter instead. Starting from the best configuration for each set, we further hand-tuned the parameters by increasing the granularity of the parameters values and by attempting to interpolate the configurations from one class to the other. This manual tuning took approx. 10 hours and was performed using EDACC. For the evaluation of the parameter configurations on the different instance classes, EDACC used between 300 and 400 CPUs from the BWGrid [3] computing grid. The parameters settings found by this procedure are:

k-SAT	c_1	c_2	c_3	ps
3-SAT	2.15	4	100 000	0.347
5-SAT	2.85	4	75 000	1.0
7-SAT	6.5	4	100 000	0.83

Table 1. SPARROW2011 parameter settings.

The parameter configuration to be used by SPARROW2011 on a given input instance is selected based on the maximum clause length found in that instance. The 3-SAT configuration is used when the maximum clause length less than 4, the 5-SAT configuration is used when the maximum clause length is less than 6, otherwise the 7-SAT configuration is used.

References

1. Balint, A., Diepold, D., Gall, D., Gerber, S., Kapler, G., Retz, R.: EDACC - an advanced platform for the experiment design, administration and analysis of empirical algorithms. In: LION-2011 (to appear)
2. Balint, A., Fröhlich, A.: Improving stochastic local search for SAT with a new probability distribution. In: SAT-2010. LNCS, vol. 6175, pp. 10–15 (2010)
3. bwGRiD (http://www.bw-grid.de/): Member of the German D-Grid initiative, funded by the Ministry of Education and Research and the Ministry for Science, Research and Arts Baden-Wuerttemberg
4. Hutter, F., Hoos, H.H., Leyton-Brown, K., Stützle, T.: ParamILS: An automatic algorithm configuration framework. Journal of Artificial Intelligence Research 36, 267–306 (2009)
5. Hutter, F., Hoos, H.H., Stützle, T.: Automatic algorithm configuration based on local search. In: AAAI-2007. pp. 1152–1157 (2007)
6. Hutter, F., Tompkins, D.A.D., Hoos, H.H.: Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In: Proceedings of the Eighth International Conference on Principles and Practice of Constraint Programming (CP-02). Lecture Notes in Computer Science, vol. 2470, pp. 233–248 (2002)
7. Pham, D.N., Thornton, J., Gretton, C., Sattar, A.: Combining adaptive and dynamic local search for satisfiability. JSAT 4, 149–172 (2008)
8. Thornton, J., Pham, D.N., Bain, S., Ferreira Jr., V.: Additive versus multiplicative clause weighting for SAT. In: PRICAI-08. pp. 405–416 (2008)
9. Tompkins, D.A.D., Hoos, H.H.: UBCSAT: An implementation and experimentation environment for SLS algorithms for SAT and MAX-SAT. In: SAT-2004 Revised. LNCS, vol. 3542, pp. 306–320 (2005)