8

A Multilevel Approach Applied to Sat-Encoded Problems

Noureddine Bouhmala
Vestfold University College
Norway

1. Introduction

1.1 The satisfiability problem

The satisfiability problem (SAT) which is known to be NP-complete (7) plays a central role problem in many applications in the fields of VLSI Computer-Aided design, Computing Theory, and Artificial Intelligence. Generally, a SAT problem is defined as follows. A propositional formula $\Phi = \bigwedge_{j=1}^{m} C_j$ with m clauses and n Boolean variables is given. Each Boolean variable, $x_i, i \in \{1, \ldots, n\}$, takes one of the two values, ***True*** or ***False***. A clause , in turn, is a disjunction of literals and a literal is a variable or its negation. Each clause C_j has the form:

$$C_j = \left(\bigvee_{k \in I_j} x_k \right) \vee \left(\bigvee_{l \in \bar{I}_j} \bar{x}_l \right),$$

where $I_j, \bar{I}_j \subseteq \{1, ...n\}$, $I \cap \bar{I}_j = \emptyset$, and $\bar{x}_i$ denotes the negation of x_i. The task is to determine whether there exists an assignment of values to the variables under which Φ evaluates to ***True***. Such an assignment, if it exists, is called a satisfying assignment for Φ, and Φ is called satisfiable. Otherwise, Φ is said to be unsatisfiable. Since we have two choices for each of the n Boolean variables, the size of the search space S becomes $|S| = 2^n$. That is, the size of the search space grows exponentially with the number of variables. Since most known combinatorial optimization problems can be reduced to SAT (8), the design of special methods for SAT can lead to general approaches for solving combinatorial optimization problems. Most SAT solvers use a Conjunctive Normal Form (CNF) representation of the formula Φ. In CNF, the formula is represented as a conjunction of clauses, with each clause being a disjunction of literals. For example, $P \vee Q$ is a clause containing the two literals P and Q. The clause $P \vee Q$ is satisfied if either P is ***True*** or Q is ***True***. When each clause in Φ contains exactly k literals, the resulting SAT problem is called k-SAT.

The rest of the paper is organized as follows. Section 2 provides an overview of algorithms used for solving the satisfiability problem. Section 3 reviews some of the multilevel techniques that have been applied to other combinatorial optimization problems. Section 4 gives a general description of memetic algorithms. Section 5 introduces the multilevel memetic algorithm. Section 6 presents the results obtained from testing the multilevel memetic algorithm on large industrial instances. Finally, in Section 7 we present a summary and some guidelines for future work.

2. SAT solvers

One of the earliest local search algorithms for solving SAT is GSAT (36)(3)(38). Basically, GSAT begins with a random generated assignment of values to variables, and then uses the steepest descent heuristic to find the new variable-value assignment which best decreases the number of unsatisfied clauses. After a fixed number of moves, the search is restarted from a new random assignment. The search continues until a solution is found or a fixed number of restarts have been performed. Another widely used variant of GSAT is the Walksat algorithm and its variants in (37)(30)(17)(26)(27)(18). It first picks randomly an unsatisfied clause, and then, in a second step, one of the variables with the lowest *break count*, appearing in the selected clause, is randomly.

Other algorithms (11)(15) (9) (10) used history-based variable selection strategies in order to avoid flipping the same variable selected.In parallel to the development of more sophisticated versions of randomized improvement techniques, other methods based on the idea of modifying the evaluation function (47)(19)(40)(34)(35) in order to prevent the search from getting stuck in non attractive areas of the underlying search space have become increasingly popular in SAT solving. A new approach to clause weighting known as Divide and Distribute Fixed Weights (DDFW) (20) exploits the transfer of weights from neighboring satisfied clauses to unsatisfied clauses in order to break out from local minima. Recently, a strategy based on assigning weights to variables (33) instead of clauses greatly enhances the performance of the Walksat algorithm, leading to the best known results on some benchmarks.

Evolutionary algorithms are heuristic algorithms that have been applied to SAT and many other NP-complete problems. Unlike local search methods that work on a current single solution, evolutionary approaches evolve a set of solutions. GASAT (21)(24) is considered to be the best known genetic algorithm for SAT. GASAT is a hybrid algorithm that combines a specific crossover and a tabu search procedure. Experiments have shown that GASAT provides very competitive results compared with state-of-art SAT algorithms. Gottlieb at al. proposed several evolutionary algorithms for SAT (13). Results presented in that paper show that evolutionary algorithms compare favorably to Walksat. Finally, Boughaci et al. introduced a new selection strategy (5) based on both fitness and diversity to choose individuals to participate in the reproduction phase of a genetic algorithm. Experiments showed that the resulting genetic algorithm was able to find solutions of a higher quality than the scatter evolutionary algorithm (4).

Lacking the theoretical guidelines while being stochastic in nature, the deployment of several meta-heuristics involves extensive experiments to find the optimal noise or walk probability settings. To avoid manual parameter tuning, new methods have been designed to automatically adapt parameter settings during the search (25)(32), and results have shown their effectiveness for a wide range of problems.

3. Multilevel techniques

The multilevel paradigm is a simple technique which at its core involves recursive coarsening to produce smaller and smaller problems that are easier to solve than the original one. The pseudo-code of the multilevel generic algorithm is shown in Algorithm 1.

The multilevel paradigm consists of four phases: coarsening, initial solution, uncoarsening and refinement. The coarsening phase aims at merging the variables associated with the problem to form clusters. The clusters are used in a recursive manner to construct a

Algorithm 1: The Multilevel Generic Algorithm

input : Problem P_0
output: Solution $S_{final}(P_0)$
begin
level := 0;
While Not reached the desired number of levels $P_{level+1}$:= Coarsen (P_{level});
level := level + 1;
/* Initial Solution is computed at the lowest level */;
$S(P_{level})$ =Initial Solution (P_{level}) ;
While $(level > 0)$ $S_{start}(P_{level-1})$: = Uncoarsen ($S_{final}(P_{level})$);
$S_{final}(P_{level-1})$:= Refine ($S_{start}(P_{level-1})$);
level := level - 1;
end

hierarchy of problems each representing the original problem but with fewer degrees of freedom. The coarsest level can then be used to compute an initial solution. The solution found at the coarsest level is uncoarsened (extended to give an initial solution for the next level) and then improved using a chosen optimization algorithm. A common feature that characterizes multilevel algorithms, is that any solution in any of the coarsened problems is a legitimate solution to the original one. Optimization algorithms using the multilevel paradigm draw their strength from coupling the refinement process across different levels. Multilevel techniques were first introduced when dealing with the graph partitioning problem (GGP) (1) (14) (16) (22) (23) (44) and have proved to be effective in producing high quality solutions at a lower cost than single level techniques. The traveling salesman problem (TSP) was the second combinatorial optimization problem to which the multilevel paradigm was applied (45) (46) and has clearly shown a clear improvement in the asymptotic convergence of the solution quality. When the multilevel paradigm was applied to the graph coloring problem (42), the results do not seem to be in line with the general trend observed in GCP and TSP as its ability to enhance the convergence behavior of the local search algorithms was rather restricted to some class of problems. Graph drawing is another area where multilevel techniques gave a better global quality to the drawing and the author suggests its use to both accelerate and enhance force drawing placement algorithms (43).

4. Memetic Algorithms (MAs)

An important prerequisite for the multilevel paradigm is the use of an optimization search strategy in order to carry out the refinement during each level. In this work, we propose a memetic algorithm (MA) that we use for the refinement phase. Algorithm 2 provides a canonical memetic algorithm.

MAs represent the set of hybrid algorithms that combine genetic algorithms and local search. In general the genetic algorithm improves the solution while the local search fine tunes the solution. They are adaptive based search optimizations algorithms that take their inspiration from genetics and evolution process (31). Memetic algorithms simultaneously examine and manipulate a set of possible solution. Given a specific problem to solve, the input to MAs is an initial population of solutions called individuals or chromosomes. A gene is part of a chromosome, which is the smallest unit of genetic information. Every gene is able to assume different values called allele. All genes of an organism form a genomem which

Algorithm 2: A Canonical Memetic Algorithm

begin

Generate initial population ;
Evaluate the fitness of each individual in the population ;
While (Not Convergence reached) Select individuals according to a scheme to reproduce ;
Breed if necessary each selected pairs of individuals through crossover;
Apply mutation if necessary to each offspring ;
Apply local search to each chromosome ;
Evaluate the fitness of the intermediate population ;
Replace the parent population with a new generation; ;

end

affects the appearance of an organism called phenotype. The chromosomes are encoded using a chosen representation and each can be thought of as a point in the search space of candidate solutions. Each individual is assigned a score (fitness) value that allows assessing its quality. The members of the initial population may be randomly generated or by using sophisticated mechanisms by means of which an initial population of high quality chromosomes is produced.

The reproduction operator selects (randomly or based on the individual's fitness) chromosomes from the population to be parents and enters them in a mating pool. Parent individuals are drawn from the mating pool and combined so that information is exchanged and passed to offspring depending on the probability of the crossover operator. The new population is then subjected to mutation and entered into an intermediate population. The mutation operator acts as an element of diversity into the population and is generally applied with a low probability to avoid disrupting crossover results. The individuals from the intermediate population are then enhanced with a local search and evaluated.

Finally, a selection scheme is used to update the population giving rise to a new generation. The individuals from the set of solutions which is called population will evolve from generation to generation by repeated applications of an evaluation procedure that is based on genetic operators and a local search scheme. Over many generations, the population becomes increasingly uniform until it ultimately converges to optimal or near-optimal solutions.

5. The Multilevel Memetic Algorithm (MLVMA)

The implementation of a multilevel algorithm for the SAT problem requires four basic components: a coarsening algorithm, an initialization algorithm, an extension algorithm (which takes the solution on one problem and extends it to the parent problem), and a memetic algorithm which will be used during the refinement phase. In this section we describe all these components which are necessary for the derivation of a memetic algorithm operating in a multilevel context. This process, is graphically illustrated in Figure 1 using an example with 10 variables. The coarsening phase uses two levels to coarsen the problem down to three clusters. $Level_0$ corresponds to the original problem. A random coarsening procedure is used to merge randomly the variables in pairs leading to a coarser problem with 5 clusters. This process is repeated leading to the coarsest problem with 3 clusters. An initial solution is generated where the first cluster is assigned the value of true and the remaining two clusters are assigned the value false. At the coarsest level, our MA wil generate an initial population and then improves

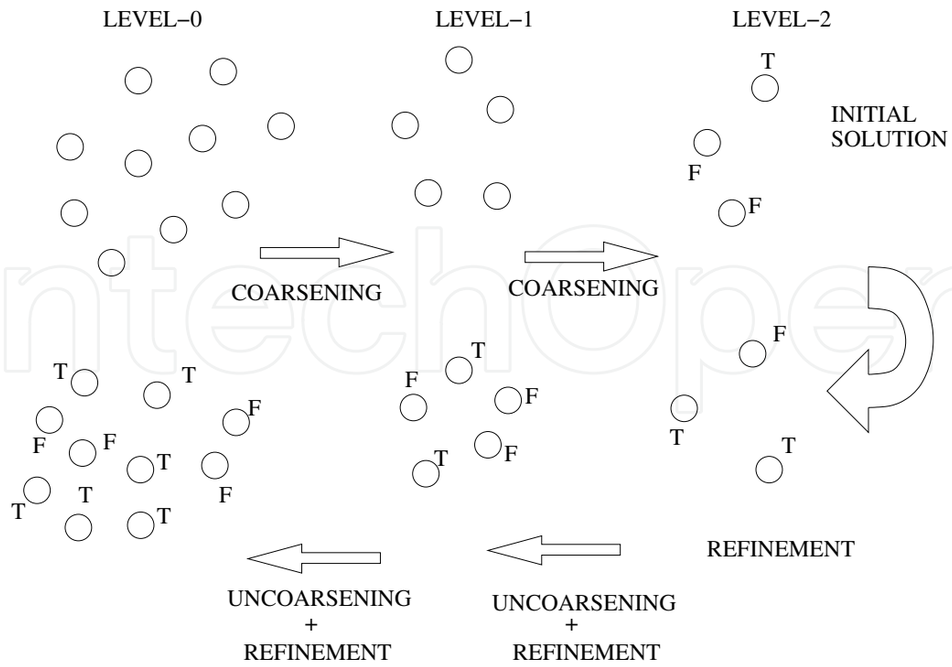


Fig. 1. The various phases of the multilevel memetic algorithm.

it. As soon as the convergence criteria is reached at $Level_2$, the uncoarsening phase takes the solution from that level and extends it to give an initial solution for $Level_1$ and then proceed with the refinement. This iteration process ends when MA reaches the stop criteria that is met at $Level_0$.

5.1 Coarsening

The coarsening procedure has been implemented so that each coarse problem P_{l+1} is created from its parent problem P_l by merging variables and representing each merged pair v_i and v_j with a child variable that we call a cluster in P_{l+1}. The coarsening scheme uses a simple randomized algorithm similar to (16). The variables are visited in a random order. If a variable v_i has not been merged yet, then we randomly select one randomly unmerged variable v_j, and a cluster consisting of these two variables is created. Unmatched variables are simply copied to the next level. The new formed clusters are used to define a new and smaller problem and recursively iterate the coarsening process until the size of the problem reaches some desired threshold.

5.2 Initial solution & refinement

As soon as the coarsening phase is ended, a memetic algorithm is used at different levels. The next subsections describes the main features of the memetic algorithm used in this work.

5.2.1 Fitness function

The notion of fitness is fundamental to the application of memetic algorithms. It is a numerical value that expresses the performance of an individual (solution) so that different individuals can be compared. The fitness of a chromosome (individual) is equal to the number of clauses that are unsatisfied by the truth assignment represented by the chromosome.

5.2.2 Representation

A representation is a mapping from the state space of possible solutions to a state of encoded solutions within a particular data structure. The chromosomes (individuals) which are assignments of values to the variables are encoded as strings of bits, the length of which is the number of variables (or clusters if MA is operating on a coarse level). The values ***True*** and ***False*** are represented by 1 and 0 respectively. In this representation , an individual X corresponds to a truth assignment and the search space is the set $S = \{0, 1\}^n$.

5.2.3 Initial population

A initial solution is generated using a population consisting of 50 individuals. According to our computational experience, larger populations do not bring effective improvements on the quality of the results. At the coarsest level, MA will randomly generate an initial population of 50 individuals in which each gene's allele is assigned the value 0 or 1.

5.2.4 Crossover

The task of the crossover operator is to reach regions of the search space with higher average quality. New solutions are created by combining pairs of individuals in the population and then applying a crossover operator to each chosen pair. Combining pairs of individuals can be viewed as a matching process. The individuals are visited in random order. An unmatched individual i_k is matched randomly with an unmatched individual i_l. Thereafter, the two-point crossover operator is applied using a crossover probability to each matched pair of individuals. The two-point crossover selects two randomly points within a chromosome and then interchanges the two parent chromosomes between these points to generate two new offspring. Recombination can be defined as a process in which a set of configurations (solutions referred as parents) undergoes a transformation to create a set of configurations (referred as offspring). The creation of these descendants involves the location and combinations of features extracted from the parents. The reason behind choosing the two point crossover are the results presented in (41) where the difference between the different crossovers are not significant when the problem to be solved is hard. The work conducted in (39) shows that the two-point crossover is more effective when the problem at hand is difficult to solve. In addition, the author propose an adaptive mechanism in order to have evolutionary algorithms choose which forms of crossover to use and how often to use them, as it solves a problem.

5.2.5 Mutation

The purpose of mutation which is the secondary search operator used in this work, is to generate modified individuals by introducing new features in the population. By mutation, the alleles of the produced child have a chance to be modified, which enables further exploration of the search space. The mutation operator takes a single parameter p_m, which specifies the probability of performing a possible mutation. Let $C = c_1, c_2, ...c_m$ be a chromosome represented by a binary chain where each of whose gene c_i is either 0 or 1. In our mutation operator, each gene c_i is mutated through flipping this gene's allele from 0 to 1 or vice versa if the probability test is passed. The mutation probability ensures that, theoretically, every region of the search space is explored. If on the other hand, mutation is applied to all genes, the evolutionary process will degenerate into a random search with no benefits of the information gathered in preceding generations. The mutation operator prevents the searching

process form being trapped into local optimum while adding to the diversity of the population and thereby increasing the likelihood that the algorithm will generate individuals with better fitness values.

5.2.6 Selection

The selection operator acts on individuals in the current population. During this phase, the search for the global solution gets a clearer direction, whereby the optimization process is gradually focused on the relevant areas of the search space. Based on each individual quality (fitness), it determines the next population. In the roulette method, the selection is stochastic and biased toward the best individuals. The first step is to calculate the cumulative fitness of the whole population through the sum of the fitness of all individuals. After that, the probability of selection is calculated for each individual as being $P_{Selection_i} = f_i / \sum_1^N fi$, where f_i is the fitness of individual i.

5.2.7 Local search

Algorithm 3: local-search

input : $Chromosome_i$
output: A possibly improved $Chromosome_i$;
begin
  PossFlips ← a randomly selected variable with the largest decrease (or smallest increase) in unsatisfied clauses;;
  v ← Pick (PossFlips);;
  $Chromosome_i$ ← $Chromosome_i$ with v flipped ;
  If $Chromosome_i$ satisfies Φ return $Chromosome_i$;
end

Finally, the last component of our MA is the use of local improvers. By introducing local search at this level, the search within promising areas is intensified. This local search should be able to quickly improve the quality of a solution produced by the crossover operator, without diversifying it into other areas of the search space. In the context of optimization, this rises a number of questions regarding how best to take advantage of both aspects of the whole algorithm. With regard to local search there are issues of which individuals will undergo local improvement and to what degree of intensity. However care should be made in order to balance the evolution component (exploration) against exploitation (local search component). Bearing this thought in mind, the strategy adopted in this regard is to let each chromosome go through a low rate intensity local improvement. Algorithm 3 shows the local search algorithm used. This heuristic is used for one iteration during which it seeks for the variable-value assignment with the largest decrease or the smallest increase in the number of unsatisfied clauses. Random tie breaking strategy is used between variables with identical score.

5.2.8 Convergence criteria

As soon as the population tends to loose its diversity, premature convergence occurs and all individuals in the population tend to be identical with almost the same fitness value. During each level, the proposed memetic algorithm is assumed to reach convergence when no further improvement of the best solution (the fittest chromosome) has not been made during two consecutive generations.

5.3 Uncoarsening

Having improved the assignment at the level L_{m+1}, the assignment must be projected onto its parent level L_m. The uncoarsening process is trivial; if a cluster $C_i \in L_{m+1}$ is assigned the value of true then the matched pair of clusters that it represents, C_j and $C_k \in L_m$ are also assigned the value true. The idea of refinement is to use the projected population from L_{m+1} onto L_m as the initial population for further improvement using the proposed memetic algorithm. Even though the population at L_{m+1} is at local minimum, the projected population at level L_m may not be at a local optimum. The projected population is already a good solution and contains individuals with high fitness value, MA will converge quicker within a few generation to a better assignment.

6. Experimental results

6.1 Boundary model checking

The instances used in our experiments arise from model checking (6) which is considered to be one among many real-world problems that are often characterized by large and complex search spaces. Model checking is an automatic procedure for verifying finite-state concurrent systems. Given a model of a design and a specification in temporal logic, one is interested to check whether the model satisfies the specification. Methods for automatic model checking of complex hardware design systems are gaining wide industrial acceptance compared to traditional techniques based on simulation. The most widely used of these methods is called Bounded Model Checking (2) (BMC). In BMC the design to be validated is represented as a finite state machine, and the specification is formalized by writing temporal logic properties. The reachable states of the design are then traversed in order to verify the properties. The basic idea in BMC is to find bugs or counterexamples of length k.

In practice, one looks for longer counterexamples by incrementing the bound k, and if no counterexample exists after a certain number of iterations , one may conclude that the correctness of the specification holds. The main drawback with model checking real systems is the so-called state-explosion problem: as the size of of the system being verified increases, the total state space of the system increases exponentially. This problem makes exhaustive search exploration intractable.

In recent years, there has been a growing interest in applying methods based on propositional satisfiability (SAT) (29)(12) in order to improve the scalability of model checking. The BMC problem can be reduced to a propositional satisfiability problem, and can therefore be solved by SAT solvers. Essentially, there are two phases in BMC. In the first phase, the behavior of the system to be verified is encoded as a propositional formula. In the second phase, that formula is given to a propositional decision algorithm, i.e., a satisfiability solver, to either obtain a satisfying assignment or to prove there is none. If the formula is satisfiable, a bug has been located in the design, otherwise one cannot in general conclude that there is no bug; one must increase the bound, and search for "larger bugs".

6.2 Test suite

We evaluated the performance of the multilevel memetic algorithm on a set of large problem instances taken from real industrial bounded model checking hardware designs. This set is taken from the SATLIB website (http://www.informatik.tu-darmstadt.de/AI/SATLIB). All the benchmark instances used in this experiment are satisfiable instances. Due to the

randomization nature of the algorithms, each problem instance was run 20 times with a cutoff parameter (max-time) set to (300*sec*). We use $|.|$ to denote the number of elements in a set, e.g., $|V|$ is the number of variables, while $|C|$ denotes the number of clauses. Table shows the instances used in the experiment. The tests were carried out on a a DELL machine with 800 MHz CPU and 2 GB of memory. The code was written in C and compiled with the GNU C compiler version 4.6. The parameters used in the experiment are listed below:

- Crossover probability = 0.85.
- Mutation probability = 0.1.
- Population size = 50 .
- Stopping criteria for the coarsening phase: The coarsening stops as soon as the size of the coarsest problem reaches 100 variables (clusters). At this level, MA generates an initial population.
- Convergence during the refinement phase: If no improvement of the fitness function of the best individual has not been observed during 10 consecutive generations, MA is assumed to have reached convergence and moves to a higher level.

6.3 Experimental results

Figures 2-9 show how the best assignment (fittest chromosome) progresses during the search. The plots show immediately the dramatic improvement obtained using the multilevel paradigm. The performance of MA is unsatisfactory and is getting even far more dramatic for larger problems as the percentage excess over the solution is higher compared to that of MLVMA. The curves show no cross-over implying that MLVMA dominates MA. The plots suggest that problem solving with MLVMA happens in two phases. The first phase which corresponds to the early part of the search, MLVMA behaves as a hill-climbing method. This phase which can be described as a long one, up to 85% of the clauses are satisfied. The best assignment improves rapidly at first, and then flattens off as we mount the plateau, marking the start of the second phase. The plateau spans a region in the search space where flips typically leave the best assignment unchanged, and occurs more specifically once the refinement reaches the finest level. Comparing the multilevel version with the single level version, MLVMA is far better than MA, making it the clear leading algorithm. The key success behind the efficiency of MLVMA relies on the multilevel paradigm. MLVMA uses the multilevel paradigm and draw its strength from coupling the refinement process across different levels. This paradigm offers two main advantages which enables MA to become much more powerful in the multilevel context:

- During the refinement phase MA applies a local a transformation (i.e, a move) within the neighborhood (i.e, the set of solutions that can be reached from the current one) of the current solution to generate a new one. The coarsening process offers a better mechanism for performing diversification (i.e, the ability to visit many and different regions of the search space) and intensification (i.e, the ability to obtain high quality solutions within those regions).
- By allowing MA to view a cluster of variables as a single entity, the search becomes guided and restricted to only those configurations in the solution space in which the variables grouped within a cluster are assigned the same value. As the size of the clusters varies from one level to another, the size of the neighborhood becomes adaptive and allows the possibility of exploring different regions in the search space while intensifying the search by exploiting the solutions from previous levels in order to reach better solutions.

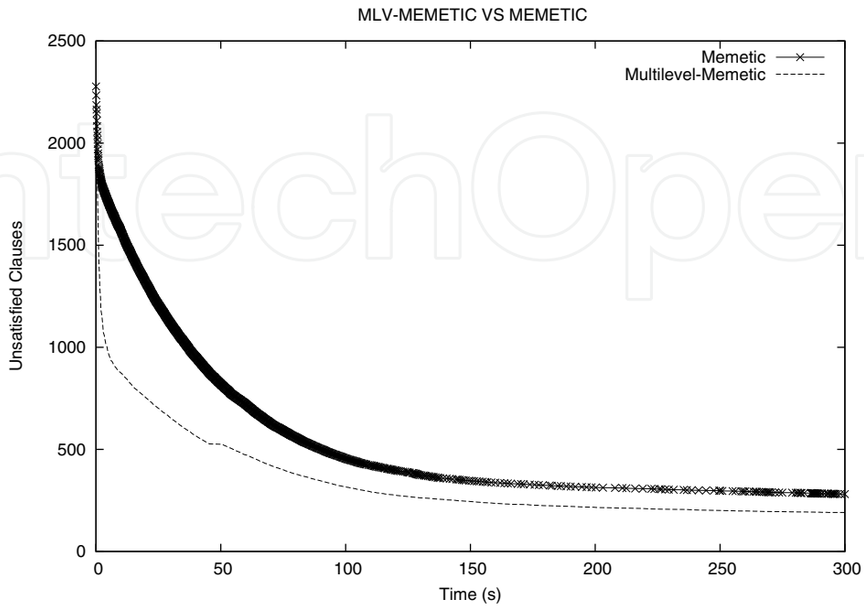


Fig. 2. bmc-ibm-2.cnf: $|V| = 3628$, $|C| = 14468$. Along the horizontal axis we give the time in seconds, and along the vertical axis the number of unsatisfied clauses.

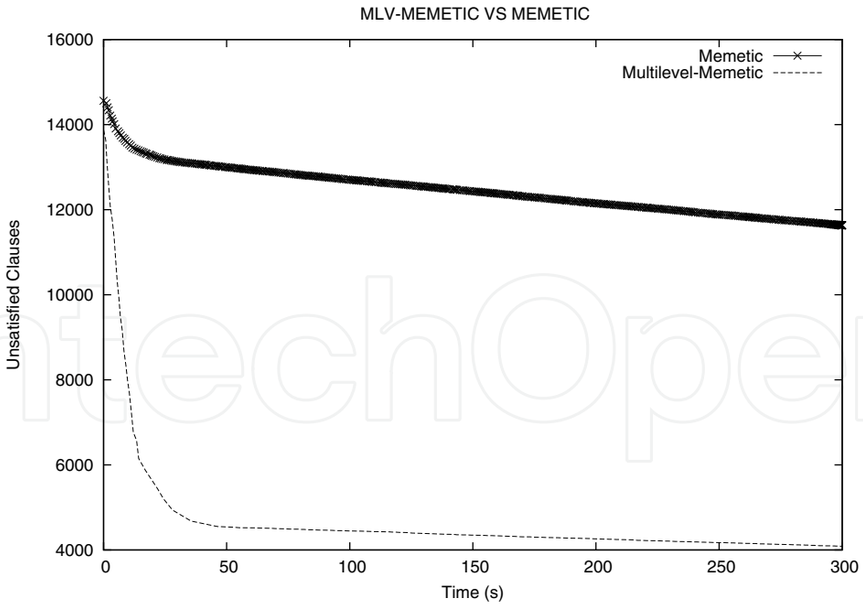


Fig. 3. bmc-ibm-3.cnf: $|V| = 14930$, $|C| = 72106$.. Along the horizontal axis we give the time in seconds , and along the vertical axis the number of unsatisfied clauses.

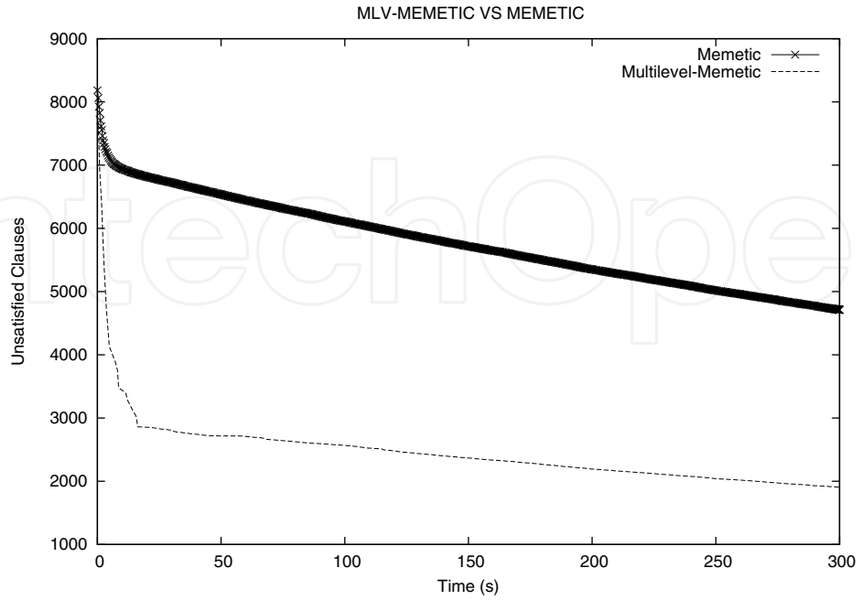


Fig. 4. bmc-ibm-5: $|V| = 9396$, $|C| = 41207$. Along the horizontal axis we give the time in seconds , and along the vertical axis the number of unsatisfied clauses.

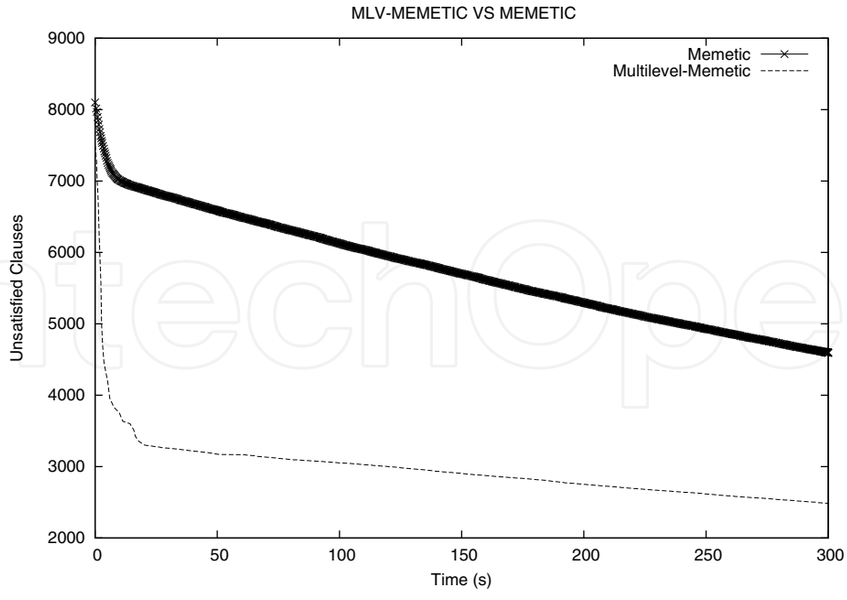


Fig. 5. bmc-ibm-7.cnf: $|V| = 8710$, $|C| = 39774$. Along the horizontal axis we give the time in seconds, and along the vertical axis the number of unsatisfied clauses.

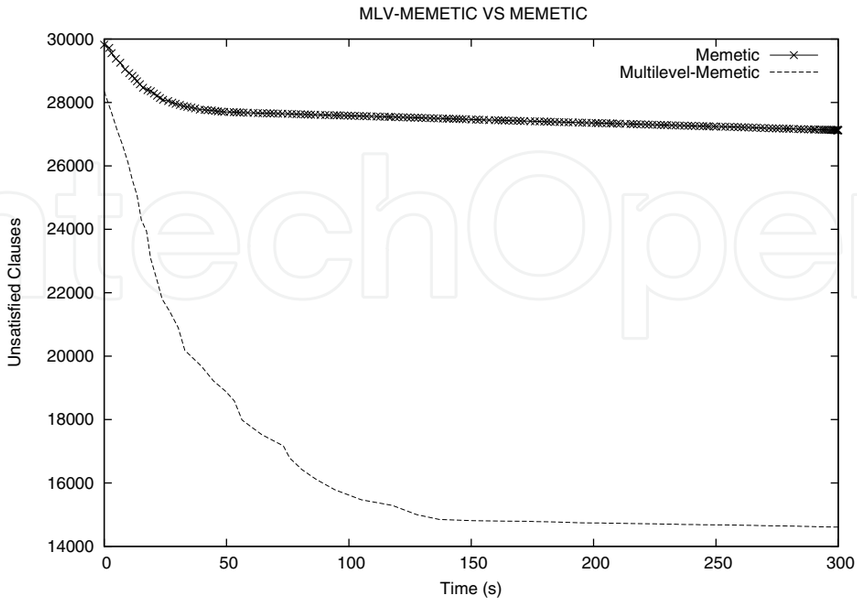


Fig. 6. bmc-ibm-11.cnf: $|V| = 32109$, $|C| = 150027$. Along the horizontal axis we give the time in seconds, and along the vertical axis the number of unsatisfied clauses.

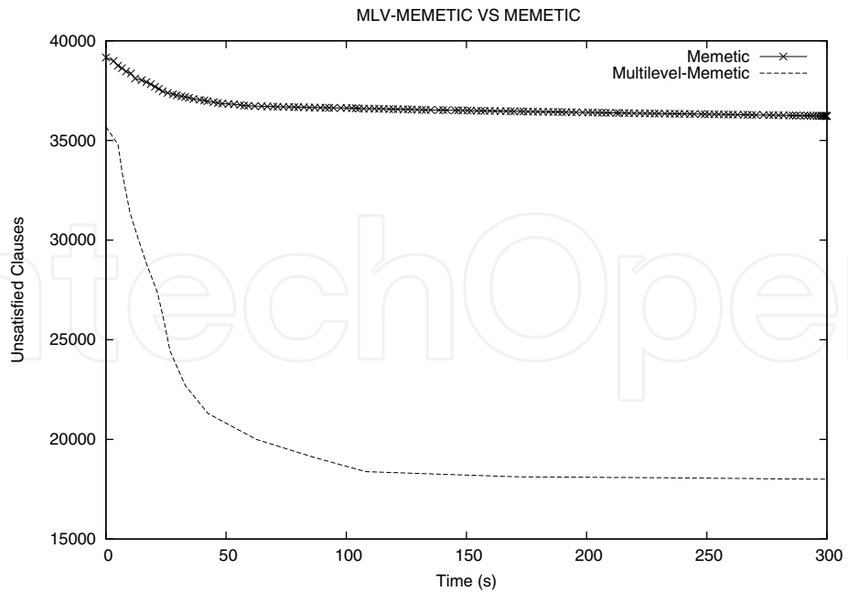


Fig. 7. bmc-ibm-12.cnf: $|V| = 39598$, $|C| = 19477$. Along the horizontal axis we give the time in seconds, and along the vertical axis the number of unsatisfied clauses.

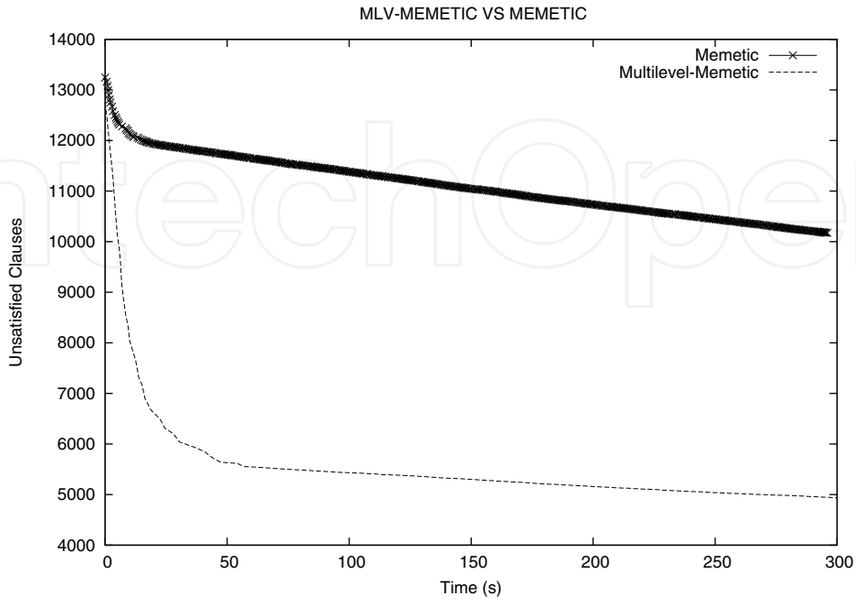


Fig. 8. bmc-ibm-13.cnf: $|V| = 13215$, $|C| = 6572$. Along the horizontal axis we give the time in seconds , and along the vertical axis the number of unsatisfied clauses.

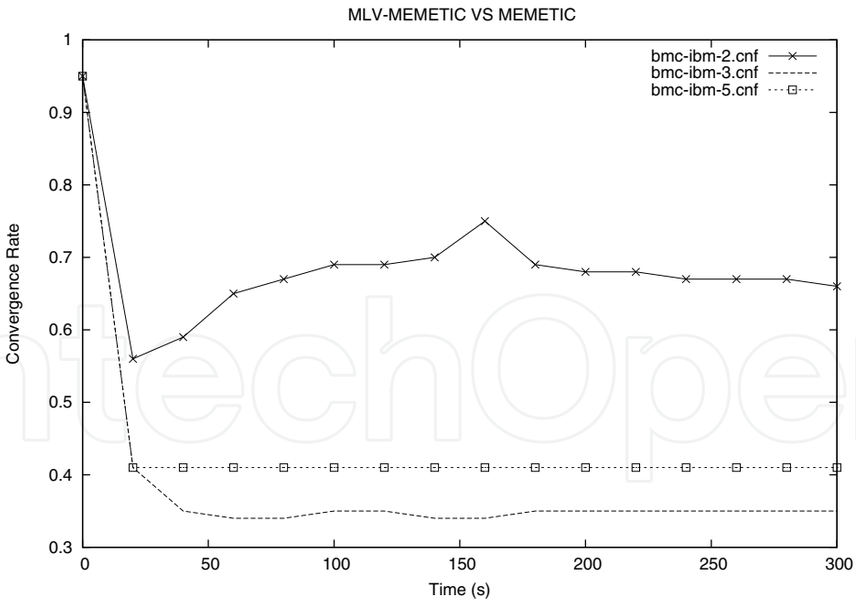


Fig. 9. Results on the convergence behavior for bmc-ibm-2.cnf, bmc-ibm-3.cnf, bmc-ibm-5.cnf. Along the horizontal axis we give the time (in seconds) , and along the vertical axis the convergence rate.

Figures 9 shows the convergence behavior expressed as the ratio between the best chromosome of the two algorithms as a function of time. The plots show that the curves are below the value 1 leading to conclude that MLVMA is faster compared to MA. The asymptotic performance offered by MLVMA is impressive, and dramatically improves on MA. In some cases, The difference in performance reaches 30% during the first seconds, and maintains it during the whole search process. However, on other cases, the difference in performance continues to increase as the search progresses.

7. Conclusion

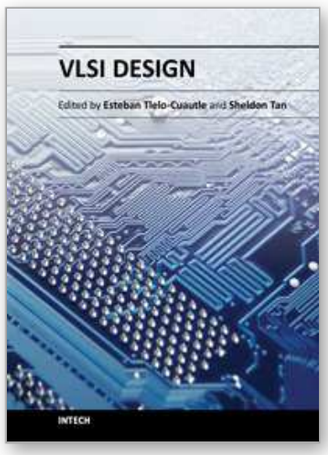
In this work, we have described a new approach for addressing the satisfiability problem. which combines the multilevel paradigm with a simple memetic algorithm. Thus, in order to get a comprehensive picture of the new algorithm's performance, we used a set of benchmark instances drawn from Bounded Model Checking. The experiments have shown that MLVMA works quite well with a random coarsening scheme combined with a simple MA used as a refinement algorithm. The random coarsening provided a good global view of the problem, while MA used during the refinement phase provided a good local view. It can be seen from the results that the multilevel paradigm greatly improves the MA and always returns a better solution for the equivalent runtime. The quality of the solution provided by MLVMA can get as high as 77%. A scale up test shows that the difference in performances between the two algorithms increases with larger problems. Our future work aims at investigating other coarsening schemes and study other parameters which may influence the interaction between the memetic algorithm and the multilevel paradigm.

8. References

[1] S.T. Barnard and H.D. Simon. A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. Concurrency: Practice and Experience, 6(2) pages: 101-117, 1994.
[2] A.Biere at al. Bounded model checking. Advances in Computers,2003.
[3] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys, 35(3) pages: 268-308, September 2003.
[4] D. Boughaci, H. Drias. Efficient and experimental meta-heuristics for MAX-SAT problems. In Lecture Notes in Computer Sciences, WEA 2005, vol. 3503/2005, pages: 501-512, 2005.
[5] D. Boughaci, B. Benhamou, and H. Drias. Scatter Search and Genetic Algorithms for MAX-SAT Problems. J.Math.Model.Algorithms, pages: 101-124, 2008.
[6] E.M. Clark, E.A. Emerson, and A.P.Sista. Automatic verification of finite state on current systems using temporal logic specifications. ACM Transactions on Programming Languages and Systems 8 , pages: 244-263, 1996.
[7] S.A. Cook. The complexity of theorem-proving procedures. Proceedings of the Third ACM Symposium on Theory of Computing, pages: 151-158, 1971.
[8] M.R. Gary and D.S. Johnson. Computers and intractability: A guide to the theory of NP-completeness. W.H. Freeman and Company, New York, 1979.
[9] I. Gent and T. Walsh. Unsatisfied Variables in Local Search. In J. Hallam, editor, Hybrid Problems, Hybrid Solutions, pages: 73-85. IOS Press, 1995.
[10] L.P. Gent and T. Walsh. Towards an Understanding of Hill-Climbing Procedures for SAT. Proceedings of AAAI'93, pages: 28-33. MIT Press, 1993.

[11] F. Glover. Tabu Search-Part1. ORSA Journal on Computing, 1(3), pages: 190-206, 1989.
[12] E. Goldberg, Y. Novikov. Bermin: a Fast and Robust SAT-solver. Proc.of the Design, Automation and Test in Europe, IEEE Computer Society, 2002.
[13] J. Gottleib, E. Marchiori, and C. Rossi. Evolutionary Algorithms for the satisfiability problem. Evolutionary Computation, 10(1), pages: 35-50, 2002.
Satisfiability Problem Using Finite Learning Automata. International Journal of Computer Science and Applications, Volume 4, Issue 3, pages: 15-29, 2007.
[14] R. Hadany and D. Harel. A multi-scale algorithm for drawing graphs nicely. Tech.Rep.CS99-01, Weizmann Inst.Sci., Faculty Maths.Comp.Sci, 1999.
[15] P. Hansen and B. Jaumand. Algorithms for the Maximum Satisfiability Problem. Computing, 44, pages: 279-303, 1990.
[16] B. Hendrickson and R. Leland. A multilevel algorithm for partitioning graphs. In S. Karin, editor, Proc. Supercomputing'95, San Diego, 1995. ACM Press, New York.
[17] H. Hoos. On the run-time behavior of stochastic local search algorithms for SAT. In. Proceedings of AAAI-99, pages: 661-666, 1999.
[18] H.Hoos. An adaptive noise mechanism for Walksat. In Proceedings of the Eighteen National Conference in Artificial Intelligence (AAAI-02), pages: 655-660, 2002.
[19] F. Hutter, D. Tompkins, H. Hoos. Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In Proceedings of the Eight International Conference of the Principles and Practice of Constraint Programming (CP'02), pages: 233-248, 2002.
[20] A. Ishtaiwi, J. Thornton, A. Sattar, and D.N.Pham. Neighborhood clause weight redistribution in local search for SAT. Proceedings of the Eleventh International Conference on Principles and Practice Programming(CP-05), volume 3709 of Lecture Notes in Computer Science, pages: 772-776, 2005.
[21] H. Jin-Kao, F. Lardeux, and F. Saubion. Evolutionary computing for the satisfiability problem. In Applications of Evolutionary Computing, volume 2611 of LNCS, pages: 258-267, University of Essex, England, UK, April 2003.
[22] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comput., 20(1) pages: 359-392, 1998.
[23] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. J. Par. Dist. Comput., 48(1), pages: 96-129, 1998.
[24] F. Lardeux, F. Saubion, and Jin-Kao. GASAT: A Genetic Local Search Algorithm for the Satisfiability Problem. Evolutionary Computation, volume 14 (2), MIT Press, 2006.
[25] C.M. Li, W. Wei, and H. Zhang. Combining adaptive noise and look-ahead in local search for SAT. Proceedings of the Tenth International Conference on Theory and Applications of Satisfiability Testing(SAT-07), volume 4501 of Lecture Notes in Computer Science, pages: 121-133, 2007.
[26] C.M. Li, W.Q. Huang. Diversification and determinism in local search for satisfiability. In proceedings of the Eight International Conference on Theory and Applications of Satisfiability Testing (SAT-05), volume 3569 of Lecture Notes in Computer Science, pages: 158-172, 2005.
[27] C.M Li, W. Wei, and H. Zhang. Combining adaptive noise and look-ahead in local search for SAT. In Proceedings of the Tenth International Conference on Theory and Applications of Satisfiability Testing (SAT-07), volume 4501 of Lecture Notes in Computer Science, pages: 121-133, 2007.
[28] M. Lozano and C.G. Martinez. Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report. Computers and operations Research, 37, pages: 481-497, 2010.

[29] Y. Mahajan, Z. Fu, S. Malik. Zchaff2004: An efficient SAT solver. To appear in SAT 2004 Special Volume
[30] D. McAllester, B. Selman, and H.Kautz. Evidence for Invariants in Local Search. Proceedings of AAAI'97, pages: 321-326. MIT Press, 1997.
[31] P.A. Moscato. On evolution search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Technical Report Caltech Concurrent Computation Program, Caltech,Pasadena, California, 1989.
[32] D.J. Patterson and H. Kautz. Auto-Walksat: A Self-Tuning Implementation of Walksat. Electronic Notes on Discrete Mathematics 9, 2001.
[33] S. Prestwich. Random walk with continuously smoothed variable weights. Proceedings of the Eight International Conference on Theory and Applications of Satisfiability Testing(SAT-05), volume 3569 of Lecture Notes, pages: 203-215, 2005.
[34] D. Schuurmans, and F. Southey. Local search characteristics of incomplete SAT procedures. In Proc.AAAI-2000, pages: 297-302, AAAI Press, 2000.
[35] D. Schuurmans, F.Southey, and R.C. Holte. The exponentiated sub-gradient algorithm for heuristic Boolean programming. In Proc. IJCAI-01, pages: 334-341, Morgan Kaufman Publishers, 2001.
[36] B. Selman, H. Levesque, and D. Mitchell. A New Method for Solving Hard Satisfiability Problems. Proceedings of AAA'92,pages: 440-446, MIT Press, 1992.
[37] B. Selman, H.A. Kautz, and B. Cohen. Noise Strategies for Improving Local Search. Proceedings of AAAI'94, pages: 337-343. MIT Press, 1994.
[38] B. Selman and H.A. Kautz. Domain-Independent extensions to GSAT: Solving large structured satisfiability problems. In R.Bajcsy,editor, Proceedings of the international Joint Conference on Artificial Intelligence, volume 1, pages: 290-295. Morgan Kaufmann Publishers Inc., 1993.
[39] W. Spears. Adapting Crossover in Evolutionary Algorithms. Proc of the Fourth Annual Conference on Evolutionary Programming, MIT Press, pages: 367-384, 1995.
[40] J. Thornton, D.N. Pham, S. Bain, and V. Ferreira Jr. Additive versus multiplicative clause weighting for SAT. Proceedings of the Nineteenth National Conference of Artificial Intelligence (AAAI-04), pages: 191-196, 2004.
[41] D. Vrajitoru. Genetic programming operators applied to genetic algorithms. In Proceedings of the Genetic and Evolutionary Computation Conference, pages: 686-693,Orlando (FL). Morgan Kaufmann Publishers, 1999,
[42] C. Walsahaw. A Multilevel Approach to the Graph Colouring Problem. Tech.Rep. 01/IM/69, Comp.Math. Sci., Univ.Greenwich, London SE10 9LS, UK, May 2001.
[43] C. Walshaw: A Multilevel Algorithm for Forced-Directed Graph-Drawing. Journal of Graph Algorithms and Applications, 7(3) pages: 253-285, 2003.
[44] C. Walshaw and M. Cross. Mesh partitioning: A multilevel balancing and refinement algorithm. SIAM J. Sci. Comput., 22(1) pages: 63-80,2000.
[45] C. Walshaw. A Multilevel Approach to the Traveling Salesman Problem. Oper. Res., 50(5) pages: 862-877, 2002.
[46] C. Walshaw. A Multilevel Lin-Kerninghan-Helsgaun Algorithm for the Traveling Salesman Problem. Tech. Rep. 01/IM/80, Comp. Math. Sci., Univ. Greenwich, 2001.
[47] Z.Wu., and B. Wah. An efficient global-search strategy in discrete Lagrangian methods for solving hard satisfiability problems. In Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00), pages: 310-315, 2000.



VLSI Design

Edited by Dr. Esteban Tlelo-Cuautle

ISBN 978-953-307-884-7
Hard cover, 290 pages
Publisher InTech
Published online 20, January, 2012
Published in print edition January, 2012

This book provides some recent advances in design nanometer VLSI chips. The selected topics try to present some open problems and challenges with important topics ranging from design tools, new post-silicon devices, GPU-based parallel computing, emerging 3D integration, and antenna design. The book consists of two parts, with chapters such as: VLSI design for multi-sensor smart systems on a chip, Three-dimensional integrated circuits design for thousand-core processors, Parallel symbolic analysis of large analog circuits on GPU platforms, Algorithms for CAD tools VLSI design, A multilevel memetic algorithm for large SAT-encoded problems, etc.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Noureddine Bouhmala (2012). A Multilevel Approach Applied to Sat-Encoded Problems, VLSI Design, Dr. Esteban Tlelo-Cuautle (Ed.), ISBN: 978-953-307-884-7, InTech, Available from: http://www.intechopen.com/books/vlsi-design/a-multilevel-approach-applied-to-sat-encoded-problems

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen