Journal on Satisfiability, Boolean Modeling and Computation 4 (2007) 33-55

Research Note

SAT Algorithms for Colouring Some Special Classes of Graphs: Some Theoretical and Experimental Results

Sriyankar Acharyya srikalpa8@yahoo.co.in
Department of Computer Science & Engineering,
School of Information Technology,
West Bengal University of Technology, India

Abstract

The local search algorithm GSAT is based on the notion of Satisfiability. It has been used successfully for colouring graphs, solving instances of the 3SAT problem, planning blocks world exercises, and other applications. The runtime performance of GSAT can be considerably enhanced by the incorporation of a noise generating component such as tabu search or random walk. This has been verified experimentally on numerous occasions, but few attempts have been made to explain the observed phenomena analytically. This paper examines, in the context of graph colouring, some aspects of the role of the tabu list in GSAT. Two slightly different SAT formulations of graph colouring are considered. Three classes of graphs are defined that have the interesting property that, for certain initial partial colourings of the nodes, a proper colouring cannot be achieved by GSAT without the use of a tabu list. The minimum length of the tabu list that is necessary for solving the problem is determined for each class. It is found that this length varies considerably from case to case, and is sometimes quite large. We study experimentally the variation of runtime with the length of the tabu list to verify and further explore the theoretical results. We examine the general performance of other local search SAT methods like WalkSAT, Novelty, RNovelty, Novelty+ and RNovelty+ on these classes of graphs and to make a comparative assessment of all these methods.

KEYWORDS: *SAT algorithm, graph colouring, tabu search*

Submitted July 2007; revised November 2007; published December 2007

1. Introduction

In recent years, local search methods such as GSAT [13] have been employed with great success to solve problems in graph colouring, 3SAT, blocks world planning, and some other domains. GSAT, however, is an *incomplete* procedure, i.e., it is sometimes unable to find a solution even when one exists. The incorporation in the basic GSAT algorithm of a noise-generating component such as tabu search or random walk frequently improves the performance quite remarkably both in terms of the number of problems solved and the average time taken to solve a problem. This has been repeatedly corroborated by empirical investigations [4, 5, 11, 12]. In fact, when colouring random graphs having 100-250 nodes with 50% edge connectivity, GSAT with a tabu list performs as well as the best methods described in [9], its performance being markedly superior[1, 3] to that of other SAT variants

such as WalkSAT [12], Novelty and RNovelty [10]. This shows that GSAT with a tabu list can compete in some problem domains both with problem specific methods and with other local search methods such as Simulated Annealing. It has been experimentally observed, however, that the performance of GSAT is very sensitive to the length of the tabu list. Sometimes a graph colouring problem cannot be solved at all by GSAT without the use of a tabu list. Can this phenomenon be explained analytically?

In this paper we examine theoretically some aspects of the role of the tabu list in GSAT. We confine ourselves to the proper colouring of graphs, and demonstrate the existence of classes of graphs that, for certain *hard* initial partial colourings, cannot be properly coloured at all without the use of a tabu list[2]. The same graphs can be easy to colour for other initial colourings. For each such graph and its hard initial colouring, we determine a minimum length of the tabu list. The results are very interesting, since in some cases the minimum length of the list that is needed for solving the problem turns out to be quite large, comparable in value to the number of variables in the SAT formulation. For lengths less than this lower bound, the problem cannot be solved at all by GSAT. We then try to explain theoretically why the minimum length of the list has a particular value for a particular problem instance. Sometimes, the tabu list is needed only for a few initial iterations and its length can be decreased to zero for the remainder of the execution.

In this paper we study experimentally the variation of runtime with the length of the tabu list to check and further explore the theoretical results. For each problem, just as there is a *minlen* there is also a best length *bestlen* at which the performance of GSAT(L) is optimal, *i.e.*, the runtime is minimum. One of the objectives of this paper is to determine *bestlen* experimentally and to find out where *bestlen* lies in the range of length L. Another issue to be examined is the existence of *maxlen*, which is the maximum length of the tabu list for which GSAT(L) outputs a solution.

It is also interesting to examine the general performance of other local search SAT methods like WalkSAT, Novelty, RNovelty, Novelty+ and RNovelty+ on these classes of graphs and to make a comparative assessment of all these methods. It would be nice to know whether the general conclusions reached on the basis of the experimental results on benchmark problems (Johnson's Random Graph and Culberson's Flat Graph)[1], carry over to some special classes of graphs.

Section 2 explains how graphs can be coloured properly using Satisfiability-based methods. Two slightly different formulations, called the Basic and Extended Formulations, are described. Section 3 introduces the three classes of graphs that GSAT finds hard to colour for certain initial colourings. Sections 4 and 5 study the properties of these graphs in greater detail, and Section 6 provides the experimental results. The concluding section summarizes the paper and lists some open problems.

2. Graph Colouring by SAT

In the Graph Colouring Problem, we are provided an undirected graph G having N nodes and e edges. We have to assign colours to the nodes in such a way that no two adjacent nodes have the same colour. This problem can be formulated as a Satisfiability Problem [6]. We define a set Σ of logical variables as follows. For each node j in G and each colour m, the variable $X(j, m)$ is true if node j is coloured m and is false otherwise. Let π be a

propositional clause over the variables in Σ. Then π is composed of literals, each of which is a variable or the negation of a variable, connected together by the (inclusive) OR operator U. π is *satisfiable* if there exists a truth assignment for the variables such that π is true under the assignment. A set Π of propositional clauses over Σ is *satisfiable* if there exists a truth assignment for the variables such that every clause in Π is true under the assignment. A Type 1 clause states that two adjacent nodes cannot have the same colour. For an edge (j, k) and a colour m, the Type 1 clause has the form $X(j, m)$ U $X(k, m)$, where is the negation operator. A Type 2 clause states that each node j must be assigned at least one of the available s colours $a_1, a_2, \ldots, a_s$. It has the form $X(j,a_1)UX(j,a_2)U \ldots UX(j,a_s)$. So there are a total of Ns variables in Σ and $N + es$ clauses in Π. This formulation, called the *BasicFormulation*, does not enforce the condition that a node should be assigned at most one colour; it allows a node to be assigned multiple colours. An alternative formulation, called the *ExtendedFormulation*, ensures that a node has at most one colour and is more widely used than the Basic Formulation. It has Type 3 clauses in Π in addition to Type 1 and Type 2 clauses. A Type 3 clause states, for each pair of colours (m, m'), that a node j can be assigned at most one colour and has the form $X(j, m)$ U $X(j, m')$. There are $N.C(s, 2)$ clauses of Type 3.

2.1 GSAT(L)

The iterative algorithm GSAT [13] can be employed to colour the graph G when supplied Π as input. Initially, the variables are assigned truth values randomly. At each iteration, GSAT *flips* (i.e., complements the truth value of) the variable that, when flipped, causes the largest decrease in the number of unsatisfied clauses. Ties among variables are resolved arbitrarily. For each *try* (i.e., for each initial random choice of truth values of variables), the procedure is iterated *maxflips* times. There is an upper bound of *maxtries* on the number of tries. The two parameters *maxtries* and *maxflips* are chosen by the user. The algorithm terminates successfully if it finds a truth assignment that satisfies all the clauses in Π. It terminates unsuccessfully if it fails to find a satisfying truth assignment and exhausts all flips and tries. As suggested by [11], a tabu list can be incorporated in GSAT to enhance its effectiveness. Let GSAT(L), L ≥ 1, denote a GSAT algorithm that uses a tabu list of length L. This list is a FIFO list (i.e., a queue) that is empty at the beginning of each try. At each step the variable that gets flipped enters the list at the left end. The list holds only L entries, and excess entries fall off the right end. Variables in the list are not available for flipping. The variable to be flipped is selected randomly [5] from among the variables not in the list that cause a maximum decrease in the number of unsatisfied clauses.

2.2 USAT(L)

This algorithm proposed by us [3] differs from GSAT(L) in the manner of selection of the variable to be flipped. Here an additional restriction is imposed on the selection of the variable, namely, we are required to select the variable from an unsatisfied clause. Given L > 0, the variables selected for flipping by GSAT(L) and USAT(L) are the same when *maxgain*, the maximum gain of a variable, exceeds 0. But when *maxgain* becomes < 0, the selection of variables differs for the two algorithms. When *maxgain* = 0, in GSAT(L)

any variable v with gain = 0 that is not in the tabu list can be selected. But in USAT(L), the same variable v can get selected only if it occurs in an unsatisfied clause.

2.3 WalkSAT(L,p)

The WalkSAT algorithm [12] also starts from a randomly chosen variable assignment but the variable selection method differs from the ones used in GSAT and USAT. Here the variable is always selected from a randomly chosen unsatisfied clause. If in the chosen clause, variables can be flipped without unsatisfying other clauses, one of these is randomly chosen. Otherwise, with a given probability p a variable is randomly chosen from the clause, and with probability 1-p a variable is picked that minimizes the number of clauses that are currently satisfied but would become unsatisfied when the variable is flipped. A tabu list can also be used with WalkSAT. The algorithm with a tabu list of length L and random walk probability p is denoted by WalkSAT(L,p).

2.4 Novelty

In Novelty algorithm [10], when selecting a variable from a randomly selected unsatisfied clause, it makes use of the concept of the age of a variable, which is the flip number of the last flip of the variable. The variable having the maximum age is therefore the most recently flipped one. Consider the best and the second best variable. If the best variable is not the most recently flipped variable in the clause, then select it. Otherwise, with probability p select the second best variable, and with probability 1-p select the best variable. The Novelty algorithm is greedier than WalkSAT since only one of the two most improving variables from a clause is selected. This can lead either to a significantly improved performance or to getting stuck in local minima. Novelty can be extended to **Novelty+**[8] by incorporating random walk. At each search step the variable is randomly picked with probability wp from the selected clause, otherwise the variable is selected according to the heusristic of Novelty.

2.5 RNovelty

The RNovelty algorithm [10] is a variant of Novelty. It is identical to Novelty except when the best variable is the most recently flipped one. In this case, let n be the difference in the objective function between the best and the second best variable. There are four cases:

1. When $p < 0.5$ and $n > 1$, pick the best.
2. When $p < 0.5$ and $n = 1$, then with probability 2p pick the second best, otherwise with probability 1-2p pick the best.
3. When $p \geq 0.5$ and $n = 1$, pick the second best.-3pt
4. When $p \geq 0.5$ and $n > 1$, then with probability 2(p - 0.5) pick the second best, otherwise with probability 1-2(p-0.5) pick the best.

The variable selection strategy in RNovelty is even more deterministic than in Novelty. Since the pure RNovelty algorithm gets too easily stuck in local minima, a simple loop breaking strategy is used. Every 100 flips, a variable is randomly chosen from the selected clause and flipped. However it is not clear whether one random walk step every 100 steps

is sufficient for effective escape from local minima. RNovelty can be extended to **RNovelty+**[8] by incorporating random walk. At each search step the variable is randomly picked with probability wp from the selected clause, otherwise the variable is selected according to the heusristic of RNovelty. Now there is no need of using loop breaking strategy mentioned above and it is omitted.

3. Description of Problem

We now examine some classes of graphs that exhibit interesting properties when coloured using GSAT.

1. $G(P, r)$ be an undirected graph having $N = P + r$ nodes, where P is an even integer ≥ 6, and r is an integer ≥ 1. P nodes are arranged in a circle with adjacent nodes connected by an edge. Initially, we assume for simplicity that P is a multiple of 6. The circle encloses a complete r-graph K_r. Every node on the outer circle is connected by an edge to every node in K_r. Thus the graph has a total of 2P edges for $r = 1$, and $P(1+r)$ + C(r,2) edges for r ≥ 2, where C(r,2) denotes the number of ways of choosing two objects from r.

2. It is easy to see that the chromatic number [7] of $G(P, r)$ is $s = r+2$. K_r can be coloured with r colours, and the nodes on the circle can be coloured with the two remaining colours. It is clearly not possible to colour $G(P, r)$ properly with less than $r+2$ colours.

3. The *hard initial colouring* I_0 partially colours $G(P, r)$ with s colours in the following way. Colours a, b, and c are assigned to the nodes 1, 2,..., P in the outer circle in rotation, *i.e.*, in the order $a, b, c, a, b, c, ...$ Nodes $P+1$, $P+2$, , $P+r$-1 of K_r are coloured with the remaining s-3 (= r-1) colours d, e, f,..., and node $P + r$ in K_r is left uncoloured. Thus I_0 never assigns the same colour to two adjacent nodes.

When $P = 6$ and $r = 1$, we get the smallest member G(6,1) of our set of graphs (see Fig 1). This is a wheel with six nodes on the perimeter, one node at the centre, and 12 edges. I_0 colours the perimeter with three colours a, b, c in rotation and leaves the central node uncoloured. We have to colour this graph properly with three colours. Such a colouring obviously exists. The basic GSAT algorithm cannot solve this simple problem without the help of a tabu list. Of course, GSAT can also solve the problem if a random walk component is incorporated, but this is not of interest to us in this paper.
It is not essential for P to be a multiple of 6. We only require that P be even, since we want to colour the nodes on the outer circle with two colours. There are three different classes of graphs. Class A graphs have $P = 6t$, $t \geq 1$; these graphs have been described above. Class B graphs have $P = 6t+2$, $t \geq 1$. Class C graphs have $P = 6t+4$, $t \geq 1$. Class B and Class C graphs are similar in their properties to Class A graphs. In the Class B graph G(8,1) (see Fig 2), I_0 assigns the three colours a, b, and c to the nodes on the perimeter of G(8,1) in rotation as before. The last two nodes on the perimeter are assigned the colours a and b (or, alternatively, b and c, it does not matter which alternative is chosen). This does not introduce any colour conflicts. In Class C graphs (see Fig 3) the last node on the perimeter is assigned the colour b to avoid introducing colour conflicts.

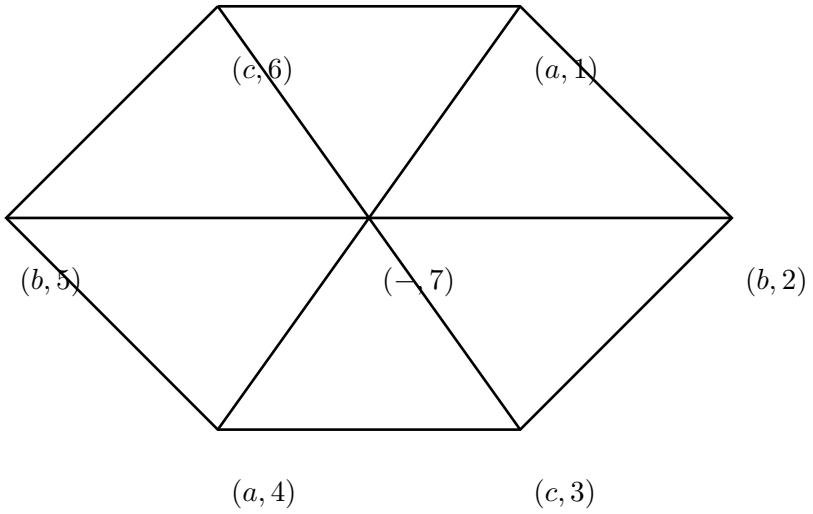

Figure 1: Class A Graph

4. Extended Formulation

Consider the graph G(6,1) with the initial assignment I_0 of colours. For convenience we name the boolean variables $a1$, $b1$, $c1$, $a2$, $b2$, $c2$, . . . , $c7$, where $a1$ is true if node 1 is assigned colour a, and so on. Initially, $a1$, $b2$, $c3$, $a4$, $b5$, $c6$ are true, and all other variables are false. There are no colour conflicts. However, the central node is not coloured. In this case the Extended Formulation has 21 variables and 64 clauses, of which 63 are initially satisfied and only one is not satisfied. For a variable v at any step in GSAT, let $make(v)$ represent the number of unsatisfied clauses that get satisfied when v alone is flipped and no other variable, and $break(v)$ the number of satisfied clauses that become unsatisfied when v alone is flipped. Let $gain(v) = make(v)$ - $break(v)$, and let $maxgain$ be the maximum among the gains of the variables. Let the graph $G(6t, r)$, $t \geq 1$, $r \geq 1$, be partially coloured initially with $s = r + 2$ colours by the hard initial colouring I_0, and let the Extended SAT Formulation be used. Also, let $minlen$ represent the minimum length of the tabu list that solves a problem. Then:

Theorem 4.1. *a) GSAT(0) cannot colour G(6t,1) properly for any $t \geq 1$;*
b) GSAT(1) can colour G(6t,1) properly for all $t \geq 1$, i.e., $minlen = 1$ in this case.

Proof: a) After the initial assignment I_0 of colours to the nodes of G(6,1), we find that $gain(a1)$ = -1, $gain(b1)$ = -2, and $gain(a7)$ = -1. Every variable has a gain of either -1 or -2, so $maxgain$ = -1. GSAT(0) therefore flips some variable v with $gain(v)$ = -1, which changes gain(v) to +1. The gains of some other variables, (like, $b1$, $c1$, if $v = a1$) get changed to 0, but no variable except v has a gain of +1, so at the next flip GSAT(0) again flips v, which restores the graph to its initial colouring. Thus $maxgain$ oscillates between -1 and +1 and GSAT(0) fails to find a solution. The number of unsatisfied clauses oscillates between 1 and 2. A similar proof holds for any $t > 1$.
b) GSAT(1) can colour the graph G(6,1) properly by means of the following sequence of nine flips: $b2$, $c2$, $c3$, $b3$, $b1$, $a1$, $a7$, $a4$, $c4$. The tabu list is initially empty. At each subsequent

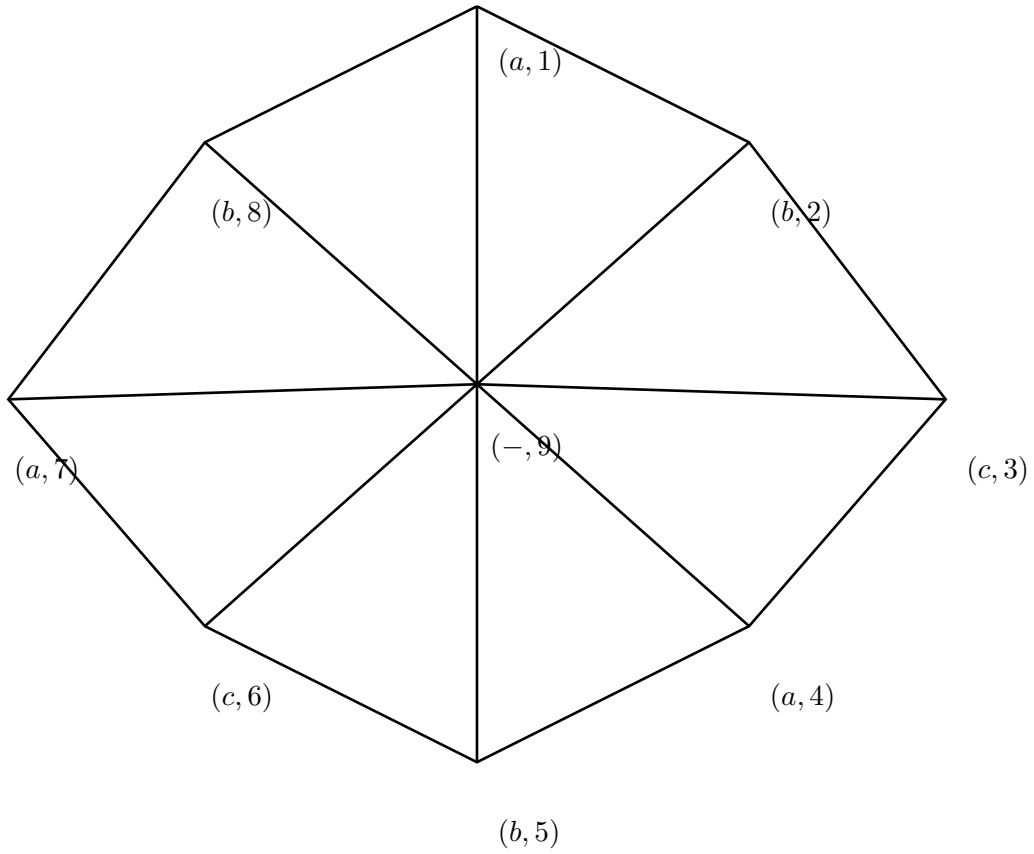

Figure 2: Class B Graph

step it contains the last variable that has been flipped, and a variable gets flipped only if it has the highest gain among all variables not in the tabu list. In this particular case, the tabu list is really needed only for the second flip, to prevent $b2$ from being flipped again. Similarly, GSAT(1) can colour G(12,1) properly by the following sequence of 17 flips: $b2$, $c2$, $c3$, $b3$, $b8$, $c8$, $c9$, $b9$, $b1$, $a1$, $c10$, $a10$, $b7$, $a7$, $a13$, $a4$, $c4$. The first four flips interchange the colours of nodes 2 and 3, and the next four the colours of nodes 8 and 9. Each interchange is initiated by flipping a true variable with gain = -1, and during each interchange, two variables with gain = 0 get flipped. The remaining nodes on the perimeter are then coloured b and c alternately by flipping false variables with gain = -1. The tabu list is really needed only at the second and sixth flips. In G($6t$,1), $t \geq 1$, the colours of the pairs of nodes (2,3), (8,9), (14,15), . . ., ($6t$-4, $6t$-3) are interchanged, then the remaining nodes are coloured appropriately, the total number of flips needed being at least ($8t$ + 1). □

Remark: All the *minlen* values given in this paper have been checked experimentally. In the experiments GSAT was run for only one try (corresponding to the hard initial colouring I_0) but for a large number of flips. All the *minlen* values given in this paper have been verified experimentally. In the experiments GSAT was run for only one try (corresponding to the hard initial colouring I_0) but for a large number of flips.

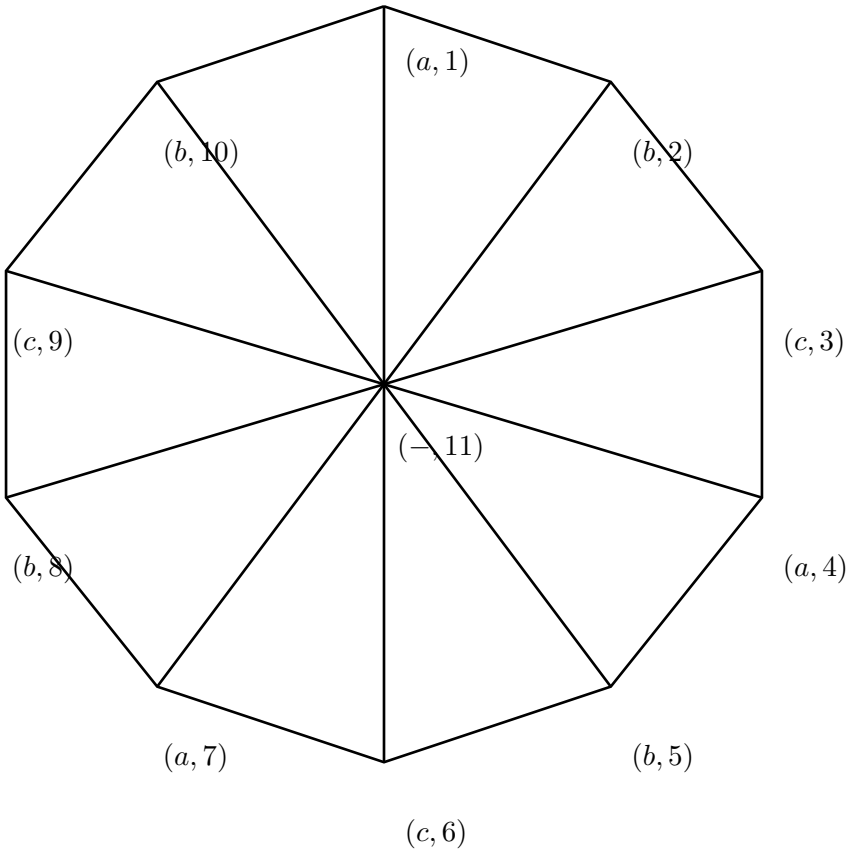

Figure 3: Class C Graph

Theorem 4.2. *a) GSAT(0) cannot colour G(6t,2) properly for any t ≥ 1;*
b) GSAT(1) can colour G(6t,2) properly for all t ≥ 1, i.e., minlen = 1 in this case.

Proof: a) In the graph G(6,2), let us suppose node 7 is coloured with d, and node 8 is uncoloured. Then $gain(d8) = 0$, and all other variables have negative gain, so GSAT(0) flips variable $d8$. This introduces a colour conflict between nodes 7 and 8, and $gain(d7)$ becomes 0 while $gain(d8)$ remains 0. No other variables have a *gain* of 0. Irrespective of whether GSAT(0) now flips $d7$ or $d8$, a replica of the initial state is reached, so no progress can be made. A similar proof holds for G($6t$,2), $t > 1$.
b) GSAT(1) can colour G(6,2) properly by the following sequence of 13 flips: $d8$, $d7$, $b2$, $c2$, $c3$, $b3$, $d7$, $d8$, $b1$, $a1$, $a8$, $a4$, $c4$. This scheme can again be generalized to larger graphs, a total of $\geq (12t+1)$ flips being needed for G($6t$,2). □

Theorem 4.3. *a) When r ≥ 3, GSAT(L), 0 ≥ L ≥ 4, cannot colour G(6t,r) properly for any t ≥ 1;*
b) When r ≥ 3, GSAT(5) can colour G(6t,r) properly for all t ≥1, i.e., minlen = 5 in this case.

Proof: a) Consider the graph G(6,3) coloured initially by I_0 with the five colours a, b, c, d, e. At start, only the variables $d9$ and $e9$ have a gain of 0 and all other variables have

negative gain. Suppose $L = 4$. (The proof is similar for $1 \geq L \geq 3$ and for $r > 3$.) It is easy to verify that at each instant during the execution of GSAT(4) there is always a variable with $gain = 0$ which is not in the tabu list. If we start by flipping $d9$, then the first six flips must be $d9$, $d7$, $e7$, $e8$, $d8$, $d9$. After these flips we return to a replica of the initial state. The situation is similar if we start by flipping $e9$. So no solution can be obtained. A similar proof holds for any $t > 1$.
b) In the execution of GSAT(5), $d9$ cannot be flipped at the sixth flip because it is in the tabu list. This allows us to flip a variable with a gain of -1 and yields a solution with the following sequence of 21 flips: $d9$, $d7$, $e7$, $e8$, $d8$, $b2$, $c2$, $c3$, $b3$, $d9$, $e9$, $e7$, $d7$, $d8$, $e8$, $b1$, $a1$, $e9$, $a9$, $a4$, $c4$. A similar sequence exists for $t \geq 2$, $r \geq 4$. At least $20t+1$ flips are needed for G($6t$,r), $r \geq 3$. □

Similar results hold for Class B and Class C graphs. In fact, we can show that:

1. When $r = 1$ or 2, GSAT(0) cannot colour G($6t$+2,r) or G($6t$+4,r) properly for any $t \geq 1$.

2. When $r \geq 3$, GSAT(L), $1 \geq L \geq 4$, cannot colour G($6t$+2,r) or G($6t$+4,r) properly for any $t \geq 1$.

3. When $r = 1$ or 2, GSAT(1) can colour both G($6t$+2,r) and G($6t$+4,r) properly for all $t \geq 1$.

4. When $r \geq 3$, GSAT(5) can colour both G($6t$+2,r) and G($6t$+4,r) properly for all $t \geq 1$.

To prove 1) and 2), proceed as in Theorems 4.1(a), 4.2(a) and 4.3(a). The sequences of flips needed for verifying 3) and 4) are given below.
Assuming that the nodes in the outer circle of G(8,1) are initially coloured in the sequence a, b, c, a, b, c, a, b, GSAT(1) can colour the graph properly by the following sequence of 9 flips: $a4$, $b4$, $b5$, $a5$, $a3$, $c3$, $c9$, $c6$, $b6$. Similarly, GSAT(1) colours G(10,1) properly by the following sequence of 11 flips: $a9$, $c9$, $a4$, $b4$, $b5$, $a5$, $b6$, $c6$, $c11$, $c3$, $a3$. These sequences eliminate colour c from the perimeter and achieve a proper colouring.

GSAT(1) can colour G(8,2) properly by flipping variables in the following sequence: $d10$, $d9$, $a4$, $b4$, $b5$, $a5$, $d9$, $d10$, $a3$, $c3$, $c10$, $c6$, $b6$. Similarly, GSAT(1) can colour G(10,2) properly by flipping variables in the following sequence: $d12$, $d11$, $a9$, $c9$, $d11$, $d12$, $a4$, $b4$, $b5$, $a5$, $d12$, $d11$, $a3$, $c3$, $c11$, $c6$, $b6$.

GSAT(5) can colour G(8,3) properly with the following sequence of flips: $d11$, $d9$, $e9$, $e10$, $d10$, $a4$, $b4$, $b5$, $a5$, $d11$, $e11$, $e9$, $d9$, $d10$, $e10$, $a3$, $c3$, $e11$, $c11$, $c6$, $b6$. Similarly, GSAT(5) can colour G(10,3) properly with the following sequence of flips: $d13$, $d11$, $e11$, $e12$, $d12$, $a9$, $c9$, $d13$, $e13$, $e11$, $d11$, $d12$, $e12$, $a4$, $b4$, $b5$, $a5$, $e13$, $d13$, $d11$, $e11$, $e12$, $d12$, $a3$, $c3$, $d13$, $c13$, $c6$, $b6$.
We can ask another interesting question about GSAT(L): Does there exist a positive integer $maxlen$ such that when $L > maxlen$, GSAT(L) is unable to colour G(P,r) properly? Clearly, if GSAT(L) is able to colour G(P,r) for some $L > 0$ without flipping any variable twice, then a variable that enters the tabu list is never selected again, implying that $maxlen$ is undefined in this case. But when a proper colouring cannot be achieved without flipping

Table 1: Minimum Length of Tabu List, Extended Formulation

Minimum Length of Tabu List				
Extended Formulation				
No	Class	t	r	*minlen*
1	A,B,C	≥ 1	1, 2	1
2	A,B,C	> 1	> 3	5

Table 2: Maximum Length of Tabu List, Extended Formulation

Maximum Length of Tabu List				
Extended Formulation				
No	Class	t	r	*maxlen*
1	A,B,C	≥ 1	1, 2	undefined
2	A,B,C	> 1	> 3	N- 2

some variable twice, *maxlen* has a finite value. The next theorem resolves the *maxlen* issue for Class A graphs. Identical results hold for Class B and Class C graphs (see Table 2).

Theorem 4.4. *a) When r = 1 or 2, GSAT(L) can colour G(6t,r) properly for all t ≥ 1, L ≥ 1. Consequently, maxlen cannot be defined in this case.*
b) When r ≥ 3, GSAT(L) can colour G(6t,r) properly for all t ≥1 when 5 ≤ L ≤ N-2. But when L ≥ N-1, GSAT(L) is unable to colour G(6t,r) properly for any t ≥ 1. Consequently, maxlen = N-2 in this case.

Proof: a) For $L \geq 1$, GSAT(L) can colour G(6t,1), $t \geq 1$, properly without flipping any variable twice, as shown in the proof of Theorem 4.1(b), so *maxlen* cannot be defined in this case. For $1 \leq L \leq 4$, it can be readily checked that GSAT(L) can colour G(6,2) properly. For $L \geq 5$, GSAT(L) can colour G(6,2) properly with the flips $d8$, $d7$, $b2$, $c2$, $c3$, $b3$, $b1$, $a1$, $a7$, $a4$, $c4$, which has no repetitions of variables. It follows that *maxlen* cannot be defined here either. Similar arguments can be given for $t > 1$.
b) When $r \geq 3$ there is no way to colour G(6t,r) without flipping some variable twice. To see this, consider the graph G(6,3). For $L \geq 1$, GSAT(L) will start by flipping variables in the sequence $e9$, $e8$, $d8$, $d7$, $e7$, ... (or in the sequence $d9$, $d7$, $e7$, $e8$, $d8$, ...). Since all three nodes at the centre have been coloured and there is a colour conflict, GSAT(L) will have to flip one of these variables later to get rid of the conflict. Since there are a total of N variables, it follows that *maxlen* $< N$ in this case.
Now suppose $L = N$-1. Then GSAT will be forced to flip every variable once in the first complete cycle. After the second complete cycle, the initial colouring will be restored. This process will repeat indefinitely, so a proper colouring cannot be achieved. This problem does not arise for $5 \leq L \leq N$-2, as can be experimentally verified. The sequence that achieves the proper colouring can be very long when L gets close to N-2. □

5. Basic Formulation

In the Basic Formulation, Π contains no clauses to enforce the constraint that a node cannot be multiply coloured. Thus a node can have more than one colour, but this does not lead to improper colourings. The values of *minlen* in few case are determined theoretically, the rest are determined experimentally.

Theorem 5.1. *a) GSAT(0) cannot colour G(6,1) properly;*
b) GSAT(1) can colour G(6,1) properly, i.e., minlen = 1 in this case.

Proof: a) Similar to the proof of Theorem 4.1(a).
b) GSAT(1) solves the problem as follows. The variable v that is flipped at the first step enters the tabu list, and is barred from selection at the next step. Suppose b2 is flipped at the first step. This makes $b2$ false and $gain(b2)$ changes to +1. At the same time, the gains of $a2$, $b1$, $b3$, $b7$ and $c2$ change to 0. Since $b2$ enters the tabu list, one of these five variables must now be flipped. Clearly our aim should be to eliminate one of the three colours from the perimeter so that it can be assigned to the central node. The sequence of flips $b2$, $c2$, $c3$, $b3$, $b1$, $a1$, $c4$, $a4$, $a7$ accomplishes this objective. There are instants during execution when a node has more than one colour assigned to it. Note that the tabu list is really needed only at the second step and L can be reset to 0 after the second flip. □

Theorem 5.2. *a) GSAT(0) cannot color G(6t,2) properly for any t ≥ 1;*
b) GSAT(1) can color G(6t,2) properly for all t ≥ 1, i.e., minlen = 1 in this case.

Proof: a) Similar to that of Theorem 4.2(a).
b) This proof is quite different from that of Theorem 4.2(b) because the distribution of the gains of the variables is different for the two formulations. GSAT(1) can colour G(6,2) properly by flipping variables in the following sequence: $d8$, $d7$, $d1$, $d8$, $d3$, $d5$, $a1$, $c3$, $b5$, $b4$, $b6$, $a4$, $a7$, $c6$, $c8$. Here we find that the colour d, which is the initial colour of node 7, has been used to colour nodes 1, 3, 5 in the periphery. This scheme can again be generalized to larger graphs. □

Theorem 5.3. *a) When r ≥ 3, GSAT(L), 0 ≥ L ≥ 4, cannot color G(6t,r) properly for any t ≥ 1;*
b) When r ≥ 3, GSAT(5) can color G(6t,r) properly for all t ≥ 1, i.e., minlen = 5 in this case.

Proof: a) Similar to the proof of Theorem 4.3(a).
b) In the execution of GSAT(5), for r =3 and $t = 1$, the following sequence of flips leads to a solution: $d9$, $d7$, $e7$, $e8$, $d8$, $e1$, $d9$, $a1$, $e7$, $e3$, $e5$, $c3$, $b5$, $b4$, $b6$, $a4$, $a7$, $c6$, $c9$. Sequences of the same general form can be found for $r > 3$ and $t > 1$. □

The results are shown in Table 3. Class B graphs are of great interest because the value of *minlen* can be high when the Basic SAT Formulation is used. To see why this is so, let us look at the graph G(8,1). Initially, only the variables $a1$, $b2$, $c3$, $a4$, $b5$, $c6$, $a7$ and $b8$ are true. Here $maxgain = 0$, and the only variables with $gain = 0$ are $c1$ and $c8$. When we flip either of these two variables, some other variables attain a gain of zero. A variable with gain of zero is always available outside the tabu list when $L < 17$. When $L = 17$ it is possible to flip a variable of $gain$ = -1, and the process of elimination of a colour from the perimeter can begin

Table 3: Minimum Length of Tabu List, GSAT(L): Basic SAT Formulation

No	Class	P	r	s	*minlen*
1	A	6	1	3	1
2	A	6t, t $>$ 1	1	3	2
3	A	6t, t $\geq$ 1	2	4	1
4	A	6t, t $\geq$ 1	$>$ 2	r + 2	5
5	B	6t+2, t $\geq$ 1	1	3	2P+1
6	B	6t+2, t $\geq$ 1	2	4	2P+2
7	B	6t+2, t $\geq$ 1	$>$ 2	r + 2	2P+6
8	C	6t+4, t $\geq$ 1	1	3	2
9	C	6t+4, t $\geq$ 1	2	4	3
10	C	6t+4, t $\geq$ 1	$>$ 2	r + 2	7

Similar arguments apply to other Class B graphs. A Class C graph like G(10,1) has two variables with zero gain at start, but there still exists a sequence of flips that selects a variable with $gain$ = -1 when $L = 2$.

6. Experimental Results

In this section we study experimentally the variation of runtime with the length of the tabu list to verify and further explore the theoretical results of the previous section. For each problem, just as there is a *minlen* there is also a best length *bestlen* at which the performance of GSAT(L) is optimal, i.e., the runtime is minimum. One of the objectives of this section is to determine *bestlen* experimentally and to find out where *bestlen* lies in the range of length L. Another issue to be examined is the existence of *maxlen*, which is the maximum length of the tabu list for which GSAT(L) outputs a solution. It is also interesting to examine the general performance of other local search SAT methods like WalkSAT, Novelty, RNovelty, Novelty+, RNovelty+ on these classes of graphs and to make a comparative assessment of all these methods. It would be nice to know whether the general conclusions reached in [1], on the basis of the experimental results on benchmark problems, carry over to the graphs in Classes A, B and C.
In the experiments reported here the values of $maxtries$ and $maxflips$ were kept fixed at 100 and 100,000 respectively. One hundred runs were taken on each graph. Note that a try here does not imply a different random initial configuration, since the initial conditions are the same. In fact, instead of 100 random initial settings we are just making 100 fresh attempts at solving the same problem. Owing to the randomness inherent in the solution method itself, the program proceeds along a different path in each case. Since there are 100 runs and 100 tries, the algorithm tries to colour a particular graph 100x100 times with the same initial configuration. Some attempts succeed in reaching a solution and some others do not. All the runs were taken on a 650 MHz Pentium III PC with 128 MB of RAM.

6.1 Variation of Runtime with L

We now try to check experimentally the minimum lengths that were derived, and to study the overall variation of runtime with the length of the tabu list. Runs were taken on graphs of different sizes belonging to all the three classes.

In Fig 4 the runtime of GSAT has been plotted against the length L of the tabu list for the Class A graph G(12,1), for which $s = 3$. The problem was encoded in SAT using the Basic Formulation. Here, $minlen = 2$. We find as expected that the problem cannot be solved when $0 \leq L < 2$ for the given values of $maxtries$ and $maxflips$, which supports the theoretical prediction. Here we find that $bestlen = 3$. As we further increase the length, the runtime increases, and after going through ups and downs it jumps to a very large value at $L = 30$. Beyond $L = 30$ we were unable to solve the problem. This suggests the existence of a maximum length $maxlen$ beyond which GSAT cannot colour the graph. When plotting the runtime against the length L we took the length L_m as n-1, where n is the total number of variables in the SAT formulation, as the upper limit. There is no point in allowing L_m to exceed n-1, since, if all the variables are in the tabu list and a variable is required to flip more than once to get the solution the program will get stuck. For convenience of plotting we have often discarded the runtime data for very large values of L, as this would extend the range of runtime too much resulting in poor resolution in the plots. But all the runtime data is shown in the tables. For all the Class A graphs coded in SAT using the Basic Formulation and coloured initially using the hard initial colouring, the nature of the plots are more or less the same (see Figs 4 and 6). There are two significant runtime minima, one near $L = minlen$ and another near the upper limit $L = L_m$; these occur approximately at $L = 0.08L_m$ and $L = 0.8L_m$. There are one or more minor minima. In Fig 4, $bestlen = 3 = 0.08L_m$, and in Fig 6, $bestlen = 40 = 0.8L_m$. It can be said that in general, for Class A and Class C graphs coded using the Basic Formulation, GSAT(L) exhibits two significant minima (see Figs 4, 6, 10, 12, 13), one at a low value of L and another at a high value of L. The $minlen$ for Class B graphs using the Basic Formulation is $2P+1$, where P is the number of peripheral nodes. Here the range of values of L is small, and two significant minima in runtime do not occur (Fig 8). But under random initial colouring, GSAT yields plots with two significant minima for all classes of graphs (Figs 16, 18, 20). For the Extended Formulation, the performance of GSAT is more or less similar for all three classes of graphs and all initial colourings. Here, unlike for the Basic Formulation, only one significant minimum is found, and this always occurs close to $minlen$. Therefore $bestlen$ is slightly greater than $minlen$ in this case (Figs 5, 7, 9, 11, 15, 17, 19, 21).

6.2 Poor Performance of GSAT

An important observation in all these experiments is that the performance of GSAT is poor even when the tabu list is used. Changing the length of the tabu list or taking random initial colouring does not improve its performance significantly. This is very surprising because in case of the benchmark problems, *i.e.*, Culberson's Flat Graphs and Johnson's Random Graphs, GSAT gives very promising results [1]. Thus in this regard, local search methods exhibit unreliability and inconsistency in performance. Tables 4 and 5 show the limitations of GSAT when it is applied to colour graphs from the different classes. Upto size $t = 4$, it can colour Class A graphs encoded in SAT for both the Basic and the Extended Formulations.

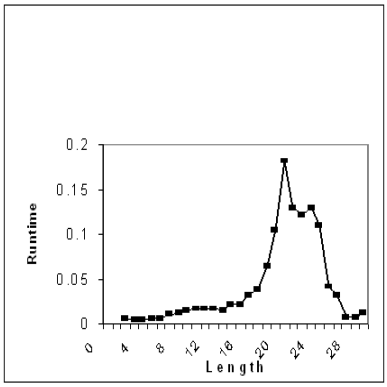

Figure 4: Class A, G(12,1), Basic Formulation

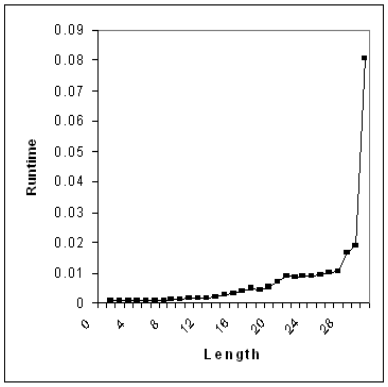

Figure 5: Class A, G(12,1), Extended Formulation

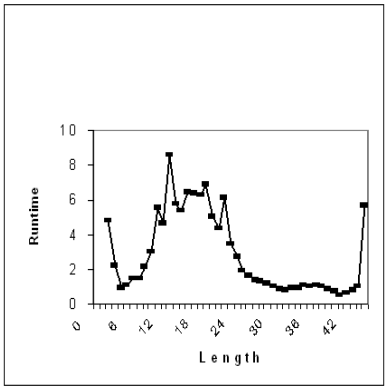

Figure 6: Class A, G(18,1), Basic Formulation

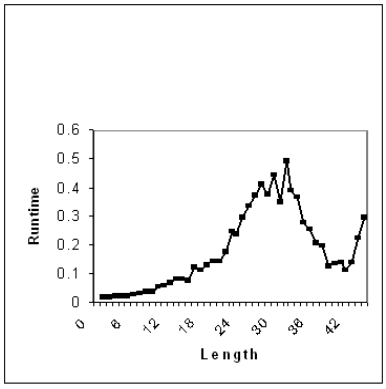

Figure 7: Class A, G(18,1), Extended Formulation

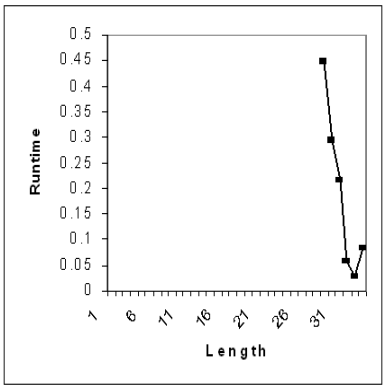

Figure 8: Class B, G(14,1), Basic Formulation

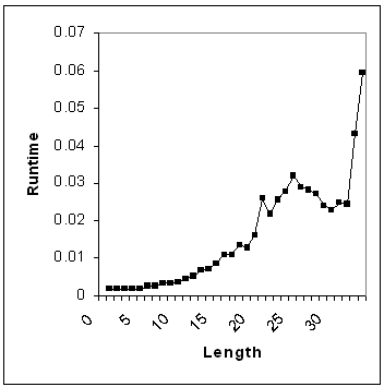

Figure 9: Class B, G(14,1), Extended Formulation

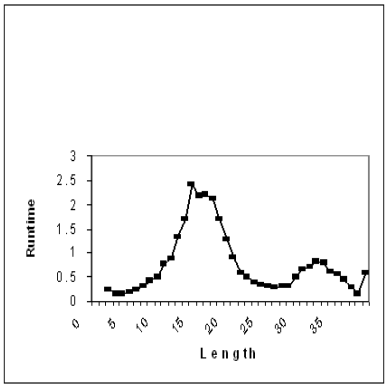

Figure 10: Class C, G(16,1), Basic Formulation

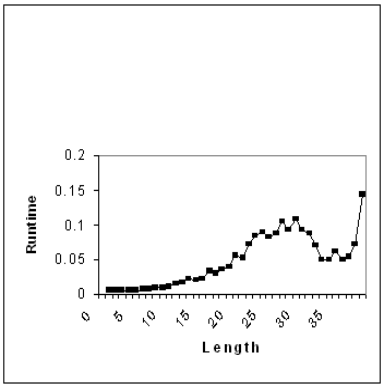

Figure 11: Class C, G(16,1), Extended Formulation

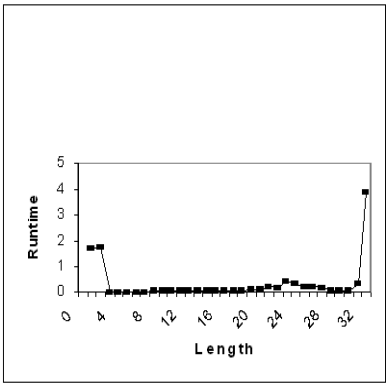

Figure 12: Class A, G(12,2), Basic Formulation

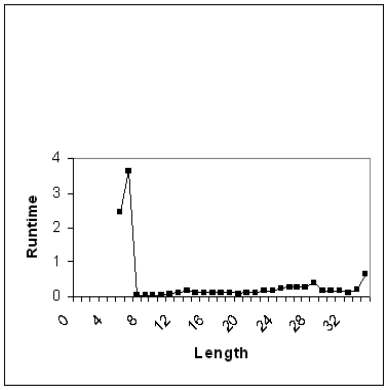

Figure 13: Class A, G(12,3), Basic Formulation

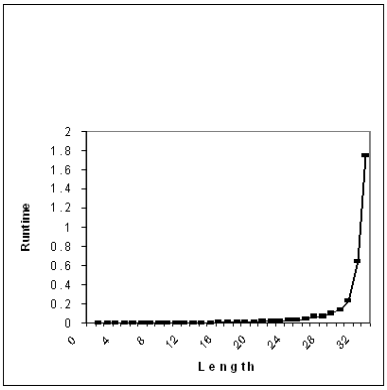

Figure 14: Class A, G(12,2), Extended Formulation

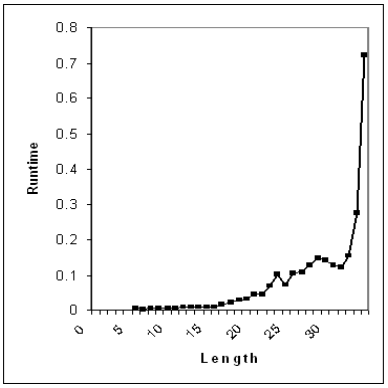

Figure 15: Class A, G(12,3), Extended Formulation

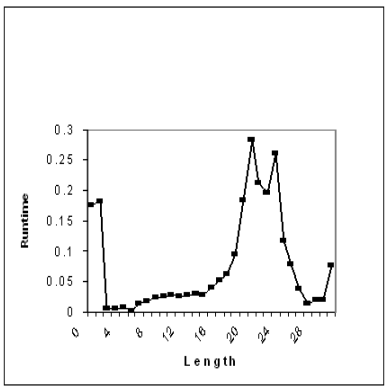

Figure 16: Class A, G(12,1), Basic Formulation

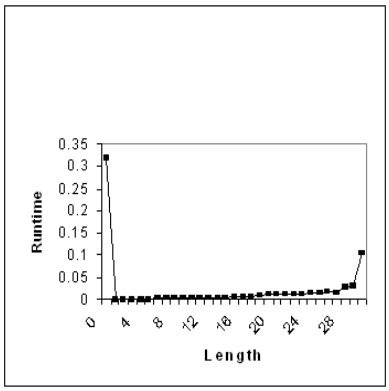

Figure 17: Class A, G(12,1), Extended Formulation

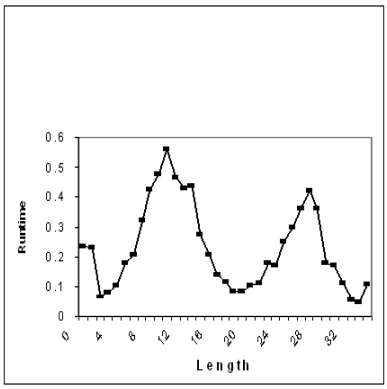

Figure 18: Class B, G(14,1), Basic Formulation

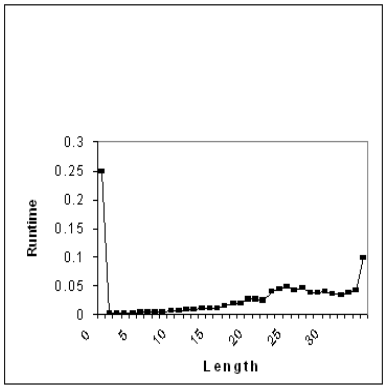

Figure 19: Class B, G(14,1), Extended Formulation

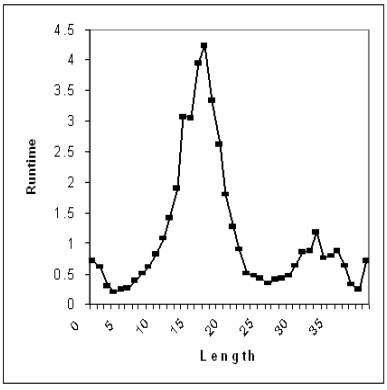

Figure 20: Class C, G(16,1), Basic Formulation

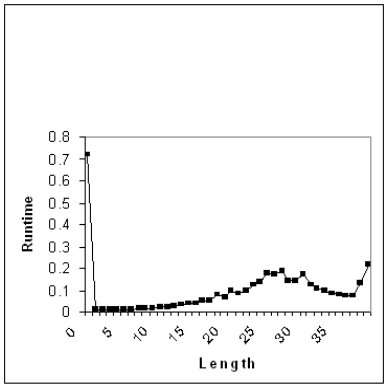

Figure 21: Class C, G(16,1), Extended Formulation

Table 4: Performance of GSAT, Hard Initial Colouring

Class	Nodes	Colours	Time		Remarks
			Basic	Extended	
A	25	3	198.27	1.31	LB = 45 LE = 1
	26	4	-	20.36	L =8
	27	5	-	51.75	L= 8
	31	3		51.55	L = 1
B	27	3	-	5.89	L = 3
	28	4	-	62.16	L= 8
C	29	3	-	32.13	L = 10
	30	4	-	87.05	L = 10

Table 5: Performance of GSAT, Random Initial Colouring

Class	Nodes	Colours	Time		Remarks
			Basic	Extended	
A	25	3	115.37	1.38	LB = 45 LE = 1
	26	4	-	21.85	L = 8
	27	5	-	55.59	L= 8
	31	3		53.09	L = 1
B	27	3	-	5.27	L = 3
	28	4	-	52.74	L= 8
C	29	3	-	30.30	L = 10
	30	4	-	69.48	L = 10

In the Basic Formulation, the problems become hard and the number of successful runs few for Class A graphs when $t = 4$, i.e., for nodes = 25 and $s = 3$. For $t = 5$ and $s = 3$, GSAT can solve Class A problem instances only in the Extended Formulation. For $t > 5$, the instances become very hard to solve. For $t > 4$, GSAT finds it difficult to solve Class B and C instances in either formulation, even when the initial colouring is random. When $t = 4$, the experimental results show that GSAT can solve the problem instances only using the Extended Formulation. It is in general true that Class B and C graphs are harder to colour using GSAT than Class A graphs. In the tables we have used LB and LE to denote the lengths of the tabu list for the Basic and the Extended Formulations when both are used for solving the same instance. We use L to denote the length of the tabu list when only one formulation is used.

6.3 Comparative Assessment of Different Methods

As the performance of GSAT is very poor on these classes of graphs, it was not considered in the comparative assessment. We compared the performance of the other six methods: USAT, WalkSAT, Novelty, RNovelty, Novelty+ and RNovelty+, choosing graphs of appropriate sizes so that the colouring can be achieved in reasonable time. Since the level of difficulty of the problem depends on the chromatic number s of the graph, we had to take

graphs of different sizes for different values of s. Three sizes were chosen: $t = 40$, 8, 2, corresponding to chromatic number $s = 3$, 4, 5 respectively. For graphs of Class A with $s = 1$, 2, 3, the number of nodes is given by $6t + 1$, $6t + 2$ and $6t + 3$ respectively. So when $t = 40$ the number of nodes is 241 for a Class A graph with $s = 3$. We then observed the average runtime needed to solve the problem instances by the different methods. We allowed 100 fresh runs, in each run 100 tries, and in each try 100,000 flips. For methods like WalkSAT, Novelty and RNovelty the suitable value for probability p was selected on the basis of exploratory experiments. For Novelty+ and RNovelty+ the probability of random walk wp was selected as 0.01 on the basis of exploratory experiments.

We can make the following comments on the performance of different methods on the basis of the experimental data presented in Tables 6 to 11.

1. Novelty+ and RNovelty+ are, in general, the best methods for colouring graphs of all classes. They do not fail to solve any instance of any size of graphs and their performances are almost comparable to each other.

2. Novelty performs well in general for colouring graphs belonging to the three classes when $s = 3$. When $s > 3$ and hard initial colouring is used, Novelty fails to solve any instance of any size of graphs of Classes A and C. This is because Novelty is greedy in selecting variables for flipping, and there is no provision for selecting a variable randomly. Only one of two variables of the randomly chosen unsatisfied clause gets selected, either the best or the second best, as determined by their gain. Even when the initial colouring is random, we observe that it takes a long time to colour the graphs when $s > 3$.

3. The average performance of RNovelty is better than Novelty. It can solve all the instances from all classes of graphs in a moderate amount of time. Though RNovelty is greedier than Novelty in selecting variables for flipping, still it does not get stuck in a local minimum because after every 100 flips it selects a variable randomly.

4. WalkSAT also can solve all the instances but takes more time than RNovelty because the probability p of selecting a variable randomly is non-zero.

5. USAT is slow compared to other methods for $s = 3$. It should be recalled, however, that on DSJ graphs and Culberson's graphs, USAT outperforms both Novelty and RNovelty

Table 6: Comparative Assessment of Different Methods, Class A Graphs, Hard Initial Colouring

Nodes	Colours	Method	Time		Remarks
			Basic	Extended	
241	3	USAT	25.73	27.42	L = 700
		WalkSAT	16.27	18.85	L =2,p=20
		Novelty	0.20	0.20	P = 90
		Novelty+	1.58	1.77	P = 90
		Rnovelty	1.05	1.08	P = 90
		Rnovelty+	0.95	0.94	P = 90
50	4	USAT	0.018	0.020	L = 150
		WalkSAT	9.14	9.02	L= 2, p = 20
		Novelty	-	-	P = 90
		Novelty+	0.095	0.126	P = 90
		Rnovelty	0.16	0.17	P = 90
		Rnovelty+	0.46	0.52	P = 90
15	5	USAT	-	-	L = 50
		WalkSAT	0.0067	0.0102	L = 2, p = 20
		Novelty	-	-	P = 90
		Novelty+	0.014	0.017	P = 90
		RNovelty	0.012	0.013	P = 90
		Rnovelty+	0.032	0.031	P = 90

Table 7: Comparative Assessment of Different Methods, Class A Graphs, Random Initial Colouring

Nodes	Colours	Method	Time		Remarks
			Basic	Extended	
241	3	USAT	24.87	23.93	L = 700
		WalkSAT	15.51	18.65	L = 2, p = 20
		Novelty	0.26	0.24	P = 90
		Novelty+	6.95	6.22	P = 90
		Rnovelty	1.21	1.28	P = 90
		Rnovelty+	1.26	1.11	P = 90
50	4	USAT	0.022	0.023	L = 150
		WalkSAT	7.85	9.96	L= 2, p = 20
		Novelty	2.48	2.05	P = 90
		Novelty+	0.117	0.095	P = 90
		Rnovelty	0.34	0.28	P = 90
		Rnovelty+	0.608	0.473	P = 90
15	5	USAT	24.16	14.84	L = 50
		WalkSAT	0.0047	0.0084	L = 2, p = 20
		Novelty	18.06	13.39	P = 90
		Novelty+	0.018	0.018	P = 90
		Rnovelty	3.28	1.98	P = 90
		Rnovelty+	0.042	0.049	P = 90

Table 8: Comparative Assessment of Different Methods, Class A Graphs, Hard Initial Colouring

Nodes	Colours	Method	Time		Remarks
			Basic	Extended	
243	3	USAT	28.78	31.01	L = 700
		WalkSAT	14.22	19.91	L = 2, p = 20
		Novelty	0.22	0.22	P = 90
		Novelty+	1.78	1.712	P = 90
		Rnovelty	1.003	1.033	P = 90
		Rnovelty+	1.037	1.176	P = 90
52	4	USAT	0.022	0.021	L = 150
		WalkSAT	10.45	14.52	L= 2, p = 20
		Novelty	0.005	0.006	P = 90
		Novelty+	0.025	0.035	P = 90
		Rnovelty	0.095	0.10	P = 90
		Rnovelty+	0.113	0.142	P = 90
17	5	USAT	-	-	L = 75
		WalkSAT	0.012	0.014	L = 2, p = 20
		Novelty	-	-	P = 90
		Novelty+	0.032	0.035	P = 90
		Rnovelty	0.015	0.016	P = 90
		Rnovelty+	0.044	0.037	P = 90

Table 9: Comparative Assessment of Different Methods, Class A Graphs, Random Initial Colouring

Nodes	Colours	Method	Time		Remarks
			Basic	Extended	
243	3	USAT	25.22	23.42	L = 700
		WalkSAT	17.38	23.34	L = 2, p = 20
		Novelty	0.27	0.29	P = 90
		Novelty+	1.65	1.57	P = 90
		RNovelty	1.25	1.24	P = 90
		Rnovelty+	1.25	1.199	P = 90
52	4	USAT	0.03	0.02	L = 150
		WalkSAT	11.61	14.73	L= 2, p = 20
		Novelty	3.52	1.97	P = 90
		Novelty+	0.174	0.168	P = 90
		Rnovelty	0.47	0.34	P = 90
		Rnovelty+	0.526	0.55	P = 90
17	5	USAT	30.89	28.70	L = 75
		WalkSAT	0.010	0.013	L = 2, p = 20
		Novelty	25.44	25.81	P = 90
		Novelty+	0.047	0.044	P = 90
		Rnovelty	3.96	2.99	P = 90
		Rnovelty+	0.054	0.062	P = 90

Table 10: Comparative Assessment of Different Methods, Class A Graphs, Hard Initial Colouring

Nodes	Colours	Method	Time		Remarks
			Basic	Extended	
245	3	USAT	24.36	27.56	L = 700
		WalkSAT	15.56	20.87	L = 2, p = 20
		Novelty	0.19	0.20	P = 90
		Novelty+	1.715	1.99	P = 90
		Rnovelty	1.06	1.09	P = 90
		Rnovelty+	0.94	1.156	P = 90
54	4	USAT	0.022	0.027	L = 150
		WalkSAT	18.15	22.23	L= 2, p = 20
		Novelty	-	-	P = 90
		Novelty+	0.205	0.236	P = 90
		Rnovelty	0.33	0.35	P = 90
		Rnovelty+	0.59	0.477	P = 90
19	5	USAT	-	-	L = 80
		WalkSAT	0.020	0.026	L = 2, p = 20
		Novelty	-	-	P = 90
		Novelty+	0.04	0.045	P = 90
		Rnovelty	0.019	0.020	P = 90
		Rnovelty+	0.06	0.066	P = 90

Table 11: Comparative Assessment of Different Methods, Class A Graphs, Random Initial Colouring

Nodes	Colours	Method	Time		Remarks
			Basic	Extended	
245	3	USAT	24.36	27.56	L = 700
		WalkSAT	17.40	25.52	L = 2, p = 20
		Novelty	0.294	0.290	P = 90
		Novelty+	1.95	2.20	P = 90
		Rnovelty	1.295	1.288	P = 90
		Rnovelty+	1.22	1.075	P = 90
54	4	USAT	0.022	0.027	L = 150
		WalkSAT	16.52	25.58	L= 2, p = 20
		Novelty	3.06	1.67	P = 90
		Novelty+	0.18	0.19	P = 90
		RNovelty	0.48	0.45	P = 90
		Rnovelty+	0.534	0.43	P = 90
19	5	USAT	31.03	36.47	L = 80
		WalkSAT	0.021	0.027	L = 2, p = 20
		Novelty	35.94	30.67	P = 90
		Novelty+	0.055	0.055	P = 90
		RNovelty	4.85	3.19	P = 90
		Rnovelty+	0.075	0.083	P = 90

7. Conclusion

In this paper we have given examples of certain interesting classes of graphs. These graphs, when partially coloured at start in a particular way, cannot be coloured properly by GSAT without the help of a tabu list. The length of the tabu list has a non-zero lower bound that depends on the class of the graph and the number of nodes in it. The length also depends on whether the Basic Formulation or the Extended Formulation is used in constructing the SAT clauses. In some cases, the lower bound on the length of the tabu list is quite large.
The experimental observations regarding the variation of runtime with the length of the tabu list generate some issues that cannot be explained satisfactorily yet. A proper colouring of G($6t$,r), $t > 1$, by GSAT(L) in either SAT formulation cannot be achieved sometimes for arbitrarily large values of L. How can we determine the exact value of *maxlen*? When we colour Class A and C graphs by GSAT(L) using the Basic Formulation, the plot of runtime against L is found to have two significant local minima, one pretty close to $0.08*n$ and the other pretty close to $0.8*n$, n = number of nodes. The value of *bestlen* corresponds to one of them depending on the problem instance. There are other less significant local minima. Can we determine *bestlen* theoretically? Can we give a theoretical justification for the observed nature of the plot?
For the Extended Formulation, the runtime of GSAT(L) tends to increase monotonically with L after a short oscillatory phase near *minlen*. We have not yet been able to explain why the curves have these particular shapes.
In the context of the poor performance of GSAT(L) on Class A, B, and C graphs, and in contrast its excellent performance[1] on DSJ graphs and Culberson's graphs, the following question can naturally be asked. Why does GSAT perform so poorly on graphs of Classes A, B, and C, even when a tabu list is used? By modifying GSAT slightly, can we make it perform at least as well as Novelty or RNovelty? Are there any harder initial colourings for the graphs than the ones considered here? It is conceivable that for some other hard initial colourings, *minlen* is even larger than what has been reported here. As the graphs get larger, it becomes more and more difficult for GSAT(L) to colour the graphs properly, whatever be the value of L, even when the initial colouring is random. These problems would be of great interest in practice because the chromatic number of each graph is known in advance.

References

[1] S. Acharyya, SAT Methods for Colouring Graphs: A Comparative Assessment. *Proceedings of National Conference on Affordable Computing, organized by Computer Society of India(CSI-06)* (2006), 255–259.

[2] S. Acharyya, A Bagchi, Local Search Methods for Coloring Graphs: Role of Tabu List. *Fifth International Conference on Knowledge Based Computer Systems(KBCS-2004)* (2004), 452–467.

[3] S. Acharyya, The Satisfiability Problem: A Constraint Satisfaction Approach. Ph D Thesis, Computer Science & Engg., Calcutta University, 2001.

[4] M. Chiarandini, I. Dumitrescu, T. Stutzle, Stochastic Local Search Algorithms for the Graph Colouring Problem. *Forschungsbericht AIDA-05* **03** (2005), 1–16.

[5] A .S. Fukunaga, Variable Selection Heuristics in Local Search for SAT. *Proc AAAI 97* (1997), 275–280.

[6] Garey & Johnson, Computers & Intractibility: A Guide to the Theory of NP-Completeness. W H Freeman & Co, (1979).

[7] F. Harary, Graph Theory. Addison Wesley, (1969).

[8] H. H. Hoos, On the Run-time Behavior of Stochastic Local Search Algorithms for SAT. *Proc AAAI 99* (1999), 661–666.

[9] D. S. Johnson, C R Aragon, L A McGeoch & C Schevon, Optimization by Simulated Annealing: An Experimental Evaluation, Part II, Graph Colouring and Number Partitioning. *Operations Research* **39** (1991), 378–406.

[10] D. MacAllester, B. Selman & H. Kautz, Evidence for Invariants in Local Search. *Proc AAAI 97* (1997), 321–326.

[11] B. Mazure, L. Sais & E. Gregoire, Tabu Search For SAT. *Proc AAAI 97* (1997), 281–285.

[12] B. Selman, H. Kautz & B. Cohen, Noise Strategies for Improving Local Search. *Proc AAAI 94* (1994), 337–343.

[13] B Selman, H J Levesque & D G Mitchell, A New Method for Solving Hard Satisfiability Problems. *Proc AAAI 92* (1992), 440–446.