International Journal of Engineering Research and Development
ISSN: 2278-067X, Volume 1, Issue 8 (June 2012), PP.27-33
www.ijerd.com

CMOL Cell Assignment using Particle Swarm Optimization

Er. Tanvi Vaidya[1], Er. Abhijeet Kumar[2]
[1]*Student M.tech (regular), M.M. Engg. College, Mullana (Ambala).*
[2]*Lecturer ECE Deptt., M.M. Engg. College, Mullana (Ambala)*

Abstract—*CMOL cell assignment Problem (BP) is a telecommunication problem. CMOL cell assignment problem is very complex due to its constraint. In CMOL cell digital circuits are mapped to the CMOL cell. Each component of digital circuit is assigned to a particular cell. For mapping a digital circuit to the CMOL cell we have to first convert the circuit in to NOR gate form. In this paper we have present a CMOL cell assignment using particle swarm optimization algorithm. Our method transforms any logically synthesized circuit based on AND/OR/NOT gates to a NOR gate circuit and maps the NOR gates to CMOL. Particle swarm optimization (PSO) is a very popular algorithm and solves many optimization problems very efficiently from the last decades. We have consider all the constraints of CMOL cell assignment problem and tested our method on ISCAS benchmark circuits. We have compared our method with the genetic algorithm method and found that our algorithm works better than the genetic algorithm.*

Keywords—*Field programmable gate array (FPGA), nanodevice, reconfigurable computing, Particle swarm optimization(PSO), Genetic algorithm.*

I. INTRODUCTION

From the last few years, Nano electronics has made tremendous progress, with advances in novel Nano devices, Nano- circuits, Nano-crossbar arrays [4], manufactured by Nano imprint lithography[5], CMOS/Nano co-design architectures [6], and applications[7], [8]. Likharev and his colleagues [6] have developed the concept of CMOL technology for nanoelectronic devices. Many researchers have been done on hybridization of nanoelectronics/cmos devices. In [7] description of a digital logic architecture for 'CMOL' hybrid circuit swhich combine a semiconductor–transistor (CMOS) stack and two levels of parallel nanowires, with molecular-scale nanodevices formed between the nanowires at every crosspoint. In [9] two new methods of "in situ" training of CrossNets, based on either genuinely stochastic or pseudo-stochastic multiplication of analog signals, which may be readily implemented in CMOL circuits. In [10] description of neuromorphic network ("CrossNet") architectures for this future technology, in which neural cell bodies are implemented in CMOS, nanowires are used as axons and dendrites, while nanodevices (bistable latching switches) are used as elementary synapses. We have shown how Cross Nets may be trained to perform pattern recovery and classification despite the limitations imposed by the CMOL hardware. In [11] novel three-dimensional (3D) architecture of the CMOL circuit is introduced. This structure eliminates the special pin requirement of the original CMOL designs, providing a feasible and efficient solution to build the practical CMOL circuits. In [12] a novel CAD approaches to cell assignment of CMOL, hybrid CMOS/molecular circuit architecture. Method transforms any logically synthesized circuit based on AND/OR/NOT gates to a NOR gate circuit and maps the NOR gates to CMOL. In [13] new method based on dynamic interchange for cell assignment task of CMOL, a hybrid integrated circuit architecture, is proposed, first transform AND/OR/NOT gates composed of logic circuits into NOT gates and two inputs NOR gates, and then map the NOR/NOT gates to CMOL cells. In [14] two theoretical results concerning whether a combinatorial circuit is place able in CMOL FPGA described. In [15] proposed genetic algorithm for cell assignment of CMOL, hybrid CMOS/molecular circuit architecture is represented. Proposed method introduces a two dimensional block partially mapped crossover operator as well as mutation operator for our genetic algorithm and conducted experiments using the ISCAS benchmarks.

In this paper CMOL cell assignment problem using one of the popular optimization algorithm particle swarm optimization (PSO) have been solved. The solution given by our proposed algorithm is close to optimal solution but do not always find optimal solution. From the result shows that our algorithm gives the promising solution and performs better than the solution provided by genetic algorithm.
The rest of the paper is organized as follows: Basic PSO algorithm is explained in section 2. CMOL cell assignment problem is described in section 3.CMOL cell assignment problem using PSO is described in section 4. In Section 5, experimental results and discussion are described. Section 6 concludes the paper.

II. PARTICLE SWARM OPTIMIZATION

Particle swarm optimization (PSO) is a population based heuristic search technique developed by Dr. Eberhart and Dr. Kennedy in 1995 [1], inspired by social behavior of bird flocking or fish schooling. PSO has some properties similar with other evolutionary techniques such as Genetic algorithms (GA), like GA PSO also selects initial population randomly from solution search space. PSO does not use the operators like crossover and mutation like GA. As PSO is inspired from bird flocking it uses velocity equation to update the solutions and fly towards the best solution. The power of the technique is its fairly simple computations and sharing of information within the algorithm as it derives its internal communications from the social behavior of individuals. The individuals, called particles henceforth, are own through the multi-dimensional search space with each particle representing a possible solution to the multi-dimensional optimization problem. Each solution's fitness is based on a performance function related to the optimization problem being solved.

The movement of the particles is influenced by two factors using information from iteration-to-iteration as well as particle-to-particle. As a result of iteration-to-iteration information, the particle stores in its memory the best solution visited so far, called pbest, and

experiences an attraction towards this solution as it traverses through the solution search space. As a result of the particle-to-particle information, the particle stores in its memory the best solution visited by any particle, and experiences an attraction towards this solution, called gbest, as well. The first and second factors are called cognitive and social components, respectively. After each iteration, the pbest and gbest are updated for each particle if a better or more dominating solution (in terms of fitness) is found. This process continues, iteratively, until either the desired result is converged upon, or it's determined that an acceptable solution cannot be found within computational limits.

First the initial population is selected randomly from the solution search space than the position of particles are updated until the maximum limit of iteration or desired feasible solution found. For an n-dimensional search space, the i^{th} particle of the swarm is represented by an n-dimensional vector, $X_i = (x_{i1}, x_{i2} \ldots .x_{in})^T$. The velocity of this particle is represented by another n-dimensional vector $V_i = (v_{i1}; v_{i2} \ldots .. v_{in})^T$. The previously best visited position of the i^{th} particle is denoted as $P_i = (p_{i1}, p_{i2}, \ldots .....p_{in})^T$. `g' is the index of the best particle in the swarm. The velocity of the i^{th} particle is updated using the velocity update equation given by

$$v_{id} = v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \quad (1)$$

and the position is updated using

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

where d = 1, 2....n ; i = 1; 2....S , where S is the size of the swarm; c_1 and c_2 are constants, called cognitive and social scaling parameters respectively (usually, $c_1 = c_2$; r1 , r2 are random numbers, uniformly distributed in [0, 1]). Equations (1) and (2) are the initial version of PSO algorithm. A constant, Vmax, is used to arbitrarily limit the velocities of the particles and improve the resolution of the search. Further, the concept of an inertia weight was developed to better control exploration and exploitation. The motivation was to be able to eliminate the need for Vmax. The inclusion of an inertia weight (w) in the particle swarm optimization algorithm was first reported in the literature in 1998 (Shi and Eberhart, 1998) [2]. The resulting velocity update equation becomes:

$$v_{id} = w * v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \quad (3)$$

Eberhart and Shi (2000) [3] indicate that the optimal strategy is to initially set w to 0.9 and reduce it linearly to 0.4, allowing initial exploration followed by acceleration toward an improved global optimum.

III. CMOL CELL ASSIGNMENT PROBLEM

The basic idea behind CMOL is hybridization of CMOS/Nano electronics. Likharev[6] developed CMOL by using the hybrid of CMOS/Nano electronics. Due to impending crisis of the exponential ("Moore's-Law") progress of micro electronics may be postponed for more than a decade by the transfer from purely CMOS technology to hybrid CMOS/nanodevice circuits. In such a circuit, a specially designed CMOS chip is complemented with a simple nanoelectronic add-on: a nanowire crossbar with simple, similar, two-terminal nanodevices at each cross point.In order to be used effectively, the crossbar needs to be interfaced to the CMOS subsystem in a way which would allow individual access to each crosspoint nanodevice. Several sophisticated techniques based on stochastic doping of semiconductor nanowires had been suggested for this purpose; however, the "CMOL" interface [6, 7] seems more general and much easier for the practical implementation. In this approach (Fig. 1) the silicon/nanowire interface is provided by sharp-tip, conical vias ("pins") distributed all over the circuit area. The main invention here was a rotation of the nanowire crossbar by a certain angle α relative to the rectangular grid of pins (Fig. 1b).The nanodevice in CMOL can be any two terminal nanodevices, e.g. a binary "latching switch" based on molecules with two metastable internal states. Fig. 1 (Fig. 1a) shows the basic CMOL circuit with the interface between CMOS and nanowires. The pins connect the CMOS upper-level metal sand the nanowires. The nanodevices are sandwiched between the two levels of perpendicular nano-imprinted nanowires. This unique structure solves the problems of addressing much denser nanodevices with sparser CMOS components. Each nanodevice is accessed by the two perpendicular nanowires which connect to the nanodevice. The nanowires are, in turn, connected by pins to the CMOS circuits.

Strukov and Likharev [6] proposed the CMOL FPGA.The nanodevices are non-volatile switches, the CMOLFPGA could program those nanodevices and route the signals from CMOS to the nanowires and nanodevices, and back to CMOS again. Logic functions are created by a combination of CMOS inverters and diode-like nanodevices. They proposed a cell like structure for CMOL circuits. Each cell contains the CMOS device. Nanowires aligned with one direction receive signals from the outputs of the CMOS inverters. Those nanowires are OR'ed together with nanowires aligned with another (orthogonal) direction according to the nanodevice configurations. The OR'ed signal goes to the inverter's input, which is on the CMOS level. This OR-NOT logic is the preferred implementation of the CMOL FPGA.

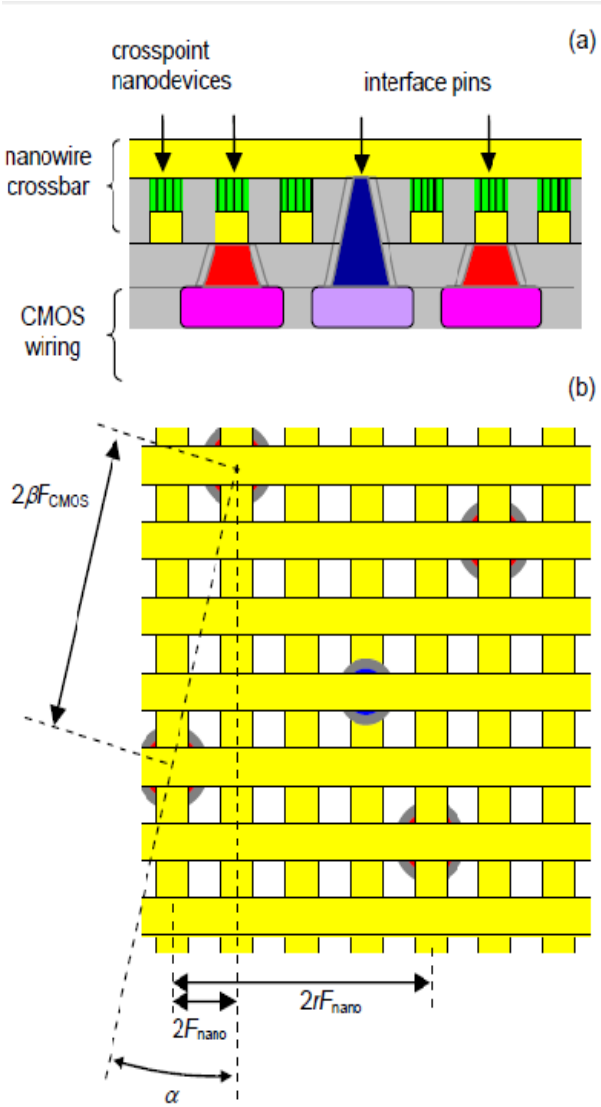

Fig. 1 Basic CMOL structure (a) Side view (b) Top view

Fig. 2 shows the basic cell structure of FPGA which contains nanodevice in each cell and this devices connected by nanowires. Each cell contains nanodevice and nanodevices are connected with nanowires. The cell structure is represented by three rows and four columns. Total 12 cells and three signals X, Y, and F connected with the three grayed cells' output pins. The X and Y get connected (ORed) using nanowire shown by brown lines. The logic generated by figure 2 is F=X+Y. First Strukov and Likharev assign Boolean circuits manually to the CMOL cell. They also presented a reconfigurable architecture [16] for CMOL FPGA ,that grouped CMOL cells to form tiles, which can be treated similar to traditional FPGA's clusters, and can utilize existing cluster-based FPGA CAD tools. However, that work also did not solve the CMOL cell assignment problem for the general case; because it was restricted to the tile abstraction and it relied on sufficient cell connectivity radius and the insertion of additional inverters for routing purposes.

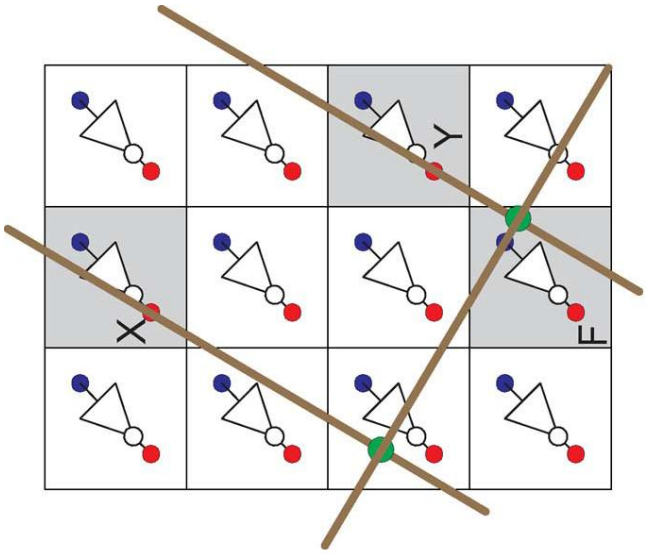

Fig. 2 CMOL FPGA Cell sample

So the cell assignment problem is a problem in which Boolean circuits are assigned to the CMOL cell. CMOL cell can only be connected to a limited number of neighbouring CMOL cells. The set of CMOL cells that can be connected to the input of a particular cell X is called the input **connectivity domain** of X. Similarly, the output connectivity domain refers to the set of cells that can be connected to the output of X.

IV. CMOL CELL ASSIGNMENT PROBLEM USING PARTICLE SWARM OPTIMIZATION

In this paper we have proposed a solution for CMOL cell assignment problem using continuous version of PSO. For solving any optimization problem we have to first formulate the problem according to optimization problem. In this case first we formulate the cell assignment problem according to PSO algorithm. Next subsection describes how we formulate the CMOL cell assignment problem.

1. Representation

To solve the problem, representation of the individual and fitness value is required. PSO algorithm is based on population (candidate solution) and each population have its own fitness value according to which it is compared from others, so we have to first represent the CMOL cell assignment problem in terms of PSO. In CMOL cell assignment, we have an Boolean circuit which we have to map on the CMOL cells as input and a sequence, which shows the cell number on which circuit devices are placed. First we transform the given Boolean circuit in the form of NOR gate circuit then we consider the NOR gate circuit as a graph whose vertex represents the number of nor gates in circuit and edges shows the connection between the nor gates. After formulating the circuit as a graph we use particle swarm optimization to assign nodes of a graph to a particular cell. PSO is based on population concept and each individual in population represents a solution, in case of CMOL cell assignment problem, solution is a sequence which shows the cell number for particular nor gate. So we have to first formulate each individual of PSO.

CMOL cell assignment problem is a discrete optimization problem. In the proposed solution continuous version of PSO is used instead of discrete version. To change the continuous version to discrete version for CMOL cell assignment problem new vector is generated from particular individual of a PSO. Individual or particle of a PSO contains the real values; to generate new discrete vector, index value is noted according to the dimension values of a particular individual. Index value of shortest dimension value is noted first then the index of second shortest dimension value is noted and so on up to the last dimension value. New generated discrete vector denotes the sequence tells that which cell has NOR gate and which cell remains empty.

We represent dimension as a number of cells in CMOL and value as sequence from the possible set of sequences order of cells to find optimal sequence. Position vector $X_{id}=\{x_1,x_2,x_3,.....x_d\}$where i is the particular individual and d represents the dimension index, is calculated using PSO. The position vector of each particle makes transformation about the continuous position. The position vector X_{id} has continuous values. By using the dimension values of X_{id} new discrete vector $S_{id}=[s_{i1}, s_{i2},\ldots\ldots s_{id}]$ generated. S_{id} is the sequence of cell number of i particle in the processing order with respect to the d dimension.

As every optimization problem has certain set of constraints CMOL cell assignment problem also have certain set of constraint. The constraints for this problem are

1. Each NOR gate is assigns to a particular cell only
2. There is a connectivity range for a particular NOR gate under which all the other NOR gate connected to that NOR gate should present. The connectivity range describes the minimum distance between the two connected NOR gate or the nodes edge.

The connectivity range is maximum length of wires connected between tow devices in a circuit. So each connected devices are present at max to the connectivity range.

2. Fitness Function

After representation of each individual we have to calculate fitness value of each individual. On the basis of fitness value we determine the optimal solution. In case of CMOL cell assignment problem optimal solution is the minimization of the value of equation (5). Our main objective is to minimize the fitness value, an individual who have the minimum fitness value is considered as the optimal solution.

$$a = \begin{cases} 1, & if\ dist(i,j) > R \\ 0, & else \end{cases} \quad (4)$$

$$f = \sum_{i,j} dist(i,j) + a * penalty \quad (5)$$

Here dist(i,j) denotes the distance between two cells in CMOL cell. R is the connectivity range of a cell, penalty value is added to the fitness f when connectivity constraint fails.

3. Algorithm CMOL Cell Assignment using PSO

To solve the CMOL cell assignment problem we have used the Particle Swarm Optimization (PSO). To assign a particular circuit in CMOL cell first the circuit is converted in the form of NOR gate circuit. We consider the NOR circuit as a graph and then apply PSO. Algorithm 1 is the proposed algorithm for the CMOL cell assignment problem.

Algorithm1: Algorithm for CMOL cell assignment using PSO
[Initialization Phase]
for s=0 to Swarm size **do**
for d=0 to dimension size **do**
Randomly initialize particle
New cell assignment sequence vector S_{id} is generated by the particle position index values
end for d
Compute fitness of initialized particle
Compute global best
end for s

[Update Phase]
Repeat
For s=0 to each swarm size **do**
ford=0 to problem dimension **do**
Update particles using PSO update equation
New cell assignment sequence vector S_{id} is generated by the particle position index values
e**nd for** d
Compute fitness of updated particle
if needed update historical information for global best(Pg)
end for s
until (stopping criteria is not met)

In the initialization phase of PSO, NOR gate is assigned randomly to the cells so we have select particular number of solution randomly. After random selection we check how many solutions satisfy the constraint. After checking the constraint we have the set of feasible solution which satisfies the constraint condition. After initialization PSO update phase updates the initial solution and generates the new solution. Again the constraint condition applied to the updated solutions or individuals. Fitness function is designed so that the solution which does not satisfy the constraint automatically not taken into consideration. A penalty value is added to the fitness of solution which does not satisfy the constraint condition. After addition of penalty value the individual is automatically not consider for good solution in next phase. All these procedure execute till the maximum number of iteration and at last we got the best result of all the iteration and consider these result as a cell assignment to the NOR gates.

V. EXPERIMENT RESULTS AND DISCUSSION

This section describes the experimental setup and result obtained after the experiment.

1. Experimental Setup

In this paper the parameter setting as suggested in [17]and [18] is used. Set swarm size S = 30. The inertia weight wis set to 0.73. c1 and c2 both are set to 2. Global variants of PSO are considered. Xmax and Xmin are the upper and lower bounds of the decision variables. Whenever the calculated position of particle exceeds the Xmax or lowers than the Xmin, particle position is set to Xmax or Xmin respectively (through extensive experiments these parameters are fine-tuned). Two criteria are applied to terminate the simulation of the algorithms. The first is the maximum number of function evaluations; set as 20000and the second is minimum error which is set to be optimal minimum total cell distance. One more parameter connectivity range of a cell is taken R=9.

2. Experimental Results

In this section we analyze the result obtained by our algorithm. To test the efficiency of our algorithm results of PSO is compared with Genetic algorithm (GA) results. The experiment is taken for ISCAS benchmark circuits. We first carve out the combinatorial logic, i.e., convert all inputs/outputs of latches to primary inputs/outputs respectively. We then use logic synthesis software (such as SIS) to convert the circuit into a NOR netlist. Table I compares our algorithm results with the genetic algorithms. The "cells" column indicates the number of CMOL cells that is assigned with NOR gates. The "Delay" indicates the maximum levels of NOR gates in both logic and routing. The "CPU time" column presents the running time of our algorithm.

From the experimental results it is clear that PSO can be used to solve CMOL cell assignment problem. PSO gives the better solution then the genetic algorithm. From the result it is also cleared that PSO does not perform well in all condition sometimes it also gives the solution worse than the GA but in most cases PSO outperforms GA.

***Table I* Experimental Results of PSO and GA**

Circuits	Genetic Algorithm			Particle Swarm Optimization		
	Cell	Delay	CPU time (sec.)	Cell	Delay	CPU time (sec.)
s27	8	7	0.06	8	7	0.02
s208	110	16	0.9	109	16	0.82
s298	85	11	2.52	85	11	2.2
s344	130	18	3.3	130	18	2.9
s349	136	18	3.70	134	18	3.21
s382	124	11	3.45	124	11	2.95
s386	140	10	3.96	138	10	3.72
s400	139	11	7.4	137	11	7.25
s420	248	22	9.2	248	22	9.2
s444	137	14	8.31	136	14	7.53
s510	270	30	27.12	266	30	24.12
s526	225	24	14.91	222	24	13.32
s641	206	18	112.14	206	18	109.35
s713	225	20	99.27	225	20	95.92
s820	402	31	121.19	400	31	116.54
s832	410	40	102.3	407	40	99.24
s838	509	38	245.67	507	38	239.23

VI. CONCLUSION

This paper represents the solution of CMOL cell assignment problem using PSO algorithm. Continuous version of PSO used in this paper. Prior methods of CMOL cell assignment can only solve small circuits with long CPU runtime and low area usage. CMOL cell assignment problem is of discrete type but the formulation is done using the continuous PSO. In CMOL cell assignment problem we have to place the circuit in CMOL cells. In initial phase cell sequences are selected randomly after that using PSO update equation new population created. Using this new population of PSO new cell sequences generated. Using the new sequence fitness function of each individual is calculated. This process continues till the maximum number of function evaluation reached. After the maximum number of evaluation the best among the all population during iterations are calculated. The best individual gives the best possible way to assign a circuit in CMOL cell and called as a optimal solution. From the experimental result shown in table I it is clear that our proposed solution using PSO performs better than the GA.

In future we can use many variants of PSO to solve the CMOL cell assignment problem and compare the result from our proposed algorithm. We can also use the concept of hybridization by mixing the property of two algorithms we can solve the CMOL cell assignment and compare the results. Hybridization of PSO also performs well so we can use that concept too.

REFERNCES

[1]. J. Kennedy and R. Eberhart, Particle swarm optimization, in Proc. IEEE International Conference Neural Networks, vol. 4, 1995, pp. 1942 - 1948.

[2]. Y. Shi and R. C. Eberhart, A modified particle swarms optimizer, in Proc. IEEE International Conference on Evolutionary Computation, Piscataway, NJ, 1998, IEEE Press, pp. 69-73.

[3]. R. C. Eberhart and Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, 2000 Congress on Evolutionary Computing, vol. 1, 2000, pp. 84-88.

[4]. Philip J. Kuekes, Duncan R. Stewart, and R. Stanley Williams. The crossbar latch: logic value storage, restoration, and inversion in crossbar circuits. Journal of Applied Physics, 97:034301–1–5, 2005.

[5]. D. J. Resnick. Imprint lithography for integrated circuit fabrication. Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures, 21:2624, 2003.

[6]. K. K. Likharev and D. V. Strukov. CMOL: devices, circuits, and architectures. In G Cuniberti and et al., editors, Introduction to Molecular Electronics, pages 447–477. Springer, Berlin, 2005.

[7]. Dmitri B. Strukov and Konstantin K. Likharev. CMOL FPGA: are configurable architecture for hybrid digital circuits with two-terminal nanodevices Nanotechnology, 16(6):888–900, 2005.

[8]. Oˆ. Tu¨rel, J. H. Lee, X. Ma, and Konstantin K. Likharev. Architectures for nanoelectronic implementation of artificial neural networks: new results. Neuro computing, 64:271–283, 2005.
[9]. J. H. Lee and K. K. Likharev, "In situ training of CMOL CrossNets," in Proc. WCCI/IJCNN'06, 2006, pp. 5026–5024.
[10]. J. H. Lee, X. Ma, and K. K. Likharev, "CMOL Crossnets:Possible neuromorphic nanoelectronic circuits," in Advances in Neural Information Processing Systems, Y. Weiss et al., Ed., vol. 18. MIT Press, 2006, pp. 755–762.
[11]. D. Tu et al., "Three-dimensional CMOL: Three-dimensional integration of CMOS/nanomaterial hybrid digital circuits,"Micro Nano Lett., vol. 2, pp. 40–45, Jun. 2007.
[12]. William N. N. Hung, Changjian Gao, Xiaoyu Song, and DanHammerstrom.Defect Tolerant CMOL Cell Assignment via Satisfiability. IEEE Sensors Journal, 8(6):823–830, June 2008.
[13]. Chu, Z. and Xia, Y. and Wang, L. and Hu, M., CMOL cell assignment based on dynamic interchange, ASIC, ASICON'09. IEEE 8th International Conference on, pp. 921-924, 2009.
[14]. Chen, G. and Song, X. and Hu, P., A theoretical investigation on CMOL FPGA cell assignment problem, Nanotechnology, IEEE Transactions on, vol. 8, pp. 322-329, 2009.
[15]. Xia, Y. and Chu, Z. and Hung, W.N.N. and Wang, L. and Song, X., CMOL cell assignment by genetic algorithm, NEWCAS Conference (NEWCAS), 2010 8th IEEE International, pp. 25—28, 2010.
[16]. D. Strukov and K. Likharev. A reconfigurable architecture for hybrid CMOS/nanodevice circuits. In FPGA'06, pages 131–140, Monterey, California, USA, 2006.
[17]. ZHANG Li-ping, YU Huan-jun and HU Shang-xu, Optimal choice of parameters for particle swarm optimization, Journal of Zhejiang University SCIENCE 2005, vol. 6A(6), pp. 528-534
[18]. Jaco F. Schutte and Albert A. Groenwold, A Study of Global Optimization Using Particle Swarms, Journal of Global Optimization (2005) vol. 31, pp. 93-108.
[19]. Deb, K., Multi-objective optimization using evolutionary algorithms, 2001, Wiley.