Ant Colony Based Approach for Solving FPGA routing

Vinay Chopra, Amardeep Singh[2]

**[1] Asstt. Prof. CSE Department DAVIET Punjab Technical university ,
Jalandhar, Punjab , India**

**[2] Associate Prof. CSE Department UCoE, Punjabi University
Patiala, Punjab India**

Abstract

This paper is based on an ant colony optimization algorithm (ASDR) for solving FPGA routing for a route based routing constraint model in FPGA design architecture. In this approach FPGA routing task is transformed into a Boolean Satisfiabilty (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, which helps in reducing procedural programming. 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 step ant colony optimization algorithm is applied on the Boolean equation for solving routing alternatives utilizing approach of hard combinatorial optimization problems. The experimental results suggest that the developed ant colony optimization algorithm based router has taken extremely short CPU time as compared to classical Satisfiabilty based detailed router (SDR) and finds all possible routes even for large FPGA circuits.

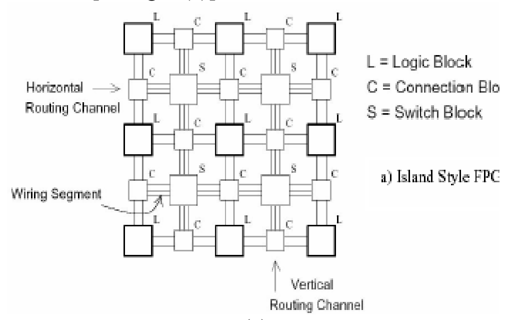
Keywords: *Ant colony optimization, FPGA Routing, Ant Satisfiabilty based detailed routing (ASDR), Boolean Satisfiabilty SAT.*

1. Introduction

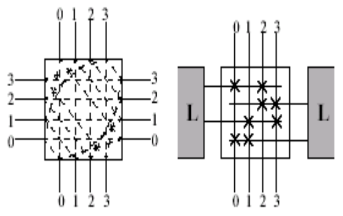
A variety of devices is currently available for developing and implementing digital systems. Usually, a designer can choose from a wealth of software-oriented devices like general-purpose-processors, micro-controllers, digital signal processors, or application-specific instruction set processors (ASIPs). On the other hand, hardware devices—so-called application-specific integrated circuits (ASICs)—are available, which are designed for predefined processing tasks. By providing programmable selection of alternate logic and routing structures, field-programmable gate arrays can be considered to be located at the intersection of software and hardware-oriented systems. Using modern design software, circuits can be designed and implemented very rapidly. There are many different FPGA architectures available from various vendors including Altera, Xilinx [XILINX, 01], Actel, Lucent, Quick Logic and Cypress. The layout structure of these FPGAs depends upon three parameters configurable logic blocks, I/O blocks and programmable routing [Wu, 97]. Our main consideration in this paper is on programmable FPGA routing shown in [Section 2]. *Boolean-based* routing is a recent approach that is used for solving routing problem in FPGA layout. 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. Quantum mechanics properties are used to solve FPGA routing by considering it as Satisfiability problem and then quantum search algorithms are applied on it [see Subsection 4.1]. Quantum search algorithms are much efficient than the classical algorithms in solving the search based problems, which take less number of iterations than other algorithms [see Section 5] [Grover, 96]. Many search style solutions have been proposed for SAT, the best known being variations of the *Davis-Putnam* procedure [Aloul, 01] which is based on a backtracking search algorithm .The other algorithms that are used in the SAT based problems are backtracking search, resolution based checker, integer linear programming based routing, BDD, recursive learning etc [Silva, 97]. A new approach for FPGA routing, which is based on the Ant colony optimization which is improvement over other SAT solvability algorithms for FPGA routing is illustrated in this paper. The ant colony optimization meta-heuristic is inspired from the natural foraging behavior of real ants and has been used to find good solutions to a wide spectrum of combinatorial optimization problems. Classically many search style solutions have been proposed for SAT, the best known being variations of the *Davis-Putnam* procedure [Aloul, 01] which is based on the backtracking search. The other algorithms that are used in the SAT based problems are backtracking search, resolution based checker, integer linear programming based routing, BDD, recursive learning etc [Silva, 97]. A new approach for FPGA routing is illustrated in this paper, which is based on the ANT colony optimization.

2. FPGA Layout

The standard island style FPGA architecture Xilinx *4000* [XILINX, 01] is used in this experiment [see Fig. 1(a)]. This architecture consists of two-dimensional array of configurable logic blocks *CLB*s, Connection blocks *C blocks* and switching blocks *S blocks* [Chan, 93]. Each CLB marked *L* in [Fig. 1(a)] contains the combinational and sequential logic that implements the functionality of a circuit. *C* and *S* blocks contain programmable switch form the routing resources [29]. *C* blocks connect CLB pins to channels via programmable switches. *S* blocks are surrounded by *C* blocks and allow signals to either pass through [5].The routing capacity of a g iven FPGA architecture is expressed by three parameters W, F_c, F_s [Iwama, 02]. The channel width W is the number of tracks in a vertical or horizontal channel. The C block flexibility F_c is defined to be the number of tracks that each logic pin can connect to [28]. The S block flexibility F_s denotes the number of other tracks that each wire segment entering an S blocks. Each wire segment entering this S block can connect to one track on each of the other three sides, hence F_s=3[see Fig. 1(b)]. Each logic pin can be connected up t o any two tracks in the C block, thus F_c=2[In Fig. 1(c)]



(a)



(b) $F_s = 3$ (c) $F_c = 2$

Fig 1: Island style FPGA model

2.1 Boolean SAT based FPGA detailed routing formulation

Boolean SAT-based routing [8, 9] is the approach in which the routing task is transformed into an atomic Boolean function which is satisfiable (has an assignment of input variables such that the generated function evaluates to constant "1") if and only if the design is routable. Any satisfying assignment to the binary variables of the Boolean function gives a legal routing solution. In this method Boolean routability function R (X) [28], has been formulated where X is a suitable Boolean vector of binary variables that encode the track number for each two-pin connection, can be expressed as the conjunction $R(X) = L(X) \wedge E(X)$. Liveness constraint function L(X) guarantees that at least one global route alternative per two-pin connection should be chosen as a final legal routing solution. Exclusivity constraint function E(X) ensures that electrically distinct nets with overlapping vertical or horizontal spans in the same channel are always assigned to different track [5].

Boolean variables represent each routing alternatives of a netlist that represent all of the detailed routes [see Fig.3 (b)]. In this problem for *NET A*, there are only three possible detailed routes indicated by the three Boolean variables *AR0, AR1, AR2 [28]*. NET B and C are designed with their routes and the corresponding variables. A particular route is validated as the routing solution if its corresponding Boolean variable is assigned the logic value 1[see Fig.3 (b)]. For a netlist with n two-pin connections, Liveness constraints yield a set of n CNF clauses, each containing F_c positive literals. A single Boolean function represents all Liveness and exclusivity constraints which gives the routability of a particular netlist.

$$R(X) = [Liveness\ (n) \wedge Exclusive\ (n)]\ for\ all\ n \in all\ Nets \quad r \in all\ \mathrm{Re}sources$$

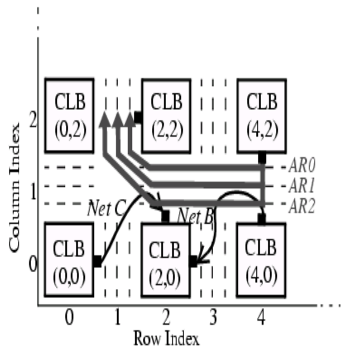
Where X is a vector of Boolean variables that represent the possible detailed routes for each of the nets.

NET A route Boolean variable (AR0, AR1, AR2)
NET B route Boolean variable (BR0, BR1, BR2)
NET C route Boolean variable (CR0, CR1, CR2)
(a)

$$\text{Liveness (A)} = (AR0 \vee AR1 \vee AR2)$$
$$\text{Liveness (B)} = (BR0 \vee BR1 \vee BR2)$$
$$\text{Liveness (C)} = (CR0 \vee CR1 \vee CR2) \quad (1)$$

(b) Liveness constraints

$$\text{Exclusively (Resource (4, 1, 0))} = \overline{AR0} \vee \overline{BR0}$$



$$\text{Exclusively (Resource (4, 1, 1))} = \overline{AR1} \vee \overline{BR1}$$
$$\text{Exclusively (Resource (4, 1, 2))} = \overline{AR2} \vee \overline{BR2}$$
$$\text{Exclusively (Resource (2, 1, 0))} = \overline{AR2} \vee \overline{CR0}$$
$$\text{Exclusively (Resource (2, 1, 1))} = \overline{AR2} \vee \overline{CR1}$$
$$\text{Exclusively (Resource (2, 1, 2))} = \overline{AR2} \vee \overline{CR2} \quad (2)$$

(c) Exclusivity constraints.

Fig 3: Global routing configuration for NETS A, B and C and three possible detailed routes for NET A.

3. Ant Colony Optimization

Ant colony optimization is encouraged by research on the behavior of ant colony [12, 13]. Ant colonies are skilled of finding shortest paths between their nest and food sources because the ants correspond indirectly by disposing traces of pheromone as they walk along a particular path. The subsequent ants follow the paths which posses the strongest pheromone information, since ants on short paths are quicker, pheromone traces on t hese paths are amplified very repeatedly. Pheromone information is permanently reduced by evaporation, which diminishes the influence of formerly chosen unfavorable paths. This combination focuses the search process on short, favorable paths. Dorigo et al. introduced a met heuristic known as ant colony optimization (ACO) which is inspired by this biological model. Artificial ants are a generalization of real ant behavior. They are sometimes enriched with some capabilities which do not find a natural counterpart. To mimic this behavior in an ACO framework, artificial ants leave numeric information called artificial pheromone trail (APT) [27]. This stigmergetic form of communication using APT among individual ants guides all ants to obtain the global optimal solution efficiently.

3.1 The Proposed algorithm: ASDR (Ant based Satisfiabilty Detailed Routing)

This Proposed algorithm consists of the following steps:

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 [27]. Take a subset of ant and insert them into the queue.

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

Step 5. Generate FPGA routing solution and perform logic simulation.

The Routing 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 ρ<1.

Step 7. If all the SAT problems are solved, then terminate. Otherwise, decrement *iterCount* and then go to step 2.

The overall processing flow of the ASDR is given by the following Flow chart

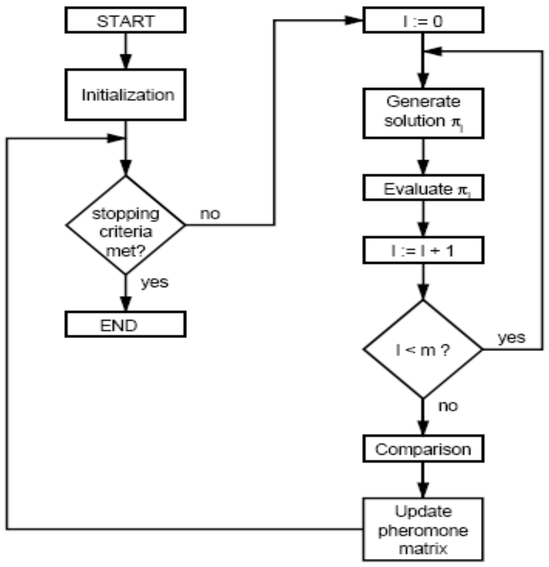


Fig 4: Processing Flow Chart of ASDR

4. Experimental Results

ANT Colony algorithms are tested for the circuits shown in Table 1. We have 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. The benchmarks are from [26], Firstly ASDR algorithm assigns each net to a sequence of routing resources consisting of C-block and S-blocks and then divides each net into horizontal and vertical segments, and sort them by channel. After that for each horizontal channel *j*, apply the left-edge channel routing algorithm [9, 24] to determine the minimum number of tracks required to route that channel Set. In the second last step for each vertical channel *I,* the routing constraint functions are formulated and CNF formula for that constraint has been generated. And lastly invoke ASDR on the CNF "clause database" generated in step. Using this approach all the benchmark circuits could be routed in about 1 minute. ANT Colony provides routing solutions very quickly and cheaply even with large-scaled circuits without using intensive computational resources. It also avoids the net-ordering problem by considering all the nets simultaneously, and finds a solution if there exists any. In this approach number of FPGA routing architecture can be easily accommodated via proper Boolean function manipulations. The results have shown that using our method the circuits' diffeq, ex1010 are routed easily by taking less CPU time. The following table gives the results that are calculated with our experiment, the second column shows the number of tracks that are calculated for each circuit and it is the minimum length that is calculated with this algorithm for each circuit. The last column depicts the actual time taken in sec to route the circuits and it has been compared with SDR [24] which is inspired by the same approach but to solve the Boolean equations GRASP has been used. It has been shown in the graph that ASDR has taken less CPU time as compared with SDR.

Table 1: Experimental results with ASDR

circuit	track	Fs	Var.	clause	CNF(s)	SAT(s)
alu4	10	7*3	38567	115757	12	4.8
apex4	12	7*3	51108	259880	22.5	12.3
clma	12	9*3	238944	750489	465.0	56.2
des	10	7*3	32415	92507	9.5	3.6
diffeq	7	5*3	24330	65874	7.5	3.2
dsip	8	6*3	20451	58332	5.2	2.5
ex1010	11	8*3	120076	388669	110.3	20.4
ex5p	12	9*3	38472	122148	11.8	5.8
misex3	11	8*3	39042	119507	11.0	5.9
seq	11	8*3	703636	337448	39.0	14.2
spla	13	10*3	172228	830077	230.1	44.2

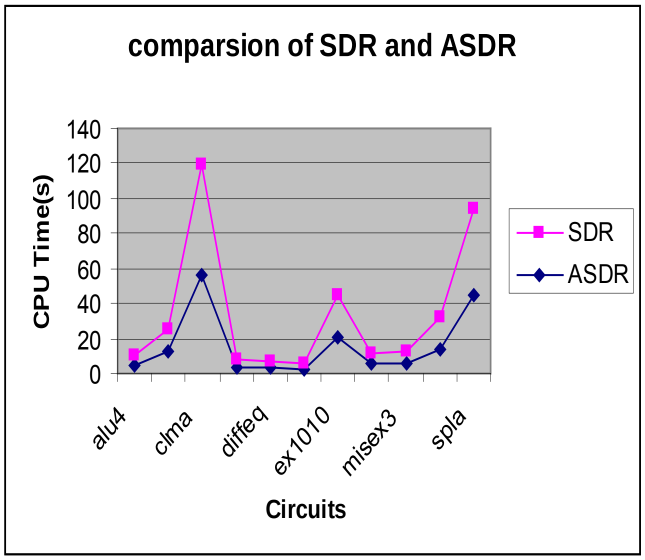


Fig3: Comparison of SDR and ASDR for different circuits

5. Conclusion

We have tried to improve the performance of the FPGA routing by solving it using ACO algorithm. 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. 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. We have compared our algorithm with the simple SAT solver and the results have shown that it has taken less CPU time as compared to simple SAT solver for every circuit that have been taken for our experiment. The results have shown that our ASDR algorithm has routed the dsip, diffeq circuits in less CPU time and the second column in the table have shown the minimum channel width needed to route the circuits and the results have shown that dsip, diffeq circuits are routed with minimum channel width with this approach. The sixth column has shown the time taken by our approach for converting the Boolean equation in CNF form and last column depicts the actual time taken(in sec) by ASDR to solve the Boolean equation corresponding to the circuits taken in this study. Also the ASDR has been compared with the previously designed SDR and the results have clearly depicted that for all the circuits, ASDR has routed them with less CPU time as compared to SDR.ANT Colony finds a feasible minimum FPGA routing architecture for a given circuit. In either case, if the routing solution exists. ANT Colony provides routing solutions very quickly and cheaply. Experimental results suggest that this approach is quite suitable for large-scaled FPGA applications

6. References

[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 F ield 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 F ield 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 pa ges 413-416 October 2002.

[24] Gi-Joon Nam, Karem A. Sakallah and Rob A. Rutenbar "Satisfiability-Based Detailed FPGA Routing" published in VLSI Design, 1999. Proceedings. Twelfth International Conference On 7-10 Jan 1999 pp(**s**)**:** 574 – 577.

[24] Ganesh K. Venayagamoorthy and Venu G. Gudise "Swarm Intelligence for Digital Circuits Implementation on Field Programmable Gate Arrays Platforms" Proceedings of the 2004 N ASA/DoD Conference on Evolution Hardware (EH'04) by IEEE 0-7695-2145-2/04 January 2004.

[25] Maidee, Cristinel Ababei, and Kia Bazargan, "Timing-Driven Partitioning-Based Placement for Island Style FPGAs" Pongstorn, IEEE Transactions on C omputer-Aided Design Of Integrated Circuits And Systems, Vol. 24, No. 3, March 2005.

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

[27] Rajamani Sethuram, Manish Parashar "Ant Colony Optimization and its Application to Boolean Satisfiability for Digital VLSI Circuits" Advanced Computing and Communications, 2006. ADCOM 2006. International Conference on 20-23 Dec. 2006 pp- 507.

[28] Gi-Joon Nam, Fadi Aloul, Karem Sakallah and Rob Rutenbar" A Comparative Study of Two Boolean Formulations of FPGA Detailed Routing Constraints" Computers, IEEE Transactions on June 2004 pp 688 – 696.

[29] 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.

Vinay Chopra is working as a A stt. 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 E ngineering from Punjabi University, Patiala. He received his PhD degree in VLSI Testing using DNA and Q uantum computing Algorithms in 2006 f rom Thapar University, Patiala. Now he is holding an ac ademic position as Reader in UCoE, Punjabi University, Patiala, India. His main interest areas include nanocomputing, embedded systems, and nonconventional algorithms.