Advances in Local Search for Satisfiability

Duc Nghia Pham, John Thornton, Charles Gretton, and Abdul Sattar

SAFE Program, Queensland Research Lab, NICTA and
Institute for Integrated and Intelligent Systems, Griffith University, QLD, Australia
{duc-nghia.pham,john.thornton,charles.gretton,abdul.sattar}@nicta.com.au

Abstract. In this paper we describe a stochastic local search (SLS) procedure for finding satisfying models of satisfiable propositional formulae. This new algorithm, gNovelty$^+$, draws on the features of two other WalkSAT family algorithms: R+AdaptNovelty$^+$ and G^{2}WSAT, while also successfully employing a dynamic local search (DLS) clause weighting heuristic to further improve performance.

gNovelty$^+$ was a Gold Medal winner in the random category of the 2007 SAT competition. In this paper we present a detailed description of the algorithm and extend the SAT competition results via an empirical study of the effects of problem structure and parameter tuning on the performance of gNovelty$^+$. The study also compares gNovelty$^+$ with two of the most representative WalkSAT-based solvers: G^{2}WSAT, AdaptNovelty$^+$, and two of the most representative DLS solvers: RSAPS and PAWS. Our new results augment the SAT competition results and show that gNovelty$^+$ is also highly competitive in the domain of solving *structured* satisfiability problems in comparison with other SLS techniques.

1 Introduction

The satisfiability (SAT) problem is one of the best known and well-studied problems in computer science, with many practical applications in domains such as theorem proving, hardware verification and planning. The techniques used to solve SAT problems can be divided into two main areas: complete search techniques based on the well-known Davis-Putnam-Logemann-Loveland (DPLL) algorithm [1] and stochastic local search (SLS) techniques evolving out of Selman and Kautz's 1992 GSAT algorithm [2]. As for SLS techniques, there have been two successful but distinct avenues of development: the WalkSAT family of algorithms [3] and the various dynamic local search (DLS) clause weighting approaches (e.g. [4]).

Since the early 1990s, the state-of-the-art in SAT solving has moved forward from only being able to solve problems containing hundreds of variables to the routine solution of problems with millions of variables. One of the key reasons for this success has been the keen competition between researchers and the public availability of the source code of the best techniques. Nowadays the SAT community organises regular competitions on large sets of benchmark problems and awards prizes to the best performing algorithms in different problem cat-

egories. In this paper we introduce the current 2007 SAT competition[1] Gold Medal winner in the satisfiable random problem category: gNovelty$^+$.

gNovelty$^+$ draws on the strengths of two WalkSAT variants which respectively came first and second in the random category of the 2005 SAT competition: R+AdaptNovelty$^+$ [5] and G^{2}WSAT [6]. In addition, gNovelty$^+$ connects the two branches of SLS (WalkSAT and DLS) by successfully employing a clause weighting heuristic to gain more efficiency.

In the remainder of the paper we describe in more detail the G^{2}WSAT and R+AdaptNovelty$^+$ techniques upon which gNovelty$^+$ is based. We then provide a full explanation of the execution of gNovelty$^+$ followed by a previously unpublished empirical evaluation of the algorithm. This evaluation examines the performance of gNovelty$^+$ on a range of structured problems and reports the effects of parameter tuning in comparison with two of the most representative WalkSAT based solvers: G^{2}WSAT and AdaptNovelty$^+$, and two of the most representative clause weighting solvers: RSAPS and PAWS. Finally we discuss these results and present our conclusions.

2 Existing Techniques

2.1 G^{2}WSAT

During the mid-1990s, Novelty [3] was considered to be one of the most competitive techniques in the WalkSAT family and was able to solve many hard problems faster than the best complete search techniques of that time. However, one key problem with Novelty is its deterministic variable selection,[2] which can cause it to loop indefinitely and fail to return a solution even where one existed [6, 7]. The first practical solution to this problem was to add a random walk behaviour with a probability *wp* to the Novelty procedure [7]. More recently Li and Huang [6] revisited this problem and proposed a more diversified heuristic to weaken the determinism in Novelty. This new Novelty^{++} solver selects the least recently flipped variable for the next move with a *diversification probability dp*, otherwise it performs as Novelty. Li and Huang [6] further improved Novelty^{++} by integrating it with a new gradient-based greedy heuristic based on the count of current false clauses. The resulting G^{2}WSAT solver (depicted in the left hand side of Figure 1) always selects the most promising variable that, if flipped, will reduce the number of false clauses the most. If there is more than one variable with the best score, G^{2}WSAT selects the least recently flipped one, and if the search hits a local minimum, G^{2}WSAT performs as Novelty^{++} until it escapes.

2.2 R+AdaptNovelty$^+$

The performance of every WalkSAT variant critically depends on the setting of a noise parameter p which controls the level of greediness of the search [3, 8].

[1] http://www.satcompetition.org

[2] Novelty deterministically selects the next move from the two best variables of a randomly selected false clause [3].

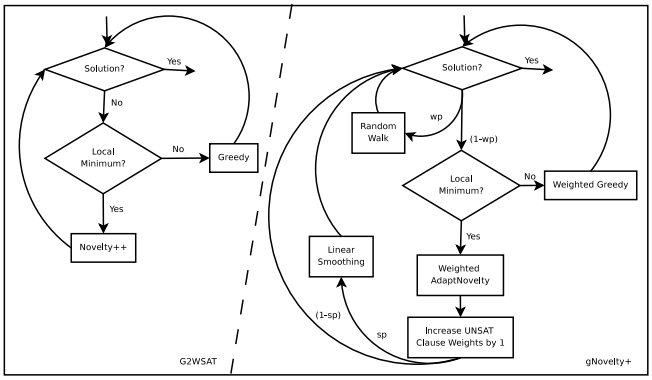

Fig. 1. Flow-chart comparison between G^2WSAT and gNovelty$^+$

This means that without extensive empirical tuning, the average case performance of a WalkSAT algorithm is quite poor. Hoos [8] addressed this problem by proposing an adaptive version of WalkSAT that dynamically adjusts the noise value based on the automatic detection of search stagnation. For example, AdaptNovelty$^+$, the adaptive version of Novelty$^+$, starts with $p = 0$ (i.e. the solver is completely greedy in searching for the next move). If the search enters a stagnation stage (i.e. it encounters a local minimum), the noise value is gradually increased to allow more non-greedy moves to be performed, and hence allow the search to eventually overcome its stagnation. As soon as the search escapes the local minimum, the noise value is reduced to make the search more greedy. Hoos [8] demonstrated experimentally that this adaptive noise mechanism is effective both with Novelty$^+$ and other WalkSAT variants.

In 2005, Anbulagan *et al.* [5] introduced R+AdaptNovelty$^+$, a two-phase SLS solver which improved the performance of AdaptNovelty$^+$ by utilising resolution in the initial phase to derive extra information from the input. In particular, the new solver applies a restricted resolution procedure to all clauses of length ≤ 3 from the input. This process adds resolvent clauses of length ≤ 3 to the problem, and also removes duplicate clauses, tautologies, and literals that appear twice in a single clause. It then runs AdaptNovelty$^+$ on the resulting problem.

3 Our Approach: gNovelty$^+$

The initial development of gNovelty$^+$ focussed on preparing for the 2007 SAT competition. This meant concentrating on the random problem category, where SLS solvers have traditionally outperformed complete solvers. Consequently we paid considerable attention to the previously best performing techniques from this category: R+AdaptNovelty$^+$, G^2WSAT and AdaptNovelty$^+$.

Firstly, we noted that although G^2WSAT came second in the random SAT category in the 2005 competition, it was the best solver for random 3-SAT instances. As neither R+AdaptNovelty$^+$ nor AdaptNovelty$^+$ were competitive

Algorithm 1 gNovelty$^+$(F)

```
for try = 1 to maxTries do
  initialise the weight of each clause to 1;
  randomly generate an assignment A;
  for step = 1 to maxSteps do
    if A satisfies F then
      A as the solution;
    else
      if within a walking probability wp then
        randomly select a variable x that appears in a false clause;
      else if there exist promising variables then
        greedily select a promising variable x, breaking ties by selecting the least recently
        flipped promising variable;
      else
        select a variable x according to the weighted AdaptNovelty heuristic;
        update the weights of false clauses;
        with probability sp smooth the weights of all clauses;
      end if
      update A with the flipped value of x;
    end if
  end for
end for'no solution found';
```

with G^{2}WSAT on those instances, we conjectured that the superior performance of G^{2}WSAT was due to its greedy behaviour. This supposition is consistent with the results reported in [6] where G^{2}WSAT was compared with Novelty$^+$.

On the other hand, R+AdaptNovelty$^+$ outperformed G^{2}WSAT on the 5-SAT and 7-SAT instances in the 2005 competition. This could not be explained by the resolution preprocessor employed by R+AdaptNovelty$^+$ because, for such instances, it simply reorders the occurrence of literals in a clause. We also empirically found that G^{2}WSAT relies heavily on its Novelty^{++} component when solving these instances. Finally we observed that AdaptNovelty$^+$ is generally a more effective Novelty variant for use by G^{2}WSAT than Novelty^{++}.

On the basis of the preceding observations, we constructed gNovelty$^+$ by replacing the Novelty^{++} heuristic in G^{2}WSAT with the AdaptNovelty heuristic to enhance performance on the 5-SAT and 7-SAT instances. We then incorporated a clause weighting heuristic to further improve overall performance. Previously, clause weighting DLS techniques have not performed competitively in the SAT competitions because of their reliance on optimally tuned parameters (the rules of the competition do not allow the hand-tuning of parameters to particular problems). However, DLS generally performs better than WalkSAT when parameter tuning is allowed [9, 10], indicating that clause weights can provide useful guidance. The problem for DLS approaches has been the lack of an effective adaptive mechanism such as the one developed for WalkSAT.[3] Our aim with gNovelty$^+$ was to cross this boundary and implement clause weight guidance within an adaptive WalkSAT framework. The resulting gNovelty$^+$ solver is sketched out in Algorithm 1 and depicted diagrammatically in Figure 1.

[3] Note, RSAPS [9] is a DLS technique that uses a WalkSAT-type adapting mechanism, but RSAPS only adapts one of three possible parameters (see [10]) and remains uncompetitive with the best WalkSAT techniques in the SAT competition.

At every search step, gNovelty$^+$ selects the most promising variable that is also the least recently flipped, based on a *weighted* objective function. The objective is to minimise the sum of weights of all false clauses and a promising variable is any variable that, if flipped, will reduce the overall weighted cost of the solution. If no such promising variable exists, the next variable is selected using a heuristic based on AdaptNovelty that again uses the *weighted* objective function. After an AdaptNovelty step, gNovelty$^+$ increases the weights of all current false clauses by one.[4]

In order to flexibly control the greediness of the search, we also incorporated a new linear version of the probabilistic weight smoothing from SAPS [9]. According to a *smoothing probability sp*, each time gNovelty$^+$ updates the clause weights, this heuristic will reduce the weight of all *weighted* clauses by one (a clause is weighted if its weight is greater than one). Finally, we added a probabilistic-walk heuristic (i.e. the *plus* heuristic from Novelty$^+$ [7]) to further improve the balance between diversification and greediness of the search.

3.1 The Question of Parameters

For SAT 2007, gNovelty$^+$ was specifically tuned for random k-SAT problems with $sp = 0.4$ for $k = 3$ and 1.0 otherwise. It should be noted, however, that an sp value of 1.0 *turns off* the weighting component of gNovelty$^+$ as each weight increase is immediately followed by a weight decrease.

This raises the question as to whether the superior performance of gNovelty$^+$ in the SAT competition was largely dependent on a fortuitous selection of fixed sp values (i.e. that exploited the narrow domain of potential competition problems), or whether the algorithm has a wider field of useful application. To answer this question, we devised an experimental study to test gNovelty$^+$ in comparison with four other state-of-the-art SLS SAT solvers and across a range of benchmark structured problems.

4 Experimental Study

As the performance of gNovelty$^+$ in the SAT random category is already a matter of public record,[5] we based our experimental study on a range of structured benchmark problems that have been used in previous SLS comparison studies.[6] Our problem test set comprises of two circuit synthesis formula problems (3bitadd_31 and 3bitadd_32), two all-interval series problems (ais10 and ais12), two blocksworld planning problems (bw_large.c and bw_large.d), two "flat" graph colouring problems (flat200-med and flat200-har), four large DIMACS graph

[4] We decided to use the additive weight increase at each local minimum as it is cheaper to maintain than its multiplicative weighting counterpart [10].

[5] See http://www.cril.univ-artois.fr/SAT07/slides-contest07.pdf

[6] See http://www.satlib.org

colouring problems (g125.17 to g250.29), two logistics planning problems (logistics.c and logistics.d), five 16-bit parity function learning problems (par16-1-c to par16-5-c), and five hard quasi-group problems (qg1-08 to qg7-13).

As gNovelty$^+$ crosses the boundary both between WalkSAT and clause weighting algorithms and between adaptive and manually-based parameter tuning, for comparison purposes we selected algorithms from each of the four possible categories, i.e. manual WalkSAT (G^{2}WSAT [6]), adaptive WalkSAT (AdaptNovelty$^+$ [8]), manual clause weighting (PAWS [10]) and adaptive clause weighting (RSAPS [9]). Using these algorithms, we manually tuned PAWS, G^{2}WSAT and gNovelty$^+$ to obtain optimal performance for each category of the problem set. These settings are shown in Table 1 (note, only one parameter setting per algorithm was allowed for each of the eight problem categories). Here we not only manipulated the gNovelty$^+$ *sp* parameter but on some categories we also manually tuned the noise parameter of its Novelty component. For G^{2}WSAT we used the optimal settings for the noise and *dp* parameters published in [6, 11], and for PAWS we tuned the w_{inc} parameter (w_{inc} is analogous to gNovelty$^+$'s *sp* parameter except that gNovelty$^+$ reduces weight with probability *sp* whereas PAWS reduces weight after every w_{inc} number of increases [10]).

Table 1. Optimal parameter settings for each problem category

Method	Parameter	Problem Category							
		3bitadd	ais	bw_large	flat200	g	logistics	par16	qg
gNovelty$^+$	noise	adapted	adapted	0.08	adapted	0.10	adapted	0.05	0.02
	sp	0.00	0.00	1.00	0.00	1.00	0.00	0.10	0.00
G^{2}WSAT	noise	0.50	0.20	0.20	0.50	0.30	0.20	0.50	0.40
	dp	0.05	0.05	0.00	0.06	0.01	0.05	0.01	0.03
PAWS	w_{inc}	9	52	4	74	4	100	40	10

4.1 Results

Table 2 shows the results obtained after manually tuning gNovelty$^+$, G^{2}WSAT and PAWS in comparison to the default adaptive behaviour of RSAPS and AdaptNovelty$^+$. Here the results for the best performing algorithm on each problem are shown in bold, with all results reporting the mean and median of 100 runs of each algorithm on each instance. It is worth noting that G^{2}WSAT uses unit propagation to preprocess input problems. We expect that the performance of G^{2}WSAT without unit propagation would be degraded as our benchmark instances, especially the quasi-group ones, contain many unit clauses. All experiments were performed on cluster of 16 computers, each with a single AMD Opteron 252 2.6GHz processor with 2GB of RAM, and each run was timed out at 600 seconds.

A brief overview shows that gNovelty$^+$ has the best results for all 3bitadd, ais, bw_large and logistics problems, is about equal with G^{2}WSAT on the flat graph colouring problems and has some success on the hardest qg problems

Table 2. Optimally tuned results shown in the form: $\frac{median}{mean}$

Instances	gNovelty$^+$		G^{2}WSAT		AdaptNovelty$^+$		RSAPS		PAWS	
	#flips	#secs	#flips	#secs	#flips	#secs	#flips	#secs	#flips	#secs
3bitadd_31	**17,948** **19,746**	**0.06** **0.07**	0% success		151,079 162,828	0.44 0.47	0% success		0% success	
3bitadd_32	**15,116** **15,510**	**0.06** **0.06**	0% success		131,019 134,891	0.45 0.47	0% success		0% success	
ais10	**8,116** **11,596**	**0.01** **0.01**	28,450 58,101	0.02 0.05	1,290,782 1,936,648	1.14 1.67	13,122 18,293	0.02 0.02	13,184 19,823	0.01 0.02
ais12	**48,480** **63,938**	**0.06** **0.08**	329,284 446,670	0.33 0.44	24,315,102 35,291,601	28.14 40.83	103,156 151,088	0.14 0.20	104,414 192,256	0.15 0.26
bw_large.c	**799,665** **1,277,197**	**1.44** **2.22**	1,807,797 2,658,924	1.84 2.68	6,350,267 9,354,872	6.55 9.70	3,606,150 5,360,847	10.72 15.92	990,959 1,603,890	1.46 2.27
bw_large.d	**937,056** **1,131,405**	**2.65** **3.10**	2,308,922 3,121,103	3.60 4.82	21,462,024 30,615,678	37.05 55.24	70% success		1,029,200 1,370,793	2.77 3.51
flat200-med	163,566 241,467	0.07 0.10	**140,542** **185,529**	**0.05** **0.07**	260,162 391,731	0.09 0.13	287,330 376,523	0.15 0.20	248,096 348,112	0.14 0.18
flat200-har	2,576,103 4,037,105	**1.04** **1.64**	3,713,653 14,243,118	1.36 5.01	17,879,059 22,572,057	6.05 7.58	3,143,106 4,199,967	1.59 2.15	**2,402,889** **3,343,704**	1.28 1.76
g125.17	687,290 1,066,447	3.06 4.79	**592,518** **634,772**	**2.00** **2.12**	981,362 1,264,481	4.11 5.41	2% success		492,028 693,579	2.31 3.42
g125.18	12,654 15,255	0.09 0.10	8,538 10,152	0.11 0.12	35,346 35,829	0.10 0.10	1,057,010 1,786,612	5.00 8.38	**10,683** **12,624**	**0.08** **0.08**
g250.15	2,585 2,668	0.11 0.11	2,387 2,417	0.41 0.43	3,310 4,033	0.11 0.12	**2,208** **2,219**	**0.08** **0.08**	2,239 2,247	0.09 0.09
g250.29	637,612 704,455	12.72 14.96	295,924 347,680	**5.99** **6.43**	755,477 894,775	9.96 11.98	0% success		**263,322** **319,511**	8.07 9.24
logistics.c	**6,332** **6,873**	**0.01** **0.01**	53,876 64,369	0.04 0.04	122,186 151,853	0.07 0.09	6,812 7,814	0.01 0.01	10,152 11,642	0.01 0.01
logistics.d	**26,880** **32,211**	**0.04** **0.04**	88,108 106,585	0.09 0.10	170,170 196,295	0.10 0.11	22,558 32,636	0.04 0.05	29,954 42,457	0.05 0.06
par16-1-c	6,943,449 9,621,258	2.92 4.06	98% success		17,185,866 32,062,102	5.87 10.98	73% success		**1,556,886** **2,470,449**	**0.86** **1.37**
par16-2-c	28,290,998 38,825,548	11.97 16.49	90% success		159,404,628 260,240,406	53.41 87.52	39% success		**3,674,518** **4,804,860**	**2.02** **2.67**
par16-3-c	18,135,772 27,626,027	7.75 11.79	70,947,285 105,838,330	27.62 41.26	66,941,711 102,782,522	23.07 35.49	42% success		**2,613,043** **4,106,440**	**1.46** **2.31**
par16-4-c	11,146,298 16,938,387	4.81 7.22	121,641,036 220,317,867	46.73 84.22	80,419,118 112,779,831	27.91 38.67	69% success		**1,034,957** **2,182,801**	**0.57** **1.21**
par16-5-c	11,829,928 17,544,960	5.02 7.44	61% success		90,819,874 126,328,695	31.23 43.84	41% success		**3,169,028** **4,092,230**	**1.74** **2.24**
qg1-08	647,290 920,411	15.64 22.48	**539,888** **950,976**	**1.90** **3.28**	99% success		59% success		80% success	
qg2-08	2,545,216 3,294,816	51.12 69.99	**3,948,297** **5,859,212**	**13.94** **20.68**	43% success		36% success		20% success	
qg5-11	99% success		74% success		1% success		**2,287,392** **3,287,772**	**24.57** **35.40**	22% success	
qg6-09	726,178 3,090,041	2.23 9.32	**20,455** **95,756**	**0.04** **0.15**	14% success		28,846 43,886	0.11 0.17	832,520 1,263,106	3.29 5.00
qg7-13	**98% success**		31% success		0% success		3% success		0% success	

(although G^{2}WSAT also performs well in this class). Of the other algorithms, PAWS is the best for the parity problems, with G^{2}WSAT and PAWS coming about equal first on the large graph colouring problems, and RSAPS winning on one quasi-group and one large graph instance. On this basis gNovelty$^+$ emerges as the best algorithm both in terms of the number of problems (ten) and the number of problem classes (four) in which it dominates.

An even clearer picture emerges when we look at the overall proportion of runs that completed within 600 seconds. Here, gNovelty$^+$ achieves a 99.88% success rate compared with 85.63% for AdaptNovelty$^+$, 85.58% for G^{2}WSAT, 80.08% for PAWS and 63.92% for RSAPS. This result is reinforced in the run time distributions (RTDs) on the left-hand of Figure 2 where the gNovelty$^+$ curve dominates over the entire time range.

Overall, gNovelty$^+$ not only outperforms the other techniques in the greatest number of problem classes, it is within an order of magnitude of the best performing algorithms in all remaining cases. It is this robust average case performance (that gNovelty$^+$ also demonstrated in the SAT competition) that argues strongly for its usefulness as a general purpose solver.

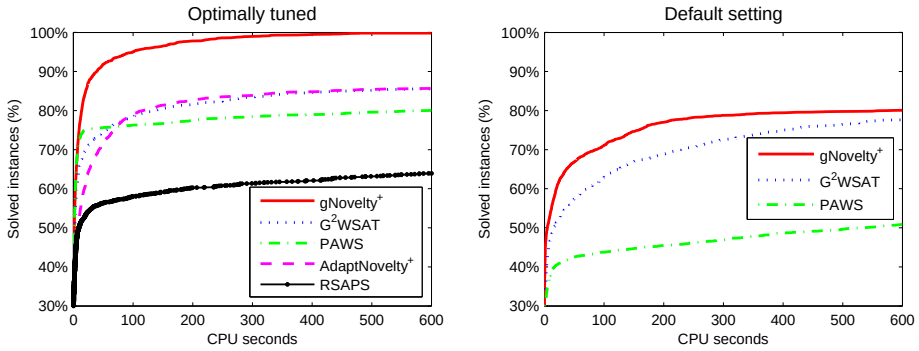

Fig. 2. Run-time distributions over the complete data set

However, if such robust behaviour depends critically on manually tuned parameter settings then the case for gNovelty$^+$ must weaken. To evaluate this we tested gNovelty$^+$ on the same problem set with a default *sp* value of 0.00 (meaning clause weights are increased in each local minimum but never decreased) and with the noise parameter adjusted using gNovelty$^+$'s adapt mechanism.[7] These results and the results of the default parameter values for G^{2}WSAT ($dp = 0.05$ and noise $= 0.5$) and PAWS ($w_{inc} = 10$) are shown in Table 3.

In this second comparison, the results of RSAPS and AdaptNovelty$^+$ from Table 2 should also be considered (as both algorithms were run without the benefit of manual tuning). Tables 2 and 3 show that AdaptNovelty$^+$ now has the best overall success rate of 85.63% followed by the default valued gNovelty$^+$ at 80.13%, G^{2}WSAT at 77.67%, with RSAPS (63.92%) and PAWS (50.88%) coming last (this is also illustrated in the RTDs in Figure 2).

However, the situation changes when we consider individual problem categories, with gNovelty$^+$ dominating in five classes (3bitadd, ais, bw_large, flat and logistics), AdaptNovelty$^+$ in two (g and par16) and G^{2}WSAT on one (qg). Looking in more detail, we can see that the main negative impact of a fixed parameter on gNovelty$^+$ has come from its failure on the parity problems. Similarly, AdaptNovelty$^+$ fails only on the quasi-group problems. If we put these two data sets aside, then the default gNovelty$^+$ shows a clear advantage over AdaptNovelty$^+$, dominating on five of the remaining six problem classes.

5 Discussion and Conclusions

The experimental evidence of this paper and the SAT competition demonstrates that gNovelty$^+$ is a highly competitive algorithm for random SAT problems, and, with parameter tuning, that it can dominate several of the previously best

[7] Although gNovelty$^+$'s noise parameter was also adjusted in Table 1, performance was not greatly improved, with the main benefits coming from adjusting *sp*.

Table 3. Default parameter setting results shown in the form: $\frac{median}{mean}$

Instances	gNovelty$^+$		G^{2}WSAT		PAWS	
	#flips	#secs	#flips	#secs	#flips	#secs
3bitadd_31	**17,948** / **19,746**	**0.06** / **0.07**	0% success		0% success	
3bitadd_32	**15,116** / **15,510**	**0.06** / **0.06**	0% success		0% success	
ais10	**8,116** / **11,596**	**0.01** / **0.01**	79,932 / 112,419	0.07 / 0.09	68,462 / 94,817	0.08 / 0.11
ais12	**48,480** / **63,938**	**0.06** / **0.08**	1,073,489 / 1,661,896	1.10 / 1.72	917,232 / 1,424,442	1.41 / 2.17
bw_large.c	**831,984** / **1,131,058**	**1.49** / **2.04**	53% success		7,237,556 / 9,440,193	17.31 / 22.49
bw_large.d	**4,802,306** / **5,498,694**	**14.52** / **16.74**	0% success		39% success	
flat200-med	163,566 / 241,467	0.07 / 0.10	**111,942** / **168,466**	**0.04** / **0.07**	145,014 / 180,628	0.08 / 0.10
flat200-har	**2,576,103** / **4,037,105**	**1.04** / **1.64**	5,422,588 / 13,873,574	2.00 / 4.89	5,922,780 / 8,168,341	3.12 / 4.30
g125.17	**3,524,594** / **4,229,083**	**15.89** / **18.81**	99% success		6% success	
g125.18	185,058 / 184,953	0.97 / 0.95	**12,314** / **13,472**	**0.16** / **0.17**	18,774 / 24,911	0.15 / 0.19
g250.15	2,254 / 2,283	**0.08** / **0.08**	2,445 / 2,475	0.43 / 0.46	**2,211** / **2,236**	0.11 / 0.11
g250.29	**4,984,444** / **5,008,310**	**136.75** / **141.15**	47% success		0% success	
logistics.c	**6,332** / **6,873**	**0.01** / **0.01**	36,389 / 54,279	0.03 / 0.04	118,259 / 157,790	0.10 / 0.13
logistics.d	**26,880** / **32,211**	**0.04** / **0.04**	2,056,840 / 3,050,451	1.75 / 2.56	275,307 / 385,814	0.28 / 0.38
par16-1-c	20% success		**97% success**		23% success	
par16-2-c	10% success		**85% success**		1% success	
par16-3-c	5% success		**114,988,942** / **166,070,580**	**44.78** / **64.59**	5% success	
par16-4-c	9% success		**98% success**		16% success	
par16-5-c	5% success		**52% success**		3% success	
qg1-08	**852,874** / **1,133,742**	18.90 / 24.94	1,175,202 / 1,660,389	**4.45** / **6.25**	83% success	
qg2-08	**3,154,862** / **4,092,638**	68.67 / 91.04	5,954,183 / 9,058,721	**20.96** / **31.67**	23% success	
qg5-11	5,012,010 / 6,863,167	39.12 / 50.50	**1,237,121** / **1,979,199**	**7.10** / **10.19**	22% success	
qg6-09	342,864 / 2,281,251	1.16 / 7.32	**11,376** / **19,469**	**0.03** / **0.04**	968,044 / 1,229,162	4.04 / 5.10
qg7-13	**74% success**		33% success		0% success	

performing SLS algorithms on a range of structured problems. If parameter tuning is ruled out (as it would be in most real-world problem scenarios), then gNovelty$^+$ still performs well, and only lost to its closest rival, AdaptNovelty$^+$, on one structured problem class.

Once again, as with PAWS and SAPS, the addition of a clause weighting heuristic to gNovelty$^+$ has required the addition of a sensitive weight decay parameter to get competitive results. Nevertheless, the situation with gNovelty$^+$'s parameter does differ from SAPS and PAWS in that highly competitive performance can be obtained from a relatively small set of parameter values (i.e. 0.0, 0.1, 0.4 and 1.0). In contrast, SAPS and PAWS require much finer distinctions in parameter values to get even acceptable results [10]. This smaller set of values means that the process of tuning *sp* is considerably simplified in comparison to other clause weight techniques. More importantly, gNovelty$^+$'s more robust behaviour indicates that it may be easier to devise an automatic adapting mechanism for *sp*. To date, procedures for automatically adapting weight decay

parameters have not produced the fastest algorithms.[8] In future work, it therefore appears promising to try and develop a simple heuristic that will effectively adapt *sp* in the structured problem domain.

In conclusion, we have introduced the new SLS random satisfiable problem category winner of the SAT 2007 competition, gNovelty$^+$. We have extended the SAT 2007 results and shown that gNovelty$^+$ is also effective in solving structured SAT problems. In fact, gNovelty$^+$ has not only outperformed four of the strongest current SLS SAT solvers, it has also demonstrated significant robustness in solving a wide range of diverse problems. In achieving this performance, we have highlighted gNovelty$^+$'s partial dependence on the setting of its *sp* weight decay parameter. This leads us to recommend that future work should concentrate on the automatic adaptation of this parameter.

Acknowledgements: We thankfully acknowledge the financial support from NICTA and the Queensland government. NICTA is funded by the Australian Government's *Backing Australia's Ability* initiative, and in part through the Australian Research Council.

References

1. Davis, M., Logemann, G., Loveland, D.: A machine program for theorem proving. Communications of the ACM **5**(7) (1962) 394–397
2. Selman, B., Levesque, H., Mitchell, D.: A new method for solving hard satisfiability problems. In: Proceedings of AAAI-92. (1992) 440–446
3. McAllester, D.A., Selman, B., Kautz, H.A.: Evidence for invariants in local search. In: Proceedings of AAAI-97. (1997) 321–326
4. Morris, P.: The Breakout method for escaping from local minima. In: Proceedings of AAAI-93. (1993) 40–45
5. Anbulagan, Pham, D.N., Slaney, J., Sattar, A.: Old resolution meets modern SLS. In: Proceedings of AAAI-05. (2005) 354–359
6. Li, C.M., Huang, W.Q.: Diversification and determinism in local search for satisfiability. In: Proceedings of SAT-05. (2005) 158–172
7. Hoos, H.H.: On the run-time behaviour of stochastic local search algorithms for SAT. In: Proceedings of AAAI-99. (1999) 661–666
8. Hoos, H.H.: An adaptive noise mechanism for WalkSAT. In: Proceedings of AAAI-02. (2002) 635–660
9. Hutter, F., Tompkins, D.A., Hoos, H.H.: Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In: Proceedings of CP-02. (2002) 233–248
10. Thornton, J.R.: Clause weighting local search for SAT. Journal of Automated Reasoning **35**(1-3) (2005) 97–142
11. Li, C.M., Wei, W., Zhang, H.: Combining adaptive noise and look-ahead in local search for SAT. In: Proceedings of LSCS-06 Workshop. (2006) 2–16
12. Hutter, F., Hamadi, Y., Hoos, H.H., Leyton-Brown, K.: Performance prediction and automated tuning of randomized and parametric algorithms. In: Proceedings of CP-06. (2006) 213–228

[8] Although machine learning techniques that are trained on test sets of existing instances and then applied to unseen instances have proved useful for setting SAPS and Novelty parameters [12]