# Estimating Problem Metrics for SAT Clause Weighting Local Search

Wayne Pullan, Liang Zhao, and John Thornton

School of Information Technology,
Griffith University, Gold Coast,
Qld., 4215, Australia
Email: {w.pullan, j.thornton}@griffith.edu.au

**Keywords**: Constraints, Search

**Abstract.** Considerable progress has recently been made in using clause weighting algorithms to solve SAT benchmark problems. While these algorithms have outperformed earlier stochastic techniques on many larger problems, this improvement has generally required extra, problem specific, parameters which have to be fine tuned to problem domains to obtain optimal run-time performance. In a previous paper, the use of parameters, specifically in relation to the DLM clause weighting algorithm, was examined to identify underlying features in clause weighting that could be used to eliminate or predict workable parameter settings. A simplified clause weighting algorithm (Maxage), based on DLM, was proposed that reduced the parameters to a single parameter. Also, in a previous paper, the structure of SAT problems was investigated and a measure developed which allowed the classification of SAT problems into random, loosely structured or compactly structured. This paper extends this work by investigating the behaviour of Maxage with regard to the structural characteristics of SAT problems. The underlying motivation for this study is the development of an adaptive, parameterless clause weighting algorithm.

## 1 Introduction

The propositional satisfiability (SAT) problem is fundamental in solving many practical problems in mathematical logic, inference, machine learning, constraint satisfaction, and VLSI engineering. Theoretically, the SAT problem is the *core* of a large family of computationally intractable NP-complete problems. Several such NP-complete problems have been identified as central to a variety of areas in computing theory and engineering. Therefore, methods to solve the satisfiability problem play an important role in the development of computing theory and systems.

In this paper, as with most other work on SAT algorithms, we only consider propositional formulae in conjunctive normal form. That is, formulae of the form $F = \bigwedge_i \bigvee_j l_{ij}$ where each $l_{ij}$ is a propositional variable or its negation. The $l_{ij}$ are termed *literals* while the disjunctions $\bigvee_j l_{ij}$ are the *clauses* of $F$. The goal of all SAT algorithms is to find an assignment of the truth values to the propositional variables in $F$ that results in no unsatisfied (*false*) clauses.

Algorithms for solving SAT problems can be divided into two categories: *complete* and *incomplete*. Complete SAT algorithms perform a systematic traversal of the search

space and will always find a solution if one exists. Incomplete SAT algorithms are stochastic algorithms in that they may find a solution but, if they fail, it cannot be concluded that no solution exists.

Some of the best known incomplete SAT algorithms are local search algorithms that, while they differ in detail, all basically implement a local search strategy which starts with an initial random assignment of truth values to all propositional variables. In each subsequent search step, the algorithm selects a variable using some heuristic and negates the truth value of that variable (i.e. *true* to *false* or *false* to *true*). Variable negations are typically performed with the goal of minimising an objective function based on the currently unsatisfied clauses.

A version of local search, that has recently become very effective in SAT problem solving, modifies the objective function by associating a weight with every clause of the given formula. These algorithms then aim to minimise the total weighted objective function rather than the number of unsatisfied clauses. By appropriate manipulation of clause weights, these clause weighting algorithms are able to escape from local minima and other attractive non-solution areas in the underlying search space. The major, inherent disadvantage in these algorithms is that additional, problem dependent, parameters are required to control the clause weights.

In 1993, clause weighting local search algorithms for SAT were simultaneously proposed in [5] and [8]. Various enhancements to clause weighting followed in the mid-90s, particularly Jeremy Frank's work on multiplicative weighting and weight decay [1]. Early clause weighting algorithms avoided plateau search by adding weight to all unsatisfied clauses as soon as a plateau was encountered [5]. However, it was not until the development of DLM [9] that these insights were translated into significant performance improvements. The main differences between Discrete Lagrangian Multiplier (DLM) and earlier clause weighting techniques are in the use of a tabu list [2] to guide the search over plateau areas, and a weight reduction heuristic that periodically reduces clause weights. The Smoothed Descent and Flood (SDF) algorithm [6] uses multiplicative weighting and a continuous renormalisation of relative weights after each increase. While SDF produced some improvement over DLM in terms of the number of variable negations required on smaller sized problems, there is a significant run-time overhead in maintaining SDF's real valued weights. SDF subsequently evolved into the Exponentiated Sub-Gradient (ESG) method [7] which has been improved on by the Scaling and Probabilistic Smoothing (SAPS) [3] method.

Another recent algorithm is Maxage [10], which is based on DLM and has comparable performance but the problem dependent parameters have been reduced from 14 to just a single parameter, the DECREASE parameter, which controls the frequency with which clause weights are decayed. The major objective of this paper is to determine if an estimate of the optimal value of the Maxage DECREASE parameter can be determined from characteristics of SAT problems. The underlying motivation for this study is the development of an adaptive, parameterless clause weighting algorithm, based on Maxage, that determines an initial estimate for DECREASE and then, using measurable characteristics of the search process, adaptively modifies DECREASE to produce an effective search.

This paper is structured as follows. Section 2 presents a more detailed discussion of clause weighting, particularly with reference to Maxage, while Section 3 contains a description of SAT problem characteristics. An analysis of four benchmark SAT problems, with regard to these characteristics, is presented in Section 4 and Section 5 presents an analysis of how these characteristics influence the Maxage DECREASE parameter. Finally, Section 6 contains a conclusion and suggestions for future research.

## 2 Local Search

At a high level, a local search algorithm can be viewed as a mechanism for traversing a highly multi-dimensional hyper-surface with the objective of locating a global minima. While features of the hyper-surface may be extremely complicated, the perception of the hyper-surface by the local search algorithm is very limited in that it only knows the hyper-surface immediately adjacent to its current position and has no memory or knowledge of the hyper-surface in any other location. In fact, its knowledge is limited to, for a single step in any given direction, the rate at which the hyper-surface is changing. To efficiently traverse the hyper-surface the local search algorithm must avoid:

- **Search Cycles** which are basically some configuration of closed paths on the hyper-surface. They arise because of the presence of local minima or some combination of other hyper-surface features.
- **Unguided Travel** which occurs when the hyper-surface is locally flat (plateau) and there is no basic, under-lying guidance for the search.

Search cycles that have short path lengths can be successfully handled by mechanisms such as Tabu lists [2], which prevent re-selection of a variable before some number of other variables have been modified. However, search cycles with longer path lengths or a variety of paths are much more difficult to detect and escape from. Tabu lists can also have a beneficial effect when traversing a hyper-surface plateau as they tend to provide an underlying direction for the search.

### 2.1 Non-clause Weighting Local Search Algorithms

The hyper-surface traversed by non-clause weighting local search algorithms for SAT problems is generally that formed by evaluating the number of false clauses for each assignment of variables identified during the search. That is, the local search is performing the global optimisation problem (for a SAT problem with $n$ variables $(x_1, \ldots, x_n)$ and $m$ clauses $(c_1, \ldots, c_m)$):

$$min \;\; f(x_1, \ldots, x_n) = \sum_{i=1}^{m} b_i \tag{1}$$

where $b_i = 0$ if clause $c_i$ is *true* and $b_i = 1$ if clause $c_i$ is *false*. At each step in the search, these algorithms evaluate the effect of negating each variable in terms of the reduction in the number of false clauses and will generally select the variable which causes the largest decrease in $f$. The fundamental differences in these algorithms are

typically in the tie-breaking rules, if more than one variable gives the best decrease, and how they handle search cycles and plateaus on the hyper-surface (random moves, random restarts and Tabu lists).

### 2.2 Clause Weighting Local Search Algorithms

Clause weighting local search algorithms traverse the weighted false clause hyper-surface formed by the weighted cost of a problem solution. That is, a clause weighting local search is addressing the global optimisation problem:

$$min \;\; g(x_1, \ldots, x_n) = \sum_{i=1}^{m} w_i b_i, \quad w_i \geq 1 \tag{2}$$

where $w_i$ is the weight associated with clause $c_i$. At each step in the search, these algorithms evaluate the effect of negating each variable in terms of the reduction in the weighted cost of the false clauses and will generally select that variable which causes the largest decrease in $g$. Clause weights act to deform the weighted false clause hyper-surface ($S_g$) from the false clause hyper-surface ($S_f$) and are typically incremented whenever a local minimum or extended plateau is found by the algorithm. This action tends to remove local minima [5] and plateaus from $S_g$. To prevent $S_g$ from becoming too deformed and "rugged" (and thus losing any natural underlying guidance from $S_f$), a clause weight reduction mechanism is normally incorporated into the search.

Clause weighting local search algorithms tend to focus on satisfying the more difficult clauses of a SAT problem as the weights for clauses that are difficult to satisfy will, on average, be higher than those for clauses that are easier to satisfy. This will make satisfying these clauses more attractive to the algorithm and is analogous to the common problem solving heuristic of attacking the most difficult parts of a problem first, and then addressing the less constrained resources until a solution is found. It is also important to note that all global minima, corresponding to $f = g = 0$, will be in the same positions on $S_g$ as they are on $S_f$. If this were not the case, clause weighting local search algorithms would be at a considerable disadvantage to non-clause weighting local search algorithms in that they would be trying to locate global minima that are moving on $S_g$ as clause weights change (as is the case when the global minima are such that $f > 0$).

There are some inherent disadvantages in clause weighting algorithms, namely that additional, problem dependent parameters are required to control the clause weighting. These include the amount by which to increase or decrease the clause weights and at what point reweighting should occur. Also, the possibility of clause weighting cycles exists where clause weights are repetitively increased and decreased causing a search cycle in the sequence of variables to be negated.

### 2.3 Clause Weighting in Maxage

As shown in Fig. 1, Maxage [10] has only one parameter, the DECREASE clause weighting parameter, which specifies how many clause weight increases must occur before a clause weight reduction is performed. Unfortunately, as with the parameters

for DLM, the Maxage DECREASE parameter is problem dependent and must be pre-determined for each particular class of problem. This typically requires performing a large number of experiments, where DECREASE is systematically varied, to identify the optimal value of DECREASE for that class of problem. Another issue is that these pre-determined values are a compromise in that they are the optimal, on average, over the entire search. It is reasonable to assume that the actual optimal value for DECREASE, at any point in the search, depends on the nature of $S_f$ at the current location of the search and this varies, with most problems, during the search.

**procedure** *MAX-AGE*
**begin**
  Generate a random starting point
  Initialise counters and clause weights to zero
  **while** solution not found and $flips <$ maxFlips **do**
    $B \leftarrow$ set of best weighted cost single flip moves
    **if** no improving $x \in B$ **then**
      **if** oldest $x \in B$ has $age(x) \geq maxAge$ **then**
        $B \leftarrow x$
        $maxAge \leftarrow maxAge + 1$
      **else if** $random(p) \leq P$ **then**
        $B \leftarrow \emptyset$
      **end if**
    **end if**
    **if** $B \neq \emptyset$ **then**
      Randomly pick and flip $x \in B$
      $age(x) \leftarrow {+}{+}flips$
    **else**
      Increase weight on all false clauses
      **if** $+{+}increases$ % DECREASE = 0 **then**
        Decrease weight on all weighted clauses
    **end if**
  **end while**
**end**

**Fig. 1.** The MAX-AGE Algorithm

When DECREASE is very close to one, clause weighting has little effect as clause weights are rapidly returned to their default value. That is, $g$ tends to be very close to $f$ and there is relatively little influence on the search from clause weighting. When DECREASE is large, clause weights tend to become large as the downward re-weighting of clauses is performed less frequently. This tends to make the difference between $f$ and $g$ more significant and $g$ becomes a more "rugged" function than $f$, in the sense that there is greater potential for the change in $g$ to be larger than that for $f$ at each step of the search.

### 2.4 Determination of Optimal DECREASE for Maxage

The benchmark problems that will be used in this study are four of the problem domains for which existing Maxage parameters have already been developed, namely random 3-SAT, parity learning, graph colouring and blocks world planning. Using the SATLIB [1] and DIMACS[2] benchmark libraries, we selected *f2000* for random 3-SAT, *par16-1-c* for parity learning, *bw_large.c* for blocks world planning and *g125.17* for the graph colouring problem. These problems provide a range of SAT problem types, from random to structured. In addition, they provide the largest range for the Maxage DECREASE parameter and all have reasonable computational requirements (which ensures that clause weighting becomes a relevant factor in the search).

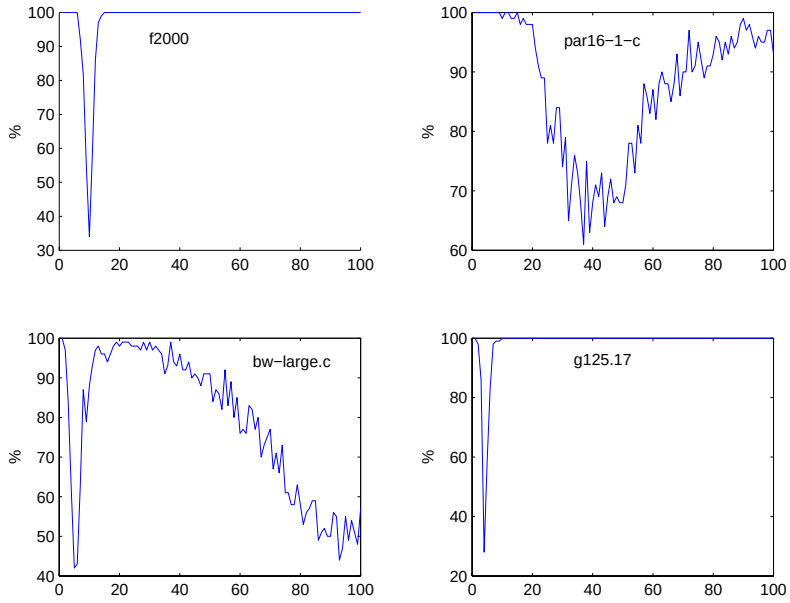


**Fig. 2.** The failure rate for Maxage as DECREASE varies from $1 \ldots 100$ . For each value of DECREASE, Maxage was executed from 100 random starting points with a maximum of 1,000,000 steps allowed. The vertical axis records the number of times for which Maxage failed to find a global minima.

The failure rate of Maxage, for a maximum of 1,000,000 steps, as DECREASE varies in the range $1 \ldots 100$ is shown in Fig. 2 for each of the four benchmark problems. From the data presented in the graphs, we see that the optimal DECREASE is

[1] http://www.intellektik.informatik.tu-darmstadt.de/SATLIB/
[2] ftp://dimacs.rutgers.edu/pub/challenge/satisfiability/benchmarks/cnf/

large (46) for *par16-1-c*, is smaller (10) for the randomly generated *f2000*, is six [3] for *bw_large.c* and is smallest (four) for *g125.17*. Clearly some aspects of the structure of SAT problems have an impact on the optimal value for DECREASE and this question is investigated in the next section.

## 3 SAT Problem Characteristics

The overall goal of this section is, from a static analysis of a SAT problem, to identify characteristics of $S_f$ that are relevant to the optimal setting for the DECREASE parameter of Maxage. Using statistical characteristics from the uniformly randomly generated SAT problem *f2000* as a basis, we categorise other SAT problems by comparing their statistical properties with those of randomly generated SAT problems. Random SAT problems are constructed by first uniformly randomly selecting the variables for each clause and then, with probability $0.5$, negating each variable. This generation technique ensures that the distributions of most characteristics will closely approximate the normal distribution.

Descriptions of the SAT problem characteristics used in this study are presented in the following sub-sections while Section 4 contains an analysis of the four SAT benchmark problems with regard to these characteristics.

- **Distribution of Variable Usage** For a randomly generated SAT problem, the expected number of times that $x_i$ will occur in clauses is $u_i = M/n$ where $M$ is the total number of literals in the problem (e.g., $M = 3m$ for 3-SAT) and $n$ is the number of variables. As variables are uniformly randomly selected, $u_i$ will be normally distributed. A simple analysis of a SAT problem produces the distribution of $u_i$ for that problem and a $\chi^2$ test with the normal distribution, median $M/n$, produces an estimate as to whether the usage of variables for the problem is random, with regard to this measure.
- **Independent Versus Dependent Variables** By analysing the SAT problem and looking for patterns in the relationships between variables, variables may be classified as either independent or dependent. For a SAT problem, dependent variables are those whose values can be determined once all independent variables have been assigned. In addition, clauses may be classified as consisting only of independent variables, only of dependent variables or a combination. Randomly generated problems are extremely unlikely to contain dependent variables.
  In addition to providing a test for randomness, this measure could also be used to limit a local search to only the independent variables which, in some problems, could provide improved search performance.
- **Variable Relationships** In a previous paper, [4], a classification of k-SAT problems as either uniform, compact or loose on the basis of variable connectivity was described. This classification method was developed as a test on problem uniformity based on a simple statistic, the count of the number of distinct variable pairs that occur in clauses. With this test, problems can be classified uniform (random), loosely or compactly structured.

[3] The computational requirements when DECREASE is large for *bw_large.c* are much greater than when DECREASE is six.

– **Distribution of Variable Signs** If we define $u_i^d$ as the absolute difference between the number of positive occurrences ($u_i^+$) and the number of negated occurrences ($u_i^-$) of $x_i$ in clauses then, for a random problem, the distribution of $u_i^d$ will approximate a normal distribution with median zero. By comparing the distribution of $u_i^d$ for a SAT problem with such a normal distribution, problems can be classified as either random or non-random with regard to this measure.
In addition, the distributions of $u_i^+$ and $u_i^-$ are indicators of the structure of the $S_f$ being traversed by the search. They place a range on the maximum value of $f_i$, the change in $f$ caused by negating $x_i$. Clearly, $-u_i^+ \leq f_i^+ \leq u_i^-$, where $f_i^+$ denotes the change in the number of *false* clauses when $x_i$ goes from *false* to *true*. Correspondingly, $-u_i^- \leq f_i^- \leq u_i^+$, where $f_i^-$ denotes the change in the number of *false* clauses when $x_i$ goes from *true* to *false*. Clearly, if $u_i^+$ and $u_i^-$ are small, both $f_i^+$ and $f_i^-$ will also be small. Of interest also is, how are $f_i^+$ and $f_i^-$ distributed within these ranges. Our hypothesis is that if $u_i^d$ is close to zero, then this will bias both $f_i^+$ and $f_i^-$ towards the zero point of their ranges. The rationale for this is that a small $u_i^d$ states that there are potentially as many clauses which will go from *true* to *false* as will go from *false* to *true* when $x_i$ is negated. Conversely, if $u_i^d$ is not close to zero, then this will bias both $f_i^+$ and $f_i^-$ towards one of the extremes of their ranges.
Supporting the argument that the distribution of $u_i^d$ can be used as an estimator of the structure of $S_f$ is the degenerate 1-SAT case where all clauses only contain a single variable. Clearly, the distribution of $u_i^d$ directly reflects the characteristics of $S_f$. For k-SAT ($k > 1$), there are situations where negating a variable will not falsify a clause (where both variables in the clause were *true*) so, in these cases, the distribution of $u_i^d$ is a worst case estimator of the characteristics of the $S_f$.

## 4 Structure of Benchmark Problems

As expected, *f2000* approximated the normal distribution with a median of 12 while the distributions for the other problems showed that there is a wide range in the usage of variables in clauses. In addition, the counts of independent and dependent variables and clauses for the four benchmark problems showed that both *par16-1-c* and *bw_large.c* are classified as non-random problems. With regards to the distributions of variable signs and with reference to Fig. 3, the following points can be made for each of the benchmark problems:

– **f2000**. For the randomly generated problem *f2000*, the distributions of $u_i^+$ and $u_i^-$ have medians around 6.4 and there are no outliers. From the distribution of $u_i^d$ it can be seen that for approximately 12% of variables, $u_i^d = 0$, the median is 0.045 and the variance is 13.2. From these observations it is reasonable to conclude that, $S_f$ for *f2000* will have a relatively small proportion of plateaus and the other areas will contain relatively small features. Accordingly, in this study, we classify $S_f$ for *f2000* as "choppy" and expect $f_i$ to be relatively small as each variable is negated.
– **par16-1-c**. With regard to $u_i^+$ and $u_i^-$, their distributions have medians $\pm 5.7$ and it can be seen that there are just a few large outliers. From the distribution of $u_i^d$ it can be seen that for approximately 80% of variables, $u_i^d = 0$ and the variance is

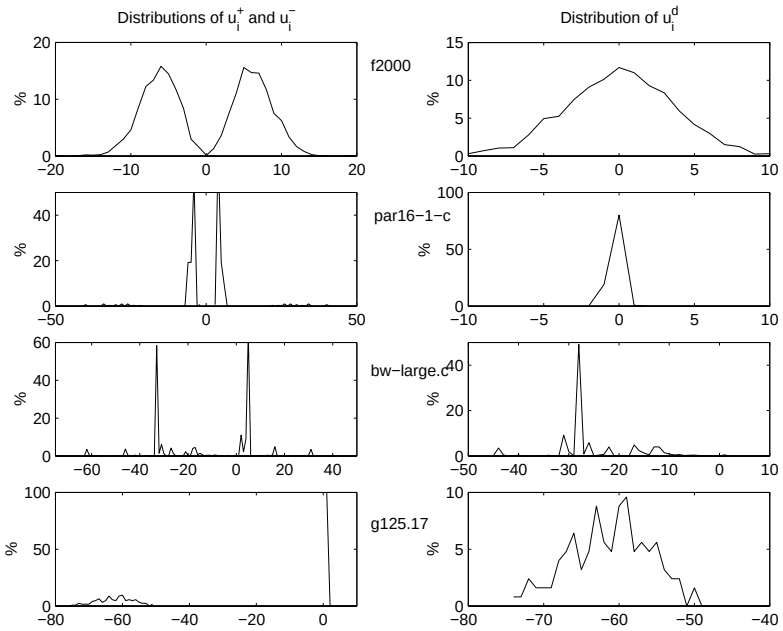


**Fig. 3.** Distributions of $u_i^+$, $u_i^-$ and $u_i^d$ for each of the four benchmark problems. The vertical axis is, in each plot, the percentage frequency for which the horizontal axis value appeared. For the distributions of $u_i^+$ and $u_i^-$, $u_i^+$ is plotted as positive values and $u_i^-$ as negative values.

0.17. Both these factors place a small range on $f_i^+$ and $f_i^-$ resulting in a relatively smooth $S_f$ for *par16-1-c*. That is, $S_f$ for *par16-1-c* contains a higher proportion of plateaus than $S_f$ for *f2000* and also contains some areas with approximately the same features as *f2000*. In this study, we classify $S_f$ for *par16-1-c* as "flat" and expect $f_i$ to be predominantly zero, but not infrequently small, as each variable is negated.

– **bw_large.c** For the *bw_large.c* problem, the distribution for $u_i^-$ has a median of -31.0 while that for $u_i^+$ is 6.0. This gives a large range for both $f_i^+$ and $f_i^-$. From the distribution of $u_i^d$ it can be seen that it is has a median of -25.0 with a variance of 62.0. This implies that $f_i^+$ and $f_i^-$ will be towards an extreme of their ranges. From these observations it is reasonable to conclude that, $S_f$ for *bw_large.c* will have virtually no plateaus and mainly consists of relatively large features. Accordingly, in this study, we classify $S_f$ for *bw_large.c* as "moderate" and expect $f_i$ to be relatively large as each variable is negated.

– **g125.17** For the *g125.17* problem, the distribution for $u_i^-$ has a median of -62 while that for $u_i^+$ is one. This gives a large range for both $f_i^+$ and $f_i^-$. From the distribution of $u_i^d$ it can be seen that it is has a median of -61 with a variance of 27. This implies that $f_i^+$ and $f_i^-$ will be towards an extreme of their ranges. From these observations it is reasonable to conclude that, $S_f$ for *g125.17* will have virtually

no plateaus and mainly consists of large features. Accordingly, in this study, we classify $S_f$ for *g125.17* as "rugged" and expect $f_i$ to be large as each variable is negated.

To confirm the observations detailed above, random sampling of $S_f$ was performed for the benchmark problems and the results are presented in Fig. 4. This random sampling was performed by first generating uniformly random assignments for variables and then evaluating the change in $f$ as each variable is negated. This process was repeated 1,000,000 times for each problem. As can be seen, the results support the arguments presented above in that $S_f$ for *f2000* contains a moderately small proportion of plateaus (22% of variable negations resulted in $f_i = 0$) and $|f_i| < 6$ in all other areas, $S_f$ for *par16-1-c* contains a larger proportion of plateaus (37% of variable negations resulted in $f_i = 0$) and $|f_i| < 5$ in all other areas. For *bw_large.c*, $S_f$ has virtually no plateaus and consists of moderately large features where $|f_i| \leq 25$, while for *g125.17*, $S_f$ has no plateaus and consists only of large features where $18 \leq |f_i| \leq 45$.

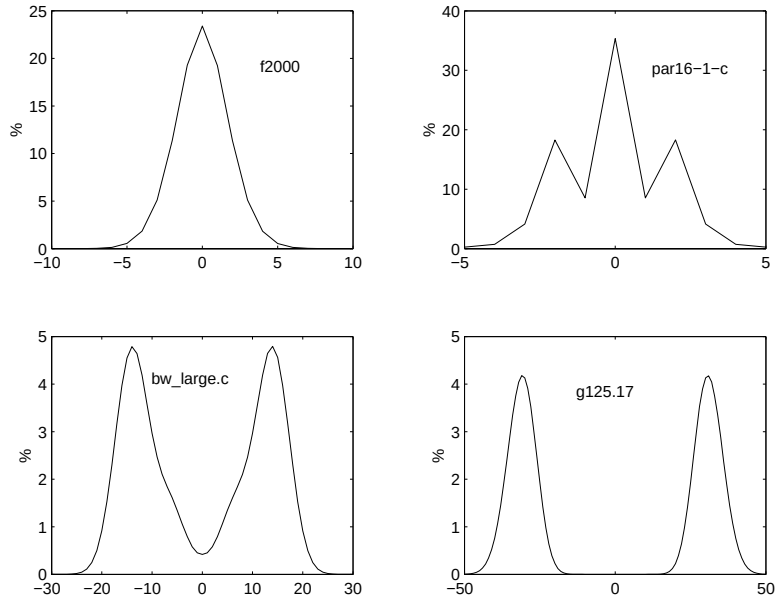


**Fig. 4.** Distributions of sampled $f_i$ for each of the four benchmark problems. The vertical axis is, in each plot, the percentage frequency for which the horizontal axis value appeared.

## 5 SAT Problem Characteristics and Maxage

The basic rationale for the analysis of SAT problems was to investigate characteristics of the benchmark problems to identify if there is any relationship between the optimal

value for DECREASE and measurable properties of a SAT problem. If this can be shown to be the case, then some form of estimate for the optimal value of DECREASE could be programmatically determined at the start of a Maxage search by an analysis of the SAT problem.

From Figs. 2 and 3 and the discussion presented above, our hypothesis is that a larger value of DECREASE (i.e. more clause weighting) will be optimal for problems where $S_f$ is relatively smooth and a smaller value optimal when $S_f$ is relatively rugged.

Intuitively, as traversing an predominantly flat $S_f$, takes some number of variable assignments, clause weighting needs to be more aggressive (higher DECREASE) to prevent unguided travel. Conversely, if there is a small proportion or no flat areas on $S_f$, the main role for clause weighting is escaping from search cycles, in particular local minima, which is a relatively localised phenomena of $S_f$ and takes relatively fewer variable assignments. Accordingly clause weighting needs to be less aggressive (lower DECREASE) so that $S_g$ stays close to $S_f$ and the inherent features of $S_f$ are used as much as possible during the search.

The measures $u_i^+$, $u_I^-$ and $u_i^d$ ranked $S_f$ for the benchmark problems as (smoothest to most rugged) *par16-1-c*, *f2000*, *bw_large.c* and *g125.17*. This is in accordance with the ranking of their optimal DECREASE value (46, 10, 6, 4). This suggests the following algorithm for setting the DECREASE parameter: If all four tests classify the SAT problem as random, then the optimal value for DECREASE is 10 otherwise, if $u_i^d, i = 1 \ldots n$ approximates a normal distribution with a median of zero, DECREASE is a linear function of the frequency of the median, otherwise DECREASE is a linear function of the median of the distribution.

The results obtained, using this algorithm, are compared to experimentally determined optimal values for DECREASE in Table 1 for the benchmark plus other representative SAT problems from different SATLIB problem classes. For the 10 non-benchmark problems, only the predicted DECREASE value for *ais10* is not in the range of optimal values. Being an under estimate, this will tend to increase the possibility of Maxage search cycles for *ais10* as it has a more "rugged" $S_f$ than those of random problems. Experiments, not reported here, with an adaptive version of Maxage which is able to identify search cycles, showed that under estimates of DECREASE for these types of problems are usually corrected during the search.

## 6 Conclusion

The aim of this study was to move towards developing an adaptive clause weighting algorithm, based on Maxage, that required no external, problem dependent parameter(s). This algorithm will analyse the SAT problem to determine an initial value for the DECREASE parameter and then adaptively modify this value during the search using run-time characteristics that determine how effectively the search is proceeding. This paper proposes a method for estimating the initial value for DECREASE.

We consider this study a step towards developing more intelligent, adaptive constraint solving technologies. Future research will include:

– the identification of structure sub-types within SAT problems to refine the initial setting of the DECREASE parameter.

| Problem | Predicted DECREASE | Optimal DECREASE | Problem | Predicted DECREASE | Optimal DECREASE |
|---|---|---|---|---|---|
| f2000 | 10 | 10 | par16-1-c | 46 | 46 |
| bw_large.c | 6 | 6 | g125.17 | 4 | 4 |
| aim-200-6_0-yes1-1 | 56 | $\geq 4$ | ais10 | 6 | $\geq 7$ |
| BMS_k3_n100_m429_140 | 15 | $\geq 6$ | flat200-1 | 7 | $\geq 7$ |
| CBS_k3_n100_m449_b90_889 | 11 | $\geq 4$ | ii32a1 | 7 | $\geq 5$ |
| RTI_k3_n100_m429_140 | 14 | $\geq 5$ | logistics.b | 7 | $\geq 5$ |
| uf100-0953 | 9 | $\geq 5$ | sw100-1 | 7 | $\leq 12$ |

**Table 1.** Predicted DECREASE compared with the experimentally determined optimal DECREASE values for benchmark and other SATLIB SAT problems.

- an investigation of why, for some SAT problems, there is a wide range of optimal DECREASE values whereas for other problems it is unique.
- the development of adaptive control mechanisms so that the DECREASE parameter can be adaptively adjusted during the search.
- an investigation to determine if clause weighting is able to identify clauses which are "globally" difficult to satisfy or if it is a localised activity which simply gives the search adequate mobility and the global minima is arrived at without any use of "global" knowledge.
- an investigation to determine what makes clause weighting algorithms such effective global optimisation algorithms. Is it in the ability to escape search cycles and the guided travel across plateaus or is it in the initial focus on satisfying the more "difficult" clauses and then addressing the "easier" clauses?
- incorporating a parameterless clause weighting algorithm as the local search within a hybrid genetic algorithm which will open the way to pool based parallel search algorithms for SAT problems.

## References


1. J. Frank. Learning short term weights for GSAT. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 384–389, 1997.
2. F. Glover. Tabu search: Part 1. *ORSA Journal on Computing*, 1(3):190–206, 1989.
3. F. Hutter, D.A.D. Tompkins, and H.H. Hoos. Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In *LNCS 2470: Proceedings of Constraint Programming, 2002*, pages 233–248, 2002.
4. O. Kravchuk, W. Pullan, J. Thornton, and A. Sattar. An investigation of variable relationships in 3-SAT problems. In *AI 2002: Advances in Artificial Intelligence*, pages 579–590, 2002.
5. P. Morris. The Breakout method for escaping local minima. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pages 40–45, 1993.
6. D. Schuurmans and F. Southey. Local search characteristics of incomplete SAT procedures. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, pages 297–302, 2000.

7. D. Schuurmans, F. Southey, and R.C. Holte. The exponentiated subgradient algorithm for heuristic boolean programming. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 334–341, 2001.
8. B. Selman and H. Kautz. Domain-independent extensions to GSAT: Solving large structured satisfiability problems. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, pages 290–295, 1993.
9. Y. Shang and B. Wah. A discrete Lagrangian-based global search method for solving satisfiability problems. *J. Global Optimization*, 12:61–99, 1998.
10. J. Thornton, W. Pullan, and J. Terry. Towards fewer parameters for SAT clause weighting algorithms. In *AI 2002: Advances in Artificial Intelligence*, pages 569–578, 2002.