A PARTICLE SWARM ALGORITHM FOR SOLVING THE MAXIMUM SATISFIABILITY PROBLEM

Abdesslem Layeb

MISC Lab., Computer science department, University mentouri of Constantine, Algeria
Layeb.univ@gmail.com

ABSTRACT

The Maximum Satisfiability problem (Max-SAT) is one of the most known variants of satisfiability problem. The objective is to find the best assignment for a set of Boolean variables that gives the maximum of verified clauses in a Boolean formula. Unfortunately, this problem was showed NP-complete if the number of variable per clause is higher than 3. In this paper we investigate the use of particle swarm optimization principles to resolve the. The underlying idea is to harness the optimization capabilities of PSO algorithm to achieve good quality solutions for Max-SAT problem. To foster the process, a local search has been used. The second great feature of our approach is the use of an adaptive objective function based on the clause weights. The obtained results are very encouraging and show the feasibility and effectiveness of the proposed hybrid approach.

Keywords: *Maximum Satisfiabilty problem, Particle Swarm Optimization, Local Search.*

1. INTRODUCTION

The combinatorial optimization plays a very important role in operational research, discrete mathematics and computer science. The aim of this field is to solve several combinatorial optimization problems that are difficult to solve. The propositional satisfiability problem (SAT) is among the well-known combinatorial optimization problems in the literature. It is the task of determining the satisfiability of a given Boolean formula by looking for the variable assignment that makes this formula evaluating to true [1]. The SAT problem is very important in many fields. In fact, many problems are easily representable in propositional logic areas such as model checking [2], graph coloring [3] and task planning [4] to cite just few. In 1971, Stephen Cook [1] had demonstrated that the SAT problem is NP-complete.

In some cases, we want to know how many clauses are true. This is another instance of the Sat problems called MAX-SAT (MAXimum SATisfiability). In recent years, there is growing interest in this problem, given its use in a wide range of domains. For a propositional formula in CNF, the problem is to determine the maximum number of clauses of this formula that can be fulfilled. A variant of this problem is the problem MAX k-SAT [5]. This problem relates to instances of clauses with at most k literals. Max 3-SAT is special case of Max k-SAT problems where the Boolean expressions are written in CNF (Conjunctive Normal Form) with 3 variables per clause.

The problem MAX k-SAT has been shown to be NP-complete for any k> 2 [6]. There are other variants of the Max-SAT problem such as Weighted Max-SAT [7] and Partial Max-SAT [8].

To solve the Max-SAT problem, many algorithms were proposed. In fact, there are two classes of algorithms for solving instances of Max-SAT in practice: exact and heuristics methods. The exact algorithms are able to find the exact solution of the Max-SAT problem, although they have an exponential complexity [6]. The most popular algorithms of this class are based on the Davis-Putnam-Loveland algorithm (DPLL) [9]; for example, a branch and bound algorithm based on DPLL is one of the most competitive exact algorithms for Max-SAT [10]. On the other hand, the metaheuristics methods find an optimal solution in good time, but they don't guarantee to give the exact solution of the problem. This class of methods contains Evolutionary Algorithms (EA) [11, 12, 13], Stochastic Local Search (SLS) methods [14, 15, 16, 17] and hybrid methods of EA and SLS or Exact methods [18, 19, 20].

Evolutionary computation has been proven to be an effective way to solve complex engineering problems. It presents many interesting features such as adaptation, emergence and learning. Artificial neural networks, genetic algorithms and swarm intelligence are examples of bio-inspired systems used to this end. In recent years, optimizing by swarm intelligence [21] has become a research interest to many research scientists of evolutionary computation fields. There are many algorithms based swarm intelligence like Ant Colony optimization, eco-systems optimization, etc. The main algorithm for swarm intelligence is Particle Swarm Optimization (PSO) [22-23], which is inspired by the paradigm of birds grouping. PSO was used successfully in various hard optimization problems. The simplicity of implementation and use are the main features of PSO compared to other evolutionary computing algorithms. In fact, the updating mechanism in the algorithm relies only on two simple PSO self-updating equations, so CPU-time cost of one generation in PSO algorithm is less than reproduction mechanism using mutation or crossover operations in typical EA. However, the use of

a pure PSO algorithm for satisfiability problems is not good enough. To overcome this weakness, the integration of the Stochastic Local Search (SLS) is required in order to increase the performances of the evolutionary algorithms to deal with the SAT problems.

The present study was designed to investigate the use of PSO algorithm hybridized with local search to deal with the maximum satisfiability problem. The proposed approach is a population based method where every individual represents a potential solution to the problem at hand. The features of the proposed method consist in applying different PSO principles to govern the dynamics of the population in order to optimize an adaptive objective function based on the clause weights. To foster the convergence to optimality, a local search based has been embodied within the optimization process.

The remainder of the paper is organized as follows. In section 2, a formulation of the tackled problem is given. Section 3 presents a state of art on the Max-SAT solvers. Some basic concepts of particle swarm optimization are presented in section 4. In section 5, the proposed method is described. Experimental results are discussed in section 6. Finally, conclusions and future work are drawn.

2. PROBLEM FORMULATION

Throughout this paper, *n* will represent the number of Boolean variables and *m* will denote the number of clauses in formula F. Given a Boolean formula F expressed in CNF and having *n* Boolean variables x_1, x_2... x_n, and *m* clauses. The Max k-SAT problem can be formulated as follows:

- An assignment to those variables is a *vector* $v = (x_1, x_2 \ldots x_n) \in \{0,1\}^n$
- A clause C_i of length k is a disjunction of k literals, $C_i = (x_1 \; OR \; x_2 \; OR \ldots OR \; x_k)$
- Each literal is a variable or a negation of a variable
- Each variable can appear multiple times in the expression.

The Max k-SAT problem requests a variable assignment $v \in \{0, 1\}^n$ that makes the formula $F = C_1 \; AND \; C_2 \; AND \ldots AND \; C_m$ evaluate to true. A satisfying complete assignment is called a model of the formula. The Max k-SAT problem can be defined by specifying implicitly a pair (Ω, SC), where Ω is the set of all potentials solution ($\{0, 1\}^n$) and SC is a mapping $\Omega \rightarrow N$, called score of the assignment, equal to the number of true clauses. Consequently, the problem consists of defining the best binary assignment that maximizes the number of satisfied clauses in the Boolean formula. Clearly, there are 2^n potential satisfying assignments for this problem, and it has been proven that the k-SAT problem is NP-complete for any k >2. In this paper we deal with the Max 3-SAT problem. It is obviously that this problem is a combinatorial optimization problem. It appears to be impossible to obtain exact solutions in polynomial time. The main reason is that the required computation grows exponentially with the size of the problem. Therefore, it is often desirable to find near optimal solutions to these problems. Efficient heuristic algorithms offer a good alternative to accomplish this goal. Exploitation of the optimization philosophy and the parallel great ability of natural computing is an attractive way to probe complex problems like the MAX 3-SAT problem. Within this perspective, we are interested in applying PSO principles improved by a stochastic local search method to solve this problem.

3. MAX 3-SAT SOLVERS STATE OF THE ART

Modern Max-SAT solvers have deeply improved the techniques and algorithms to find optimal solutions. In practice there are two broad classes of algorithms for solving variants of SAT: Complete and Incomplete methods. The complete methods examine the entire search space, they fully answer for the problem of satisfiability of an instance. In the worst case, the computation time required for the execution of such methods, increases exponentially with the size of the instance to solve. The method of Davis and Putnam (DP) [24] is one of the first methods dedicated to solving SAT problem. The DP method is based on the application of the resolution rule in order to eliminate variables. Unfortunately, this mechanism may generate an exponential number of resolved clauses. The procedure of Davis, Putnam, Logemann and Loveland (DPLL) [9] is an enhancement of the DP algorithm. The DPLL is a highly efficient and still forms the basis of the most effective SAT solvers. DPLL escapes the exponential clause generation and the high memory usage of DP. This method replaces the resolution mechanism for the separation of the problem into two problems. The idea is to construct a binary tree in which each node represents a DPLL recursive procedure. All the leaves are the search stopping unless an empty clause if the formula is satisfiable. The process search uses the backtracking technique to go back in the search tree and thus to test new branches. The DPLL takes advantage of two techniques that improve the selection of assignments to variables: the Literal Elimination technique [10] and Unit Propagation technique [25]. Currently, the most of exact methods are based mainly on the DPLL procedure. The main difference between them is the branching heuristics used to explore the search tree [26, 27].

The fact that complete algorithms require huge amount of computing time, even for relatively small problem instances, incomplete methods have been developed to resolve SAT or MAX-SAT problems of large size in reasonable time. Incomplete methods do not guarantee finding the best solutions, because they do not explore the entire space of possible solutions. However, the incomplete methods are particularly suitable for large-scale practical MAX-SAT problem

instances. They are able to find quickly a good solution towards the cost and constraints of the problem. The incomplete methods are principally based on metaheuristics such as: local search methods, evolutionary algorithms or hybrid methods. The local search methods are widely used to solve Max 3-Sat problems [28]. Among the popular methods of this class, we can cite the popular method GSAT [14] and its improved variant Walksat [29, 30]. GSAT starts with a random assignment and iteratively apply a set of flips by using a specific heuristics in order to enhance the number of satisfied clauses. Walksat is an evolution of GSAT in which the heuristics used to select the variable to flip is improved. The main difference lies in the estimated size of the neighborhood: Walksat greatly reduces the number of neighbors as it selects a false clause, and only one variable of this clause may be flipped. Unfortunately, Walksat suffers from local minima problem. New heuristics were integrated in Walksat to choose a variable of the clause to flip such as, Best Novelty, Rnovelty and Tabu [28].

Evolutionary algorithms have been applied to SAT problems. However, the use of a pure evolutionary algorithm has been unsuccessful in Max-SAT problems. In fact, Rana and Whitley [31] showed that a classical genetic algorithm is unsuitable for Max 3-Sat problem, because this later requires more intensification search than diversification search. Therefore, hybridization between evolutionary algorithms and local search methods are needed to find success. This kind of hybridization is called memetic algorithms. Recently, several memetic algorithms were proposed to deal with the Max-SAT problems for example: FlipGA [12] based on hybrid genetic algorithm and a special flip heuristics; GASAT [18] based on hybrid genetic algorithm and tabu search, QGASAT [13] based on quantum evolutionary algorithm and local search procedure.

Recently, a new kind of hybridization based on incomplete and complete algorithms has emerged. Although the exact methods are able to provide an optimal solution for instances of small size, incomplete methods are usually able to find solutions close to optimum for instances of large sizes. It is therefore natural that, starting from this observation, many hybrid methods of these two resolution paradigms were considered. There are several methods in the literature based on this kind of hybridization [19].

4. PARTICLE SWARM OPTIMIZATION

The Particle Swarm Optimization algorithms (PSO) are population-based search technique like genetic algorithm but they do not use evolutionary algorithms like mutation and crossover. PSO algorithms were introduced in 1995 by Kennedy and Eberhart [21] as an alternative to the standard genetic algorithms. These algorithms are inspired by the swarms of insects (or of the fish benches or the clouds of birds) and their coordinated movements. The individuals of the algorithm are called particles and the population is called swarm. PSO uses a population (swarm) of particles to explore the search space. Each particle represents a candidate solution of an optimization problem and has an associated position, velocity, and memory vector. In this algorithm, a particle decides on its next movement according to its own experience, which is in this case the memory of the best position than it met, and according to its best neighbor. This vicinity can be defined spatially by taking for example the Euclidean distance between the positions of two particles or socio-metrically (position in the particle swarm). The new speeds and direction of the particle will be defined according to following three tendencies: propensity to go on its own way, its tendency to return towards its best position reached and its tendency to go towards its best neighbor. The algorithms with swarm of particles can apply as well to discrete optimization problems as to continuous problems. The main part of the PSO algorithm [22, 23] can be described formally as follows:

Suppose a swarm of *S* particles. Each particle consists of three elements:

- The first one, is its position in the search space (x_i),
- The second element (v_i), expresses the velocity,
- The third element ($P*_i$), is its memory, used to store the elite particles of the best global particle (Pg) found, as well as the best solutions as found by each individual particle (Pi) so far. This vector is often referred to as personal best in the PSO. The number of P_i particles stored in the memory of the algorithm is equivalent to the number of particles in the swarm.

Unlike replacement methods in conventional genetic algorithms, it's not required to put always in future populations any elite individual found, although each particle in the population tries to be near to the Pg and Pi solutions in the memory by using an updating process governed by PSO weight update equations. Moreover, there is no selection scheme is incorporated in PSO, since the updating of the particles is done by the self-updating equations, and the elite particles (Pi and Pg) are stored in the memory, to prevent the degeneration of the overall fitness of the particle swarm. The self-updating equations of PSO are as follows:

$$v_{id}^{t+1} = w.v_{id}^{t} + c1.\psi 1.(P_{id}^{t} - x_{id}^{t}) + c2.\psi 2.(P_{gd}^{t} - x_{id}^{t}) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (2)$$

v_{id}^{t} : Represent the velocity of the i^{th} particle in iteration *t*.

x_{id}^{t} : Represent the position of the i^{th} particle in iteration *t*.

C_1, C_1: Learning factors

Pi: Best position achieved so long by particle *i*.

Pg: Best position found by the neighbors of particle *i*.

$\psi 1, \psi 2$: Random factors in the [0, 1] interval.

w: Inertia weight.

For binary optimization problems, Kennedy and Eberhart [32,33] have adapted the PSO to search in binary spaces, by applying a sigmoid transformation to the velocity component Eqn. (1) to squeeze the velocities into a range (0,1), and to force the component values of the particles position to be 0's or 1's. The earlier equation for updating positions Eqn. (2) is then replaced by the following equations:

$$sigmoid(v_{id}^t) = \frac{1}{1+e^{-v_{id}^t}}$$
$$x_{id}^t = \begin{cases} 1 & \text{if } rand < sigmoid(v_{id}^t) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

5. THE PROPOSED APPROACH

To solve the Maximum Satisfiability problem, a new approach based on PSO principles and enhanced by a local search procedure is proposed. In this approach, a new objective function is adapted based on the clause weights. In order to show how PSO concepts have been tailored to the problem at hand, we need first to derive a representation scheme which includes the definition of an appropriate particle representation of potential Boolean assignments and the definition of PSO operators. Then, we describe how these defined concepts have been combined with local search algorithm to deal with the Max 3-Sat problem. For correspondence with the swarm system, each particle corresponds to a potential solution of the underlying problem.

5.1 OBJECTIVE FUNCTION

The objective function is very important because it helps to determine if a solution is enough good and generally has a great impact on the performance of evolutionary algorithms. For the Max 3-sat problem, several fitness functions [11] were proposed to be used by the evolutionary algorithms. The standard evaluation function used in the most incomplete methods is the number of satisfied clauses. The objective is to maximize the number of the true clauses. Unfortunately, this kind of objective function cannot lead an evolutionary algorithm to find the best solution. In fact, there are many different solutions having the same quality concerning the Max 3-sat evaluation function, which makes it hard for an evolutionary algorithm to select the appropriate solution leading to the best solution [11]. This observation is obviously clear when dealing with the satisfiable SAT instances. For this reason, we have used an adaptive objective function based on the mechanism of *stepwise adaptation of weights* (SAW) introduced by Eiben et al. [34]. Theobjective function is given by the following formula:

$$F(x) = \sum_{i=1}^{m} W_i C_i(x) \quad (4)$$

Each clause C_i has a weight *Wi*. The objective function is adapted by modifying the weights *Wi* appropriately in order to identify the hard clauses in the preceding search process. A high weight *Wi* indicates that the clause C_i is difficult and leads to a higher fitness of those solutions that satisfy C_i. Consequently, the weights lead the search process towards solutions satisfying the most difficult clauses. In the beginning all weights are initialized by W0 = 1, and later the weights are adjusted according to following formula (x* is the current best particle):

$$W_{i+1} = W_i + 1 - C_i(x^*) \quad (5)$$

5.2 PSO REPRESENTATION OF SOLUTIONS

To successfully apply PSO principles on Max 3-Sat problem, we have needed to map potential solutions into particle representation that could be easily manipulated by the PSO operators. We used a binary representation to encode each position of the particles which represents potential solutions. There are several possible representation of the Max-SAT space search like the floating point representation and the clausal representation [11]. However the most successful evolutionary algorithms for the Sat problems use the bit string representation. Thus, one potential solution is represented by a binary vector of size *n* where *n* is the number of Boolean variables in the CNF formula (Fig. 1). The initial solutions are randomly generated. However, we can start with initial solutions obtained by using greedy heuristics in order to accelerate the optimization process.

0	**1**	**0**	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**

FIGURE 1**:** Binary representation of particle swarm.

5.3 LOCAL SEARCH

The study of the behavior of the PSO algorithm with simple flight operation for Max-Sat problem showed that the convergence speed is slow. In a simple PSO, the positions are set depending on the velocity values using a sigmoid transformation. Unfortunately, this transformation does not give good results for the Max-Sat problem. In order to increase intensification capabilities of search, we have replaced the flight operator by a local search algorithm. The local search is based Flip heuristic used in flipGA [11, 12] in the core of PSO algorithm in order to improve its

performances. Flip heuristic consists in flipping the variables of the Boolean formula (fig. 2).

```
Input: particle (Boolean assignment), formula ϕ in CNF, maxflip.
-   improvement=1;
-   nbrflip=0   ;
-   repeat
-   improvement=0;
-   For i=1 to nVar     // nVar: number of  variables
-   flip the i-th variable of the particle;
-   nbrflip++;
-   compute the gain of flip;
-   If (gain>=0)
-   accept the flip;
-   improvement= improvement +gain;
-   EndIf
-   Else
-   undo the flip of i-th variable of particle;
-   End For
-   Until (improvement > 0 && nbrflip < maxflip)
output: boolean assignment VB
```

FIGURE 2. Local search procedure.

The flip is accepted if there is improvement in the number of satisfied clauses. The process is repeated until there is no improvement. However, the experiments showed that the performance to find the exact solutions for satisfiable instances is low and the convergence speed is slowed down. In this section, we show how PSO algorithm can be greatly improved by using the objective function SAW (eq.4) in order to focus on difficult clauses and to escape from local optima.

5.4 OUTLINE OF THE PROPOSED FRAMEWORK

Now, we describe how the representation scheme including particle representation and PSO operators has been hybridized with local search algorithm and resulted in a hybrid stochastic algorithm performing variable truth assignment search [20]. In more details, the proposed approach can be described as in Figure 3.

Firstly, a swarm of particles is initialized using binary strings, each encoding a given solution. The population size is important factor in evolutionary algorithms. A large population represents a large space search of solutions, but increases the computational cost. On the other hand, a small population size can lead to local solutions. Seen, we use a local search method in the core of PSO algorithm, it is preferably to reduce the population size. We have found that a population between 10 and 30 can give good results. The proposed algorithm progresses through a number of generations. On each generation, we apply the velocity and the position updates operations to make a new population. However, we have noted that the position update operation which is based on the sigmoid transformation is not efficient to deal with Max-SAT problem. In order to enhance the capacity of the intensification search space and to explore the particle to a better position, we have replaced this transformation by the flip heuristic described previously. The whole process is repeated until the stopping criterion is met.

```
Input: Boolean formula in CNF form A
1) Initialize the population, positions and velocities
2) Evaluate the fitness of the each particle (Pi)
3) Apply randomly the interference operation on each particle
4) Save the individuals highest fitness (Pg)
5) Modify velocities based on Pi and Pg position
6) Update the particles position using flight
7) Update particle best
8) Update global best
9) Terminate if the condition is met
10) Go to Step 2
Output: best variable assignment, number of satisfied clauses
```

FIGURE 3. The proposed approach for max 3-sat problem.

6. EXPERIMENTAL RESULTS

The developed method called WPSOSAT (Weighted Particle Swarm Optimization for SATisfiability problems) is implemented in C++ and tested on a micro-computer 3 GHz and 1 GB of memory. To assess the experimental performance of WPSOSAT, several tests are conducted with instances taken from the AIM benchmark instances and three hard instances from the family benchmarks *f* (f600, f1000 and f2000). The AIM instances are all generated with a particular Random-3-SAT instance generator [35]. The particularity is that the generator produces yes-instances and no-instances independently for wide ranges. The used AIM instances include yes-instances with low and high clause/variable ratios that have exactly one solution. We have focused our experiments on the satisfiable benchmarks because we want to evaluate the capabilities of our approach to find the exact solutions for yes-instances.

To analyze the performances of the approach, we performed a comparison with other popular methods in the literature. We have compared WPSOSAT against two version of sat solver base on PSO algorithm. The first is PSO-LS which use a flight operation based on sigmoid transformation. The purpose of this comparison is to show the effect of the local search on the PSO algorithm. The second method is PSOSAT which is based on PSO algorithm with the standard objective

function (number of true clauses). The objective of this comparison is to show the effect the objective function on particle swarm optimization. Furthermore, we compared our results with those of another evolutionary algorithm Clonsat which is based on a hybrid Clonal selection algorithm and Walksat procedure [13]. We have made a comparison with three state-of-art stochastic local search programs: Walksat, Novelty [30] and an Iterated Robust Tabu Search for Max-Sat IROTS [36], for these programs, we have used the default parameters used in the program UBCSAT [28]. Finally, statistical tests of Freidman were carried out to test the significance of the difference in the accuracy of each method in this experiment. In all experiments, we have run the WPSOSAT program using the parameters' settings shown in Table 1. For each program, the best of five consecutive runs is taken.

TABLE 1. PARAMETERS OF WPSOSAT

Parameter name	Parameter value
Population Size	20
Topology	Ring
Total number of generations	1000
Neighborhood Size	1
Velocity_Init_[Min,Max]	-1,1
Velocity_[Min,Max]	-1,1
Learning_Factor1- Factor2 (velocity)	1,7-2,1
Inertia_Factor (velocity)	1
Number of flip for 'Aim instances'	30000

The Table 2 summarizes the obtained results for unsatisfiable tests. All the programs succeed in this experiment. The table three summarizes the obtained results for satisfiable tests. The results found seem to be very promising and demonstrate the feasibility and efficiency of using the adaptive objective function to deal with the Max 3-Sat problem (Table 3). Indeed, the found results are similar to the exact solutions as it's confirmed by the Friedman test (Fig. 4). As we can see, except the program WPSOSAT, all the other programs used in this experiment fail to find the exact solutions for the benchmarks *aim-100-1_6-yes1-1:4* and *aim-100-2_0-yes1-1:4*. Compared to PSOSAT, WPSOSAT is obviously better, thanks to the adaptive weighted function. PSO-LS is not competitive to deal with yes-instances; it ranks the last in the Freidman test (Fig. 4). Therefore, the performance of PSO algorithm without the local search algorithm is rather poor. By another hand, the experiment shows that the programs Walksat, Novelty+ and PSOSAT have similar performances, but they rank third in the Freidman test. However, the results of the programs IROTS and Clonsat are very close to the exact solutions.

The study of the optimization behaviour of our approach reveals that finding the optimal solution is not conducted in a random manner. The best solution is gradually improved until the terminal conditions are reached. Table 4 shows the performance of WPSOSAT in three hard instance f600, f1000 and f2000. The results are compared to those found by Max-SAT solver based on Scatter Search algorithm SAT_SS [38]. The WPSOSAT results in these three instances are near to those of SAT_SS and better than Walksat results. Figure 5 shows an example of the best solution variation for the three hard instances f600, f1000 and f2000. The great problem encountered using the local search method is the big number of flips used to find the best solution. To overcome this problem, we can introduce the tabu search principles [17] in order to reduce the number of neighbors. We put in the tabu list each variable that does not improve the objective function.

TABLE 2. RESULTS FOR AIM UNSATISFIABLE TESTS.

Tests	#clause	Novelty+	Walksat	IROTS	PSOSAT	Clonsat	WPSOSAT
aim-50-1_6-no	80	79	79	79	79	79	79
aim-50-2_0-no	100	99	99	99	99	99	99
aim-100-1_6-no	160	159	159	159	159	159	159
aim-100-2_0-no	200	199	199	199	199	199	199
aim-200-2_0-no	400	399	399	399	399	399	399

TABLE 3. RESULTS FOR AIM SATISFIABLE TESTS.

Tests	PSO-LS	PSOSAT	WPSOSAT	Novelty+	Walksat	IROTS	Clonsat
aim-50-1_6-yes1:4	79	79	80	79	79	79,75	79,75
aim-50-2_0-yes1-1:4	98,25	100	100	100	100	100	100
aim-50-3_4-yes1-1:4	165	170	170	170	170	170	170
aim-50-6_0-yes1-1:4	288,25	300	300	300	300	300	300
aim-100-1_6-yes1-1:4	154,25	159	160	159	159	159	159
aim-100-2_0-yes1-1:4	191	199	200	199	199	199	199
aim-100-3_4-yes1-1:4	319,5	340	340	340	340	340	340
aim-100-6_0-yes1-1:4	557	600	600	600	600	600	600
aim-200-2_0-yes1-1:4	373,25	399	400	399	399	399	399
aim-200-6_0-yes1-1:4	1095,25	1200	1200	1200	1200	1200	1200

TABLE 4. RESULTS FOR F600- F1000- F2000 SATISFIABLE TESTS.

test	#var	#clause	WPSOSA	SAT_S	Walksa
f600	600	2550	2549	2550	2547
f1000	100	4250	4249	4247	4243
f2000	200	8500	8483	8495	8486

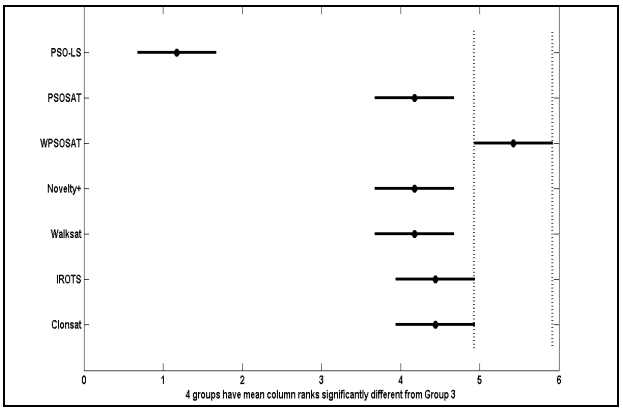

FIGURE 4. Freidman test for satisfiable tests.

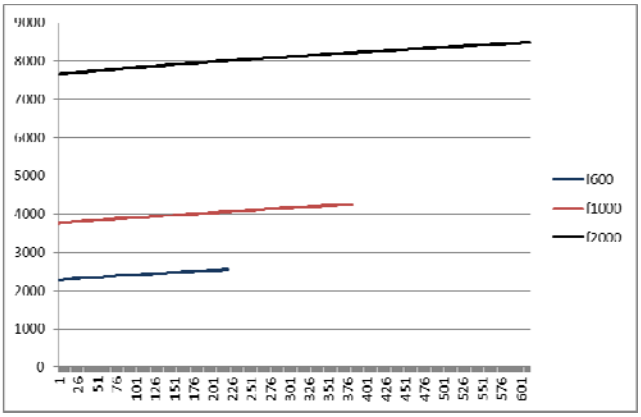

FIGURE 5. The behavior of the best solutions for the benchmarks F600, F1000, and F2000.

7. CONCLUSION

In this paper, a new approach to solve the Max-3-SAT problem is proposed. The developed approach is based on hybridizing binary particle swarm optimization algorithm with a local search algorithm based on flip heuristic. The adaptive objective function based on the clauses weights has been used in order to increase the performance of our approach. Our approach yields high quality solutions. It reaps advantage from characteristics of PSO algorithm and local search, better balance between intensification and diversification of the search is achieved. According to the experimental results, the choice of the local search procedure is crucial for the effectiveness of the resulting algorithm. In most cases, our program gives comparable or better solutions than other programs of literature. In addition, the proposed framework provides an extensible platform for evaluating different variant of satisfiability problems.

REFERENCES

[1] Cook S.A., "The Complexity of Theorem Proving Procedures," In: Proc. 3rd Ann. ACM Symp. *On Theory of Computing, Association for Computing Machinery*, pp 151–158, 1971.

[2] Maric, F. "Formal verification of a modern SAT solver by shallow embedding into Isabelle/HOL". *Theoretical Computer Science*, Vol.411, No.50, pp. 4333–4356, 2010.

[3] Bouhmala, N. and Granmo, O. "Solving Graph Coloring Problems Using Learning Automata," *In Proceedings of the Eighth European Conference on Evolutionary Computation in Combinatoral Optimisation, Lecture Notes in Computer Science* Vol. 4972, pp. 277-288, 2008.

[4] Kautz H. and Selman B. "Pushing the envelope: planning, propositional logic and stochastic search," *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI 96), AAAI Press*, pp 1194 -1201, 1996.

[5] Kautz H. and Selman B. "The state of SAT," *Discrete Applied Mathematics*, 155 (12):1514-1524, 2007.

[6] Zhang H, Shen H. "Exact Algorithms for MAX-SAT," *Electronic Notes in Theoretical Computer Science*, Vol. 86(1), 2003.

[7] Choi, A., Standley,T. and Darwiche, A. "Approximating weighted Max-SAT problems by compensating for relaxations," *Proceedings of the 15th International* Conference on Principles and Practice of Constraint Programming, pp. 211–225, 2009.

[8] Menai M, Batouche, M. "A Backbone-Based Co-evolutionary Heuristic for Partial MAX-SAT," *Artificial Evolution*, pp 155–166, 2005.

[9] Davis M, Putnam G, Loveland D. "A machine program for theorem proving," *communication of the ACM*, pp 394–397, 1962.

[10] Li CM, Manyà F, Planes J. "Exploiting unit propagation to compute lower bounds in branch and bound Max-SAT solvers," *In: Proc. of the 11th Int. Conf. on Principles and Practice of Constraint Programming, CP'05*, Sitges, Spain, in: LNCS 3709, pp. 403–414, 2005.

[11] Gottlieb J, Marchiori E, and Rossi C. "Evolutionary algorithms for the satisfiability problem," Evolutionary Computation, Vol. 10. No. 2, pp. 35–50, 2002.

[12] Marchiori E, Rossi C. "A Flipping Genetic Algorithm for Hard 3-SAT Problems," *In: Proc. of the Genetic and Evolutionary Computation Conference*, Vol. 1, pp. 393–400, 1999.

[13] Layeb A, Saidouni D. "A New Quantum Evolutionary Local Search Algorithm for MAX 3-SAT Problem," *In Proceedings of the 3rd international Workshop on Hybrid Artificial intelligence Systems. Lecture Notes in Artificial Intelligence*, Vol. 5271, 172–179, 2008.

[14] Selman B, Levesque H, Mitchell D. "GSAT- A new method for solving hard satisfiability problems," *In Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, San Jose, Ca., pp 440–446, 1992.

[15] Skvortsov, E.S. "A Theoretical Analysis of Search in GSAT," *Proceedings of the SAT*, pp. 265–275, 2009.

[16] Marques-Silva JP, Sakallah K.A. "GRASP: A Search Algorithm for Propositional Satisfiability," *IEEE Transactions on Computers*, Vol. 48, No. 5, pp. 506–521, 1999.

[17] Mastrolilli M, and Gambardella LM. " Maximum satisfiability: how good are tabu search and plateau moves in the worst-case*?", European Journal of Operational Research*, Vol. 166, No, 1, pp. 63–76,2005.

[18] Lardeux F, Saubion F, Hao JK. "GASAT: A genetic local search algorithm for the satisfiability problem," *Evolutionary Computation*, Vol. 14, No. 2, pp. 223–253, 2006.

[19] Kroc L., Saharawi A, Gomes CP, Selman B. "Integrating systematic and local search paradigms: A new strategy for MaxSAT," *Proceedings of IJCAI*, pp 544–551, 2009.

[20] Layeb, A., Deneche, A. H, Meshoul, M. "A New Artificial Immune System for Solving the Maximum Satisfiability Problem," *In: N. García-Pedrajas et al. (Eds.): IEA/AIE 2010*, part II, Springer LNAI, 6097: 136–142, 2010.

[21] Eberhart, R. and Kennedy, J. "A new optimizer using particle swarm theory," *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, 1995.

[22] Sha, D.Y. and Hsu, C.Y. "A hybrid particle swarm optimization for job shop scheduling problem," *Computers and Industrial Engineering*, Vol. 51, pp. 791–808, 2006.

[23] Fan, S.S. and Zahara, E. "A hybrid simplex search and particle swarm optimization for unconstrained optimization," *European Journal of Operational Research*, Vol.181, pp. 527–548, 2007.

[24] Davis M, Putnam H. " A computing procedure for quantification theory," *Journal of the ACM*, 7(3):201–215, 1960.

[25] Li, C.M., Manyà, F. and Planes, J. "Exploiting unit propagation to compute lower bounds in branch and bound Max-SAT solvers," *Proc. of the 11th Int. Conf. on Principles and Practice of Constraint Programming, CP'05*, Lecture Notes in Computer Science, Vol. 3709, pp. 403–414, 2005.

[26] Kilani, Y. "Comparing the performance of the genetic and local search algorithms for solving the satisfiability problems," *Applied Soft Computing*, Vol. 10, No. 1, pp. 198–207, 2010.

[27] Xu, L., Hutter, F. , Hoos, H.H., and Leyton-Brown, K, "SATzilla: portfolio-based algorithm Selection for SAT," *Journal of Artificial Intelligence Research*, Vol. 32, pp. 565–606, 2008.

[28] Tompkins D.A.D, Hoos H.H. "UBCSAT: An Implementation and Experimentation Environment for SLS Algorithms for SAT and MAX-SAT," *In: Hoos HH and Mitchell DG (Eds.) SAT 2004*, LNCS, Vol. 3542, pp. 306–320, 2005.

[29] Selman B, Kautz H, Cohen B. "Local Search Strategies for Satisfiability Testing," *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 26:521–532, 1993.

[30] Prestwich, S. "Random walk with continuously smoothed variable weights," *Proceedings of the Eight International Conference on Theory and Applications of Satisfiability Testing (SAT-05)*, Lecture Notes, Vol. 3569, pp. 203–215, 2005.

[31] Rana S, Whitley, D, Heckendorn R.B. "A Tractable Walsh Analysis of SAT and its Implications for Genetic Algorithms," *In proceedings of the 15th National Conference on Artificial Intelligence*, pp 392–397, 1998.

[32] Kennedy J, Eberhart RC. "A discrete binary version of the particle swarm algorithm," *In: Proceedings of IEEE conference on systems, man and cybernetics*, p. 4104–4109, 1997.

[33] Bratton, D. and Kennedy,J. " Defining a standard for particle swarm optimization," *Swarm Intelligence Symposium*, pp. 120–127, 2007.

[34] Gottlieb J, and Voss N. "Adaptive fitness functions for the satisfiability problem". *In: Parallel Problem Solving from Nature - PPSN VI 6th International Conference*, Paris, France. Springer Verlag. LNCS 1917, 2000.

[35] Asahiro Y, Iwama K, Miyano E. "Random Generation of Test Instanzes with Controlled Attributes," *In: Johnson, D.S., Trick, M.A. (eds.) Cliques, Coloring, and Satisfiability: The Second DIMACS Implementation Challenge. DIMACS Series on Discr. Math. and Theor. Comp. Sci*, Vol. 26, pp. 377–394, 1996.

[36] McAllester,D. Selman, B. and Kautz, H. "Evidence for invariants in local search,". *In: Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pp. 321-326, 1997 .

[37] Smyth, K., Hoos, H. H., Stützle, T. "Iterated Robust Tabu Search for MAX-SAT", *In: Canadian Conference on AI, Lecture Notes in Computer Science*, Vol. 2671, pp. 129-144, 2003.

[38] Boughaci D., Drias H., Benhamou B." Solving Max-Sat Problems Using a memetic evolutionary Meta-heuristic," *In proceedings of 2004 IEEE CIS*, pp 480-484, 2004.