International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:4, 2009

Comparative study of Ant Colony and Genetic Algorithms for VLSI circuit partitioning

Sandeep Singh Gill, Rajeevan Chandel, and Ashwani Chandel

***Abstract*—**This paper presents a comparative study of Ant Colony and Genetic Algorithms for VLSI circuit bi-partitioning. Ant colony optimization is an optimization method based on behaviour of social insects [27] whereas Genetic algorithm is an evolutionary optimization technique based on Darwinian Theory of natural evolution and its concept of survival of the fittest [19]. Both the methods are stochastic in nature and have been successfully applied to solve many Non Polynomial hard problems. Results obtained show that Genetic algorithms out perform Ant Colony optimization technique when tested on the VLSI circuit bi-partitioning problem.

***Keywords*—**Partitioning, Genetic Algorithm, Ant Colony Optimization, Non Polynomial Hard, Netlist, Mutation.

I. Introduction

THE advancement in VLSI semiconductor technology has led to a phenomenal development in Electronics Industry, leading to more chip complexity and higher integration. However as the chip density increases numerous issues like ease of design, testing, increased delay, interconnect area optimization arise which need to be handled at the design stage. Improved physical design tools are necessary to handle these issues. Circuit net list partitioning is an important step in VLSI physical design. This involves the breakup of a circuit into smaller parts for ease of design, layout and testability. The main objectives of circuit partitioning can include minimization of number of interconnections between the partitions, minimization of delay between partitions, power consumption optimization and Area optimization [12].

In the present work the versatility of the techniques of Ant colony and Genetic algorithms in solving the bi-partitioning problem is evaluated. Multiple partitions can be obtained by recursively applying the Method on obtained partitions. Recursive bi-partitioning has been rated better than direct multiway partitioning [15].

Efficient easily applied algorithms for optimal clustering to minimize delay in digital networks have been developed by Lawler et al.[1]. Kernighan and Lin [2] propose a heuristic for two way partitioning which is the first interactive algorithm based on swapping of vertices. A more practical model based on hyper graphs is proposed, but was inefficient due to time complexity [3]. A new data structure bucket list for cell gains and proposed cell move with better time complexity is proposed [4]. Krishnamurthy [5] modified [4] to introduce the concept of look ahead to choose the cell move.

Various multiway partitioning algorithms are proposed by modifying [4] [5] and developing appropriate data structures [6], top down clustering and iterative primal-dual approach [7], dual intersection graph representation and ratio cut metric [8]. Ariebi and Vanneli [9] describe the application of Tabu search to circuit partitioning problem.

A Genetic Algorithm based evolutionary approach for circuit partitioning giving a significant improvement in result quality is proposed [10]. Comparative evaluation of Genetic algorithm and Simulated annealing is done with Genetic algorithm giving better results [11]. A new hyper graph partitioning algorithm hMetis is proposed, giving faster and better cutsize [13].

Areibi [14] discusses the implementation issues for applying memetic algorithm for VLSI physical design. A multi objective hMetis partitioning for simultaneous cutsize and circuit delay minimization is proposed [16].

Various algorithms using different optimization techniques are developed for SoC and hardware software partitioning [17] [18].

Banos et al. [20] give a parallel evolutionary algorithm where parallelism improves the solutions found by corresponding sequential algorithm. Sait et al. [21] prepare a new heuristic called PowerFM which modifies FM algorithm and also considers minimization of power consumption.

Kolar et al. [22] obtain good results by using simulated annealing for two way partitioning of a circuit. Ghafari et al. [24] focus on minimizing the dynamic and sub threshold leakage power in CMOS circuits. An algorithm for application partitioning on programmable platforms using Ant Colony optimization is proposed [26]. Ariebi and Ali [28] develop an embedded computing system based on FPGA chip to accelerate the FM algorithm for circuit partitioning with excellent results. Comparative study of Evolutionary model and clustering methods in circuit partitioning is given [29].

From the review of literature it is found that various researchers have applied numerous optimization techniques for the partitioning optimization problem with mixed results.

In the present work two excellent optimization methods of Ant Colony and Genetic algorithms have been applied to the above problem.

Sandeep Singh Gill is Asstt. Prof. (Deptt. of ECE) at Guru Nanak Dev Engg. College, Ludhiana, Punjab, India. (e-mail: ssg270870@yahoo.co.in).

Rajeevan Chandel is working as Asstt. Prof. & Head of Deptt. of ECE at National Institute of Technology, Hamirpur, India.

Ashwani Chandel is working as Asstt. Prof. (Deptt. of Electrical Engg.) at National Institute of Technology, Hamirpur, India.

II. Problem Formulation

Problem of VLSI circuit partitioning is non polynomial hard and cannot be effectively solved by deterministic algorithms. Ant colony and Genetic algorithms belong to probabilistic and iterative class of algorithm and are stochastic in nature. Therefore they can be effectively applied for VLSI circuit partitioning.

The problem involves dividing the circuit net list into two subsets and some of the connections (edges) are also cut. The number of edges belonging to two different partitions is the cost of a partition. The objective function captures the interconnection information and partitioning solution is optimized with respect to interconnection between the partitions with the constraint of forming balanced partitions.

The mathematical representation of the objective function is given as

Minimize the cost function as shown in eq (1) below:

$$C = \sum_{i=1}^{k}\sum_{j=1}^{k} I_{ij} \quad (i \neq j) \tag{1}$$

Where i, j are the vertices of an edge

C = cost of cut

I_{ij} = cost of an edge.

As the problem involves bi-partitioning of a circuit so equality condition must be satisfied as eq (2):

$$\sum_{i=0}^{k} m_i = \sum_{j=0}^{k} n_j \tag{2}$$

Where m_i and n_j are nodes in the two partitions.

III. Solution Methodology: Ant Colony Optimization

Ant Colony Optimization (ACO) is a multi agent approach that simulates the foraging behavior of ants for solving difficult combinatorial optimization problems. Ants are social insects whose behavior is directed more towards the survival of colony as a whole than that of a single individual of the colony. An important and interesting behavior of an ant colony is its indirect co-operative foraging process. While walking from the food sources to the nest and vice versa, ants deposit a substance, called pheromone trail. Ants can smell pheromone; When choosing their way they tend to choose, with high probability, paths marked by strong pheromone concentration (shorter path). Also, other ants can used pheromone to find the location of food sources found by their nest mates. Therefore, ACO simulates the optimization of ant foraging behaviour [27].

Mathematical Formulation/Algorithm for circuit partitioning using ACO:

a) Take file as input and convert it into the matrix form.

b) Total ant or nodes equal to $\sum N_i$ Where i varies from 1 to n.

c) Now divide the circuit into two parts.

Partition $\sum P_j$ and $\sum P_k$

i. Where j varies from1 to n_1 and k varies from (n_1+1) to n.

ii. The no. of nodes in both partition should be equal i.e. Balance criteria as shown in eq (3) and eq (4):

$$\sum P_j = \sum P_k \tag{3}$$

and

$$\sum N_i = \sum P_j + \sum P_k \tag{4}$$

d) Calculate the gain(g) of the circuit using Internal cost (I) and External cost (E) as follows:

$$I = \left(\sum c(1:n_1, 1:n_1)\right)\sum c\left((n_1+1):n, (n_1+1):n\right) \tag{5}$$

$$E = \left(\sum c(1:n_1, (n_1+1):n)\right)\sum c\left((n_1+1):n, 1:n_1\right) \tag{6}$$

$$g = E - I \tag{7}$$

e) Start the movement of ants and initialize the parameters.

f) Take the first ant suppose from $\sum P_j$ and Move the ant based on the following probability:

$$p(v) = g\,\nu_c + p\,\nu_p + P_{\min} \tag{8}$$

Where v_c is the number of vertices adjacent to v, g is the connectivity weight or gain and v_p is the amount of pheromone of the animat's species on vertex v, p is the pheromone weight and P_{min} is a fixed amount added to prevent any probabilities from being zero.

a) Update the pheromone value of the nodes by using following formula

$$\text{Pheromone value} = \tau + \Delta\tau \tag{9}$$

Where τ = Pheromone value=Previous pheromone value

i. Increment in pheromone value

$$\Delta\tau = y\,\tau\,x\,e^{-(\tau)x} \tag{10}$$

ii. Forage Pheromone X-Scale=X = 0.4

iii. Forage Pheromone Y-Scale=Y = 1.5

Graph of above function shown in Fig. 1.

b) Evaporate the pheromone value based on the following formula:

$$\text{Pheromone value} = \tau * e^{-(\tau)(r)} \tag{11}$$

Where τ = Pheromone value = Previous pheromone value

Evaporation rate r =0.025

Graph of pheromone evaporation is shown in Fig. 2.

c) Store the node in tour.

d) Note down the cutset

e) Repeat the above steps for different nodes and note down the cut set value.

f) Store the cut set values for different partition in a variable and find the minimum cut set value.

Repeat the above steps until stopping criterion is met.

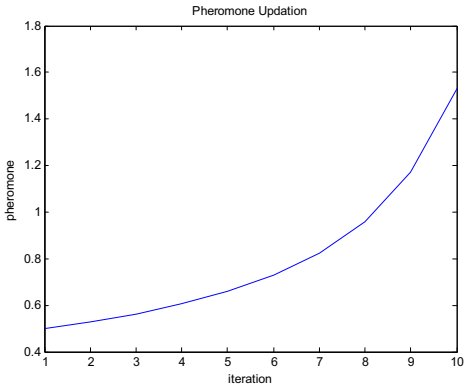

Fig. 1 Update Pheromone value after Every Movement

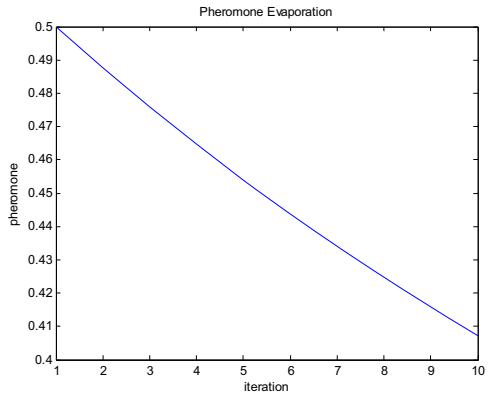

Fig. 2 Evaporation of Pheromone value

Genetic algorithms [23] are evolutionary computational models based on Charles Darwin's theory of natural evolution based on the concept of the survival of the fittest. Darwin observed that, as variations are introduced into a population with each new generation, the less-fit individuals tend to die off in the competition for food and this survival of fittest principle leads to improvements in species. The concept of natural selection was used to explain how species have been able to adapt to changing environments and how, consequently, species that are very similar in adaptivity may have evolved.

All genetic algorithms work on a population or a collection of several alternate solutions to the given problem. Each individual in the population is called a string or chromosome, in analogy to chromosomes in natural systems. The population size determines the amount of information stored by the GA. The GA population is evolved over a number of generations. All information required for the creation of appearance and behavioural feature of a living organism is contained in its chromosome.

The proposed algorithm follows the following steps –

Net list processing Circuit information is accepted in the form of circuit netlist, in accordance with ISPD 98 benchmark suite [30]. Netlist processing is done so as to convert the circuit netlist in the form of chromosome.

BFS Algorithm: The information of interconnection between the components in the netlist is converted in form of adjacency matrix. This Adjacency matrix information is then used to traverse the circuit in BFS algorithm so that the connected components remain clustered together as far as possible.

Initial population: Once the BFS order of components is obtained it is processed to form the initial solution for GA by converting it into 32-bit chromosome. The 32-bit chromosome contains integer values, with each integer value corresponding to each element of chromosome encoded to represent the partition number assigned and number of elements clustered to form single chromosome element.

In the Fig. 4, Value of j^{th} cell of chromosome is n1n2, where, n1 indicates the partition number assigned and n2 indicates the number of components clustered.

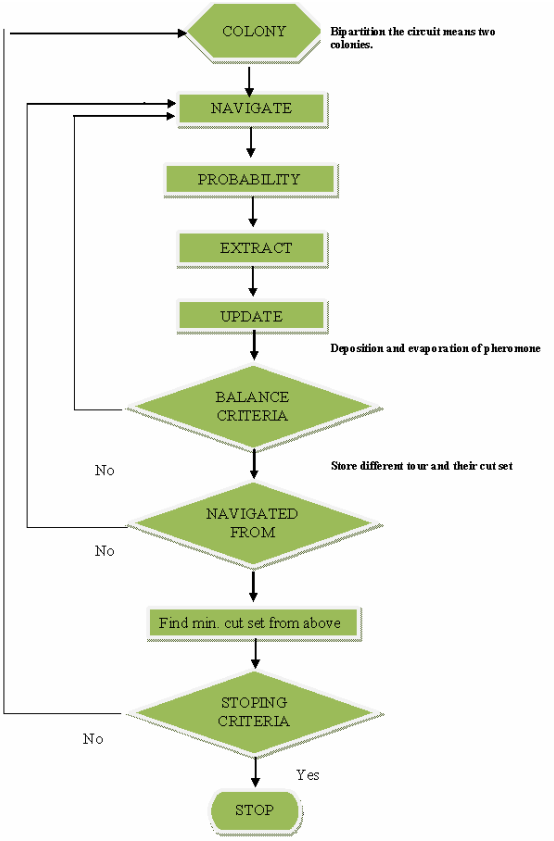

Fig. 3 Flow Chart for Circuit Partitioning Using ACO: Genetic Algorithms (GA)

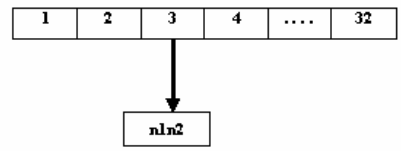

Fig. 4 32-bit Chromosome

Though other sorting data structure algorithm can be used such as depth first search algorithm, spanning tree algorithm etc, breadth first search algorithm has been found to capture circuit information more effectively [19]. Using the initial solution, random population is generated of the population size specified by the user. For each individual of the population, cost is computed. Objective function captures the cost of number of interconnections cut between the partitions.

Fitness Evaluation: Using the cost computed, each individual is evaluated for its fitness function. Based on Fitness values individuals are randomly selected using roulette wheel selection for crossover operation.

Crossover: Each individual is considered for selection as parent for crossover, with probability of selection proportional to its fitness value. Flexibility is incorporated in crossover operation with the user specifying the value for multipoint crossover. Offsprings generated from crossover replace the lowest fit individuals of the population if their fitness value is higher else, no replacement is made in the original population. In this algorithm, new offsprings replace the equivalent number of worst solutions from previous population which helps in survival of better solutions over several generations.

Mutation: After population replacement, mutation is performed on the bits randomly with small probability of mutation. Probability of mutation is very important, because the number of bits to be mutated depends on this probability. Mutation of bits is not similar to the traditional binary mutation operator, which is simple inversion of any random bits (depending on Probability of mutation), in the population.

Mutation changes the partition assigned to random number of components, where number of components depends on the probability of mutation. Even the partition assigned is generated randomly. Generally low values of probability of mutation are preferred so that population is not changed drastically which is critical. The population with mutated bits is then evaluated for fitness and again whole cycle of selection, crossover, replacement and mutation is followed and repeats for number of iterations of GA specified by the user.

No stopping criteria is specified in the algorithm itself because one of the advantages of evolutionary approach to partitioning is availability of ready solution at any stage, which if not globally optimal at least guarantees a good solution. But if no improvement is seen in the fitness and mincut results for consecutive 100 runs on a small scale circuit, GA is terminated.

The proposed algorithm is shown as flowchart in Fig. 5.

IV. Results and Discussion

The performance of the ACO and GA algorithms in tested on circuit partitioning instances (net lists) given on the MARCO GSRC VLSI CAD Bookshelf website [30]. The circuit net lists are in the ISPD 98 net list format. (.Net D files). Table I shows the comparison of average results for ACO and GA based partitioners on numerous net lists.

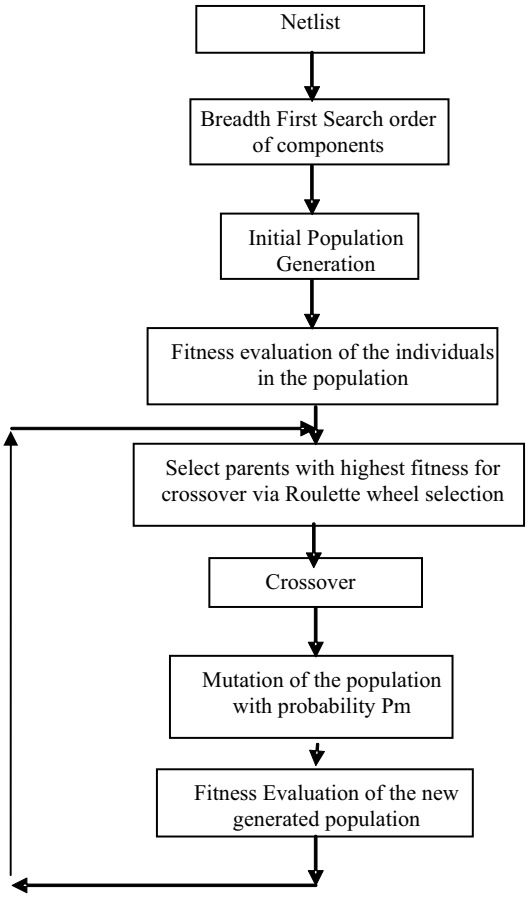

Fig. 5 Flowchart of the GA based partitioning algorithm

TABLE I
Comparison of Mincut Results for GA and ACO based Partitioners

File Name	Size	No. of Files	Mean Cut by GA based Partitio-ner	Mean Cut by ACO based Partitio-ner	Percentage Variation
Spp-N10 series	10	483	4.05	4.52	10.4%
Spp-N15 series	15	184	5.29	7.10	25.49%
Spp-N20 series	20	121	7.12	8.5	16.23%
Spp-N25 series	25	107	8.0	10.10	20.79%
Spp-N30 series	30	52	7.8	9.22	15.4%
Spp-N35 series	35	31	10.32	11.50	10.26%
Spp-N40 series	40	41	8.5	10.56	19.5%
Spp-N45 series	45	28	10.8	12.55	13.91%
Spp-N50 series	50	24	10.75	12.95	16.98%
Spp-N55 series	55	20	11.5	12.8	10.15%
Spp-N60 series	60	9	11.4	14.55	21.6%
Spp-N65 series	65	6	10.8	12.50	13.6%

The average results have been obtained on multiple number of partitioning instance groups in each size range. The

partitioning instances have been generated by the top down partitioning based placement process employed by the UCLA Capo Placer.

As seen from Table I, average results obtained by GA based partitioner are consistently better than these obtained by ACO based partitioner for all partitioning instances over all size ranges. Percentage improvement varies between 10.40 percent to 25.49 percent with an average variation of 16.19 percent.

V. Conclusion

Genetic and Ant Colony algorithms are applied to VLSI circuit partitioning problem. While both the algorithms have been successfully applied to this Non polynomial hard problem, GA's outperform ACO by an average of 16.19 percent over all test instances.

The main problem of a pure genetic based partitioning algorithm is that its run time increases quickly as the problem size increases. In order to reduce the run-times, a fast hybrid/memetic algorithm that employees local optimization in every generation is worth evaluating. GA combined with ACO for local search may give good solutions in less run time.

References

[1] Eugene L. Lawler, Karl N. Levitt, and James Turner, "Module Clustering to minimize delay in Digital Networks", IEEE Transactions on Computers, Vol. C-18 , No.1, pp. 47-57,Jan, 1969.

[2] B.W. Kerhinghan, S. Lin, "An efficient heuristic procedure for partitioning graphs", Bell System Tech. Journal, Vol. 49, pp. 291 – 307, Feb,1970.

[3] D.G. Schweikert and B.W. Kernighan, "A proper model for the partitioning of electrical circuits," Proc. ACM/IEEE Design Automation Workshop, pp. 57-62, 1972.

[4] C.M. Fiduccia and Mattheyses, "A Linear time heuristic for improving network partitions", Proc. 19th IEEE Design and Automation Conference, IEEE Press, Piscataway, NJ, USA , pp. 175-182, 1982..

[5] B. Krishnamurthy, "An improved min-cut algorithm for partitioning VLSI circuits", IEEE Trans. on Computers, Vol. C-33, May, 1989.

[6] L.A. Sanchis, "Multiple way network partitioning", IEEE Trans. on Computers, Vol. 38, No. 1, pp. 62-81 ,1989.

[7] Wei and Cheng, "Ratio-cut partitioning for hierarchical design", IEEE transc. on Computer Aided Design, pp. 911-921, July 1991.

[8] Jason Cong,L.Hagen, Andrew Kahng, "Net partitions yield better module partitions" , 29th ACM/IEEE Design Automation Conference , Anaheim, CA, USA , pp. 47-52 ,1992.

[9] Shawki Areibi and Anthony Vannelli, "Circuit partitioning using a tabu search approach", IEEE International Symposium on Circuits and Systems , Chicago, Illinois, pp. 1643-1646, March,1993.

[10] K.Shahookar and Mazumder "Genetic Multiway partitioning", IEEE 8th International Conference on VLSI Design, New Delhi, India, pp. 365-369, 4-7 Jan 1995.

[11] James Cane, Theodre Manikas, "Genetic Algorithms vs Simulated Annealing: A comparison of approaches for solving circuit partitioning problem", Technical report, University of Pittsburgh.

[12] S. M. Sait, and H. Youseff, "VLSI Physical Design Automation", McGraw Hill Publishers, New Jersey, 1995.

[13] George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar, "Multilevel Hypergraph Partitioning: Applications in VLSI Domain", IEEE 34th Design Automation Conference., Anaheim, California, United States, ACM Press New York, NY, USA., pp 526-529,1997.

[14] S. Areibi, "Memetic Algorithms for VLSI Physical Design: Implementation Issues", Genetic and Evolutionary computation Conference, San Fransisco, California, pp140-145, July, 2001.

[15] S. Areibi, "Recursive and flat partitioning for VLSI circuit design", The 13th International Conference on Microelectronics, Rabat, Morocco, pp.237-240, 2001.

[16] C. Ababei, S.Navaratnasothie, K. Bazargan and G. Karypis, "Multi-objective Circuit partitioning for Cutsize and path-base delay minimization", IEEE International Conference on Computer aided Design,2002.

[17] Maurizio Palesi,Tony Givargis, "Multi-Objective Design Space Exploration Using Genetic Algorithms", Proceedings of the 10th International symposium on Hardware/software codesign, ACM Press, Estes Park, Colorado , pp. 67-72 ,2002.

[18] Greg Stitt, Roman Lysecky, Frank Vahid, "Dynamic Hardware/Software Partitioning: A First Approach" ACM/IEEE Design Automation Conference 2003, Anaheim, California, USA,pp 250-255, June 2-6, 2003.

[19] P. Mazumder, E.M. Rudnik, "Genetic Algorithms for VLSI Design, Layout and Test Automation", Pearson Education, 2003.

[20] R. Banos, C. Gil, M.G. Montoya, and J. Ortega, "A Parallel evolutionary algorithm for circuit partitioning", Proceedings of the Eleventh Euromicro conference on Parallel, Distributed, and network based Processing, 2003.

[21] Sadiq M. Sait, Aiman H.El-Maleh, and Raslan H. Al-Abaji, "Enhancing performance of iterative heuristics for VLSI net list partitioning", ICECS-2003, pp. 507-510, 2003.

[22] D. Kolar, J. Divokovic Puksec and Ivan Branica, "VLSI Circuit partitioning using Simulated annealing Algorithm", IEEE Melecon, Dubrovnik, Croatia, May 12-15, 2004.

[23] D.E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine learning", Pearson Education, 2004.

[24] P. Ghafari , E. Mirhard, M.Anis, S. Areibi and M. Elmary, " A low power partitioning methodology by maximizing sleep time and minimizing cut nets", IWSOC, Bauf, Alberta, Canada, pp. 368-371, July, 2005.

[25] Naveed Sherwani, "Algorithms for VLSI Physical Design and Automation", Third edition, Springer (India) Private Limited, New Delhi, 2005

[26] G. Wang, W. Gang and R.Kastner, "Application Partitioning on programmable platforms using Ant Colony Optimization", Journal of Embedded Computing, Vol.2, Issue 1, 2006.

[27] Marco Dorigo and Thomas Stutzle, "Ant Colony Optimization", Prentice Hall of India Private Limited, 2006.

[28] Shawki Areibi and Fujian Ali, "A Hardware/Software co-design approach for VLSI circuit partitioning", IEEE International Workshop on SoC, Cairo, Egypt, pp.179-184, December, 2006.

[29] K.A. Sumitra Devi, N.P. Banashree and Annamma Abraham, "Comparative Study of Evolutionary Model and Clustering Methods in Circuit Partitioning Pertaining to VLSI Design", Proceeding of World Academy of Science, Engineering and Technology, April, 2007.

[30] http://www.gigascale.org/bookshelf