Ant Colony Optimization approach for Solving FPGA routing with minimum Channel Width

Vinay Chopra (Asstt. Prof)
CSE Deptt.
DAV Institute of Engg. & Technology
Jalandhar, Punjab India
Vinaychopra222@yahoo.co.in

Amardeep Singh (Associate Prof*)*
CSE Deptt.
UCoE Punjabi University
Patiala Punjab India
amardeep_dhiman@yahoo.com

Abstract— **In this paper ANT colony optimization algorithm has been proposed to solve FPGA routing in FPGA design architecture with minimum umbers of tracks per channel. In our method geometric FPGA routing task is transformed into a Boolean satisfiability (SAT) equation with the property that any assignment of input variables that satisfies the equation specifies a valid route. The satisfiability equation is then modeled as Constraint Satisfaction problem. Satisfying assignment for particular route will result in a valid routing and absence of a satisfying assignment implies that the layout is unroutable. In second phase of this method ant colony optimization algorithm is applied on the Boolean equation for solving routing alternatives utilizing approach of hard combinatorial optimization problems. The ACO based solution to SAT is then compared with the other SAT solver algorithms such as zChaff and GRASP. The experimental results suggested that the developed ant colony optimization algorithm is taking fewer amounts of time and minimum channel width to route a FPGA chip**.

Keywords- FPGA routing, route based model, constraint satisfaction programming, and Boolean satisfiability, channel width

I. INTRODUCTION

Field-Programmable Gate Arrays (FPGAs) have revolutionized access to VLSI since its origin. FPGAs are programmable chips to implement any digital circuit, as divergent to custom ASICs that have a huge start up cost in terms of money and development time. However, even with huge popularity of FPGAs ,its contribution to the digital silicon market is not much to appreciate . This is because the benefits of FPGAs come at a high cost in terms of area, speed, and power [5]. FPGA's is the blend of both software and hardware-oriented systems. Circuits can be designed and implemented very quickly using recent design software with minimum cost of designing custom circuits. The whole CAD process that is used to realize a circuit in an FPGA consists of the logic optimization, Technology mapping, placement; routing. The outline structure of FPGAs consists of three parameters configurable logic blocks, I/O blocks and programmable routing [Wu, 97]. Among the above mentioned steps routing is the main step of the process because of the FPGA's area is committed to the interconnect, and the interconnection delays are greater than the logic delays of the designed circuit [9]. Because of , an efficient routing algorithm tries to shrink the total wiring area and the lengths of critical-path nets to perk up the performance of the circuit; and for this, the router needs the interconnect information of the target FPGA architecture. In this paper we have used ANT colony optimization approach for programmable FPGA routing shown in [Section 3]. *Boolean-based* routing is a recent approach that is used for solving routing problem in FPGA layout. Ant colonies are capable of finding shortest paths between their nest and food sources because the ants communicate indirectly by disposing traces of pheromone as they walk along a chosen path. Ants most likely prefer those paths possessing the strongest pheromone information. ACO has successfully solved several combinatorial optimization problems, including scheduling [6], vehicle routing [7], constraint satisfaction [8], and the quadratic assignment problem [9]. Boolean based routing problem can be represented as a large atomic

Boolean function, which is satisfiable if the layout is routable otherwise routing option is not considered i.e. any satisfying assignment to the variables of the routing Boolean function represents a legal routing solution [Nam, 99]. Recent advances in SAT solving algorithms (learning and non-chronological backtracking [Aloul, 01]) and efficient implementation techniques (e.g. fast implication engine) have dramatically improved the efficiency and capacity of solving the routing tasks in FPGA's. But there is still need to improve it so that FPGA routing task can be optimized. In this paper, we adapt an ACO algorithm to field programmable gate arrays (FPGAs). The ant colony optimization meta-heuristic is based on the natural foraging behavior of real ants and has been used to solve a wide spectrum of combinatorial optimization problems. There are lot of other algorithms that are used in the SAT based problems such as backtracking search, resolution based checker, integer linear programming based routing, BDD, recursive learning etc [Silva, 97]. In this paper, a new method based on the behavior of ants has been implemented for FPGA routing, which improvement over other SAT solvability algorithms for FPGA is routing.Academic research has considered a simplified version of island style FPGA architecture from Xilinx because of its highest share in the market as compared to Altera and Actel. In academia the most familiar simplifications made to the island style model are:

- Each logic block has 4 inputs pins and 1 output pin, and all logic blocks are alike as compared to commercial FPGAs which has logic blocks with different number of inputs, ranging from 3 to 7, and they provide two or more outputs.
- The C box is constructed with pass transistors rather than multiplexes for input connections as a result two or more tracks can be electrically connected via the input pin by revolving on individual switches in the C box. This is called input pin doglegs as compared to commercial FPGAs which implement the C box via multiplexes to save area, so only one track may be connected to the input pin and no input pin doglegs are possible. (fig 1).
- The wire segments span only one logic block before stopping. This means that all interconnections have to pass as many C boxes and S boxes as logic blocks there are between the two connecting points.

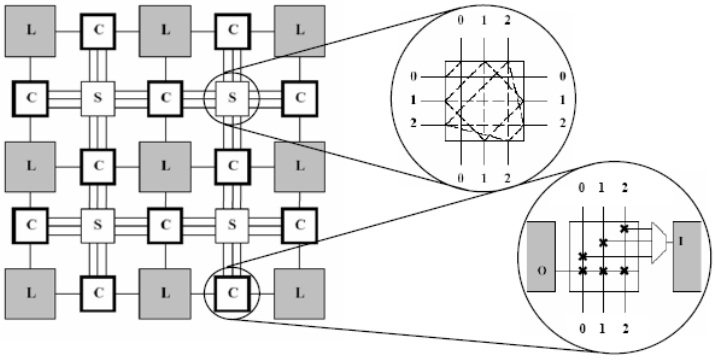

Fig 1: FPGA Model

II. FPGA ROUTING AS BOOLEAN SATISFIBILTY (SAT) PROBLEM

Boolean SAT-based routing [8, 9] transforms the FPGA routing task as an atomic Boolean function. The generated Boolean function is satisfiable (i.e. value 1) if and only if the design is routable. Thus, if the generated Boolean formula is satisfiable, then any satisfying assignment corresponds to a feasible routing of the channel; otherwise the channel is unroutable. In this paper routing problem which is solved by our method is illustrated in Fig. 2 [25, 27]. It consist of four nets labeled A, B, C, and D (Fig. 2a). The goal is to assign a track number to each net such that distinct nets are nonoverlapping both horizontally and vertically. Two matrices are designed for this routing problem; one represents the horizontal overlapping and second represents vertical overlapping (Fig. 2b). An exclusivity constraint is defined to insure that nets whose horizontal spans overlap are assigned to different tracks. This constraint is typically represented by a horizontal constraint graph and can be conveniently expressed as a Boolean function (Fig. 2d). The second constraint insures that, for any two nets with pins in the same column on opposite sides of the channel, the net associated with the top pin is assigned a higher track number. This constraint is defined by the vertical constraint graph (VCG) shown in Fig. 2c, which is also represented as a Boolean function V of Fig. 2e. The conjunction (AND) of these two functions is the complete routability constraint Boolean function R for the channel (Fig. 2f). Any track number assignment that makes R=1 corresponds to a feasible routing solution. Two possible solution are shown in Fig. 2g.

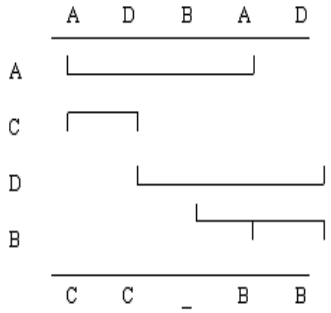

Fig. 2.a Channel to be routed

	A	B	C	D
A	0	1	1	1
B	0	0	0	0
C	0	0	0	1
D	0	1	0	0

	A	B	C	D
A	0	1	1	0
B	0	0	0	0
C	0	0	0	0
D	0	1	1	0

Fig. 2.b Matrix representation of Channel routing problem

$E = (A \neq C) \Lambda (A \neq D) \Lambda (A \neq B) \Lambda (C \neq D) \Lambda (D \neq B)$

Fig. 2.c Exclusivity Constraint

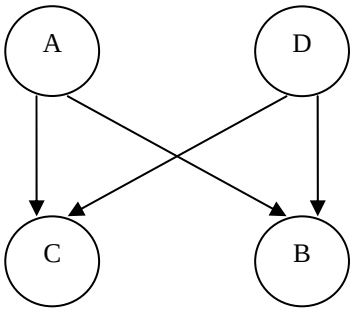

Fig. 2.d Vertical Constraint Graph (VCG)

$V = (A>C) \Lambda (A>B) \Lambda (D>C) \Lambda (D>B)$

Fig. 2.e Vertical Ordering Constraint

$R = E \Lambda V$

Fig. 2.f Channel routability constraint

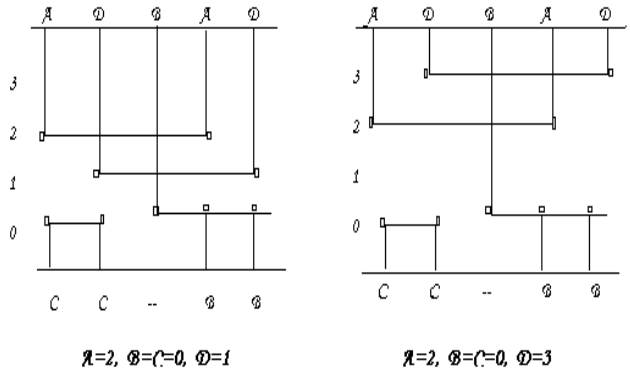

Fig. 2.g Two feasible solutions

Fig. 2. Boolean SAT modeling of channel routing problem.

III. ANT COLONY OPTIMIZATION

Ant Colony Optimization (ACO) is a population-based, general search technique for the solution of difficult combinatorial problems which is inspired by the pheromone trail laying behavior of real ant colonies. The ant colony optimization algorithm (ACO) is a probabilistic technique for solving computational problems which can

be concentrated to finding good paths through graphs. They exchange information indirectly by depositing pheromones, all detailing the status of their "work". The information exchanged has a local scope, only an ant located where the pheromones were left has a notion of them. This system is called "Stigmergy" and occurs in many social animal societies. The advantages of Ant colony optimization are their inherent parallelism, positive feedback accounts for rapid discovery of good solutions, Efficient for TSP and similar problems, used in dynamic applications .Ant colony optimization algorithms have been applied to many combinatorial optimization problems, ranging from quadratic assignment to fold protein or routing vehicles and a lot of derived methods have been adapted to dynamic problems in real variables, stochastic problems, multi-targets and parallel implementations. They have an advantage over simulated annealing and genetic algorithm approaches of similar problems when the graph may change dynamically; the ant colony algorithm can be run continuously and adapt to changes in real time. In the first phase the ACO algorithm initialize the pheromone matrix by setting each pheromone entry to a starting value $\tau_{\text{init}} > 0$. For problems with an item x encoded pheromone matrix, the pheromone entries on the diagonal are set to 0. *m* ants generate solutions $\pi_0, \ldots \pi_{m-1}$ in each iteration of the algorithm. An ant generates a solution by building a sequence of local decisions. Each decision is made randomly according to the following probability distribution over the r unchosen items in selection set *S*:

$$\forall_j \in S : p_{ij} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{z \in S}\tau_{iz}^{\alpha}\eta_{iz}^{\beta}} \quad (1)$$

Where α and β are used to establish the relative weight of pheromone values and heuristic values. Initially, the selection set *S* contains all items. After each decision, the selected item is removed from *S* and every solution is evaluated according to the individual objective function. After *m* solutions have been constructed, the solution qualities are compared to determine the best solution π^* of the current iteration.

The pheromone matrix is then restructured in two steps:

(1) *Evaporation*: Each pheromone values in the matrix are reduced by a relative amount: $\forall i, j \in [0, n-1] : \tau_{ij} \mapsto (1-\rho)\tau_{ij}$.

(2) *Intensification*: The pheromone values along the best solution π^* are increased by an absolute amount: $\forall i \in [0, n-1] : \tau_{i\pi^*(i)} \mapsto \tau_{i\pi^*(i)} + \Delta$.

The ACO algorithm executes a number of iterations until the following specified stopping criterions has been met,

a) A predefined maximum number of iterations have been executed.

b) A specific level of solution quality has been reached e.g. minimum channel width, less CPU time.

c) The best solution remains constant over a certain number of iterations.

IV. THE DESIGNED ALGORITHM : ANT BASED SATISFIABILTY DETAILED ROUTING

In order to apply ACO algorithms [3] to SAT instances of FPGA routing to find the shortest paths, in the first step a global router is invoked which assigns each net a path through regions of the routing fabric composed of several S-blocks and C-blocks in the FPGA and after that assign to each net a vector of Boolean variables for each region of routing fabric through which it passes. This vector of variables encodes each possible decision for how to assign the net to physical resources in the region. In the next step a Boolean connectivity constraint equation is formulated that ensures that each net actually connects through a set of legal, contiguous routing resources from source to sink. This function is essentially a characteristic function over the encoding of net-to-resource assignments that is “1” for connected paths. After that a Boolean exclusivity constraint equation is constricted that ensures that no two electrically distinct nets try to use a common routing resource in any block of the routing fabric.

This function is essentially a characteristic function over the net-to-resource assignments that is “1” for sets of noninterfering paths and then the final Boolean routability function is designed that determines all the possible

routes for these nets. To apply ACO algorithms to the FPGA routing problem, it is convenient to see it as a combinatorial optimization problem with the capability of being represented on a graph. To formulate the solution, these two classes of routing violations must be prevented. First, each two-point connection has a connectivity constraint associated with it whose purpose is to ensure that the connection makes a contiguous path through the channel. Also, exclusivity constraints are needed for all pairs of connections of distinct nets that interact: that is, if the interval defined by their endpoints overlaps in one or more C-blocks. The exclusivity constraint between any two bit vectors is just the vectored exclusive-or between them. This ensures that at least one bit differs in their respective vectors.

The steps that are used for our ACO framework are illustrated below.
Step 1. Initialize the pheromone values and generate all the SAT problems and let *iterCount*=2^M (where M is the number of nets in an FPGA circuit).The pheromone information is encoded in an MxM pheromone matrix.
Step 2. Do until *iterCount* is not zero.
Step 3. Take a subset of the SAT problems and generate ants to represent each of the chosen SAT problems.
Step 4. Simulate the ant movement.take a subset of ant and insert them into the queue [2].

```
insertASubsetOfAntsInQueue(&q, A);
while(!isEmptyQueue(q))
{
//obtain the next ant to simulate
ant = getNextAntF romQueue(q);
if(thisAntAchievedGoal(ant))
markT hisAntDead(ant);
else if(thisAntReachedInput(ant))
replaceBackAnt(ant);
else
simulateMovement(q, ant);
}
```

Step 5. Generate test pattern and perform logic simulation.
The test pattern can be generated by generating the permutations based upon the number of nets. The permutations are used as inputs for the SAT problem and are solved for those values. The satisfying values represent the valid routes.
Step 6. Update pheromone strengths.
The pheromone matrix is updated by multiplying the pheromone values with an evaporation factor $\rho<1$.
Step 7. If all the SAT problems are solved, then terminate. Otherwise, decrement *iterCount* and then go to step 2.

V. EXPERIMENTAL RESULTS

We have tested the effectiveness of ANT Colony algorithms for the circuits shown in Table 1. In our approach it is assumed that global routing and placement of circuits are given. In our experiments we have varied the S block flexibility from 3 to 3*w and C block flexibility was set to F_c=W so that CLB pin can connect to any number of tracks. It shows ANT Colony routing results for reasonably large circuits reflecting practical FPGAs. The benchmarks are from [24].The experimental work was carried out by writing a program in C++ and running on SUN Ultra SPARC-III processor under Sun Blade workstation architecture. Xilinx 4000 series architecture model was employed for this work. The experiment result shows [see Fig. 4] that our approach of solving the FPGA routing using ant colony optimization is improvement over other SAT based FPGA routing solvers. Experiments shows that ANT Colony algorithm is able to solve the FPGA routing problem with minimum number of tracks per channel which is the improvement over the other satisfiability based detailed routers. Also routing conflicts among different routing solutions can be removed by representing routing solutions parallel using Ants parallelism techniques. For K2 Circuit, ACO has taken only 9 tracks in channel as compared to Grasp and Zchaff which has taken 12 and 10 tracks respectively. Different alternatives for particular solution are represented parallel to solve the conflicts among different solutions. The results are compared with different Boolean solvers such as GRASP, Zchaff.

TABLE 1: COMPARISON ON THE BASIS OF CHANNEL WIDTH AMONG GRASP , ZCHAFF AND ACO

circuits	Variables	clauses	GRASP(W)	Zchaff(W)	ACO(W)
asymmt	2604	32450	6	6	5
Alu2	38567	115757	8	7	6
apex7	51108	259880	5	6	4
C8880	4623	72021	7	6	5
Example2	3603	41023	6	7	6
K2	13176	445254	12	10	9
Too_large	3972	60432	7	6	5
Vda	7436	168604	10	9	9

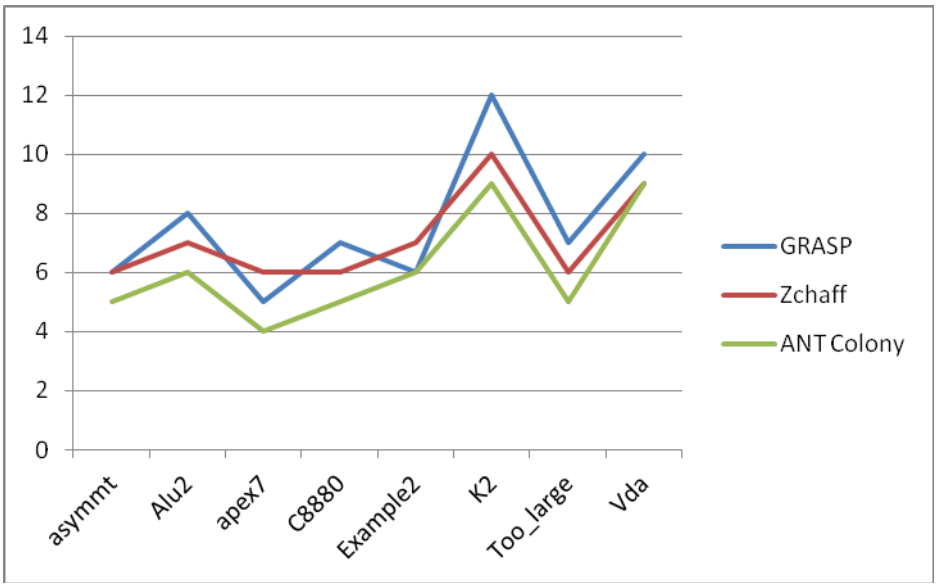

Fig1: Comparison on the basis of Channel width among GRASP, Zchaff and ACO

VI. CONCLUSION

In this paper ACO algorithm is implemented to improve the performance of the FPGA routing .Our results have shown that the ACO algorithm by taking less CPU time, which is an optimal solution and have taken minimum wire length to route FPGA chips. Our algorithm works as a collection of agents work collaboratively to explore the different routes. A stochastic decision making strategy is proposed in order to combine global and local heuristics to effectively conduct this exploration. In all the circuits which are taken for our experiment ACO has routed the channel with minimum numbers of tracks. In all the circuits that have been considered in our study ACO has performed better and has routed the circuit with minimum channel width as compared to other classical algorithms such as GRASP, Zchaff. Our algorithm is more effective in finding the near optimal

solutions and scales well as the problem size grows. It is also shown that with substantial less execution time, the proposed method achieves better solutions than the popularly used zChaff and GRASPS satisfiabilty solvers.

REFRENCES

[1] Eliezer L. Lozinskii, Impurity"Another phase transition of SAT, Journal on Satisfiability", Boolean Modeling and Computation, vol. 1, pp. 123-14. January 2006.

[2] R. Sethuram, M. Parashar " Ant Colony Optimization and its Application to Boolean Satisfiability for Digital VLSI Circuits" Advanced Computing and Communications, International conference 10.1109/ADCOM.2006.4289945. January 2007.

[3] Alaya, I. Solnon, C. Ghedira, K" Ant Colony Optimization for Multi-Objective Optimization Problems" Tools with Artificial Intelligence, 2007. ICTAI Patras volume: 1 pages 450-457 Oct 2007.

[4] Hong-qi Li Li "A Novel Hybrid Particle Swarm Optimization Algorithm Combined with Harmony Search for High Dimensional Optimization Problems" Intelligent Pervasive Computing, International Conference On page(s): 94-97 Oct 2007.

[5] Gi-Joon Nam and Karem A. Sakallah." Detailed Routing of Complex FPGAs Via Search-Based Boolean SAT", in symposium on Field Programmable Gate Arrays, Monterey, CA, 167-175. December 2004

[6] Gang Wang Wenrui Gong Ryan Kastner," System Level Partitioning for Programmable Platforms Using the Ant Colony Optimization", pp 150-202, August 2004

[7] J. P. M. Silva and K.A. Sakallah. "GRASP--A New Search Algorithm for Satisfiability", In Proc. ACM/IEEE ICCAD. pp 156- 168 , Nov. 1997

[8] P. K. Chan et.al., "On Routability Prediction for Field Programmable Gate Arrays", In Proc. IEEE DAC.pp.190-201. , June 1993

[9] R. G. Wood and R. A. Rutenbar, "FPGA Routing and Routability Estimation via Boolean Satisfiability," IEEE Trans. VLSI Systems, pp. 222-231. June 1998

[10] Neumann, F. and Witt, C., "Runtime Analysis of a Simple Ant Colony Optimization Algorithm", Electronic Colloquium on Computational Complexity (ECCC), Report No. 84, July 2006.

[11] R. Bayardo Jr. and R. Schrag, "Using CSP Look-Back Techniques to Solve Real World SAT Instances," Proc. 14th Nat'l Conf. Artificial Intelligence, pp. 203-208, December 1997.

[12] Stützle, T. and Dorigo M., "A Short Convergence Proof for a Class of ACO Algorithms", IEEE Transactions on Evolutionary Computation, 6 (4), November 2002

[13] Rizzoli,A.E., Oliverio F., Montemanni R. and Gambardella L.M.. "Ant Colony Optimization for vehicle routing problems: from theory to applications", Technical Report IDSIA-15-04, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, September 2004.

[14] L. Andrew C. "Field Programmable Gate Array Logic Synthesis using Boolean Satisfiability", M.Tech Thesis, Computer Science & Electrical Engg. Deptt., University of Toronto.

[15] Gang Wang Wenrui, Gong Ryan Kastner, "System Level Partitioning for Programmable Platforms Using the Ant Colony Optimization", September 2004.

[16] Marchiori, E and Rossi, C, Gottlieb, J. "Evolutionary Algorithms for satisfiability Problem", In Genetic and Evolutionary computation Conference. pp 170-175 May 2002

[17] Armin Biere and Carsten Sinz. "Decomposing SAT problems into connected components", Journal on Satisfiability, Boolean Modeling and Computation, 2:191–198, June 2006.

[18] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik, "Engineering an efficient SAT solver," in Design Automation Conference, June 2001, pp. 530–535.

[19] V. Betz and J. Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," presented at International Workshop on Field Programmable Logic and Applications, London, July 1997.

[20] Geem, Z.W., J.H. Kim and G.V. Loganathan,."A new heuristic optimization algorithm: Harmony search. Simulation", page 76: 60-68. June 2001

[21] Niklas E´en and Niklas S¨orensson. "An extensible sat solver", In Proceedings of the Sixth International Conference on Theory and Applications of Satisfiability Testing, LNCS 2919, pages 502–518, 2003.

[22] Xiao Yu Li." Optimization Algorithms for Minimum Cost Satisfiability", Ph.d Thesis, Computer Science & Engg. Deptt., Raleigh North California 2004.

[23] Marchiori, E and Rossi, C, Gottlieb, J. "Evolutionary Algorithms for satisfiability Problem", In Genetic and Evolutionary computation Conference pages 413-416 October 2002

[24] http://www.eecg.toronto.edu/~vaughn/challenge/challenge.html

[25] G.-J. Nam, K.A. Sakallah, and R.A. Rutenbar, "A New FPGA Detailed Routing Approach via Search-Based Boolean Satisfiability," IEEE Trans. Computer-Aided Design, vol. 21, no. 6, June 2002.

[26] Gi-Joon Nam, Member, IEEE, Karem A. Sakallah, Fellow, IEEE, and Rob A. Rutenbar, Fellow, IEEE "A New FPGA Detailed Routing Approach Via Search-Based Boolean Satisfiability" IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, Vol. 21, No. 6, June 2002

[27] Saveena, Vinay Chopra and Dr. Amardeep Singh "Optimized FPGA Routing using Soft Computing" published in International Journal of Computer Application No.8 article 2 on 2010

AUTHORS PROFILE

Vinay Chopra is working as a Astt. Prof.(CSE) at DAVIET jalandhar. He has received his bachelor degree in B.Sc Computer Science from G.N.D.U., M.Sc (Maths) from G.N.D.U, M.E. (Software Engg) from Thaper University. And Pursuing PhD from Punjabi University Patiala .He is life member of CSI and Punjab Science Congress.

Amardeep Singh received his BTech degree in Electronics and Communication in 1995 from MIT and his MTech degree in Computer Science and Engineering from Punjabi University, Patiala. He received his PhD degree in VLSI Testing using DNA and Quantum computing Algorithms in 2006 from Thapar University, Patiala. Now he is holding an academic position as Reader in UCoE, Punjabi University, Patiala, India. His main interest areas include nanocomputing, embedded systems, and nonconventional algorithms.