FUZZY SIMULATED EVOLUTION FOR POWER AND PERFORMANCE OPTIMIZATION OF VLSI PLACEMENT

Sadiq M. Sait Habib Youssef Junaid A. Khan Aiman El-Maleh

King Fahd University of Petroleum and Minerals Dhahran 31261, Saudi Arabia
{sadiq,youssef,jakhan,aimane}@ccse.kfupm.edu.sa

Abstract

In this paper, an algorithm for VLSI standard cell placement for low power and high performance design is presented. This is a hard multiobjective combinatorial optimization problem with no known exact and efficient algorithm that can guarantee finding a solution of specific or desirable quality. Approximation iterative heuristics such as Simulated Evolution (SE) are best suited to perform an intelligent search of the solution space. SE comprises three steps, evaluation, selection and allocation. Due to imprecise nature of design information at the placement stage, the various objectives and constraints are expressed in fuzzy domain. The search is made to evolve towards a vector of fuzzy goals. In this work, a new method to calculate membership in evaluation stage is proposed. Selection stage is also fuzzified and a new controlled fuzzy operator is introduced. The proposed heuristic is compared with Genetic Algorithm (GA) and the proposed fuzzy operator is compared with fuzzy ordered weighted averaging operator (OWA). Fuzzified SE (FSE) with controlled fuzzy operators was able to achieve better solutions.

1. Introduction

With advancement in VLSI technology, there has been a constant reduction in devices feature size. As a result, delay and power dissipation due to interconnects have become very significant [1]. In this work, the problem of minimizing interconnect delay and interconnect power dissipation is addressed at the placement stage [2].

Semi-formally, the placement problem can be stated as follows: Given a set of modules $M = \{m_1, m_2, \cdots, m_n\}$, and a set of signals $V = \{v_1, v_2, \cdots, v_k\}$, each module $m_i \in M$ is associated with a set of signals V_{m_i}, where $V_{m_i} \subseteq V$. Also each signal $v_i \in V$ is associated with a set of modules M_{v_i}, where $M_{v_i} = \{m_j | v_i \in V_{m_j}\}$. M_{v_i} is called a signal net. Placement consists of assigning each module $m_i \in M$ to a unique location such that a given cost function is optimized and constraints are satisfied [2]. Objectives addressed in this work are the minimization of wire-length, power dissipation, and circuit delay. Layout width is considered as a constraint. These are estimated as follows.

Estimation of Wire-length: The wire-length cost can be computed by adding wire-length estimates for all the nets in the circuit.

$$\text{Cost}_{wire} = \sum_{j \in M} l_j \tag{1}$$

where l_j is the wire-length associated with net v_j and M is the set of all cells in the circuit. This wire-length is computed using Steiner tree approximation.

Estimation of Power: In CMOS circuits, over 90% power dissipation is due to the switching activity [3], expressed as:

$$P_t \simeq \sum_{i \in M} \frac{1}{2} \cdot C_i \cdot V_{DD}^2 \cdot f \cdot S_i \cdot \beta \tag{2}$$

where P_t denotes the total power, V_{DD} is the supply voltage, S_i is the switching activity of cell i (module m_i), f is the clock frequency. The node total capacitance is denoted by C_i, and β is a technology dependent constant. Assuming that clocking frequency and power voltages are fixed, the total power dissipation of a CMOS circuit is a function of C_i and S_i. The capacitive load C_i of a gate comprises input gate capacitances of the cells driven by cell i (C_j^g) and that of interconnects capacitance at the cell output node (C_i^r), expressed as: $C_i = C_i^r + \sum_{j \in F_i} C_j^g$ where F_i is the set of fanout cells of cell i. In case of standard cell design, cell characteristics are fixed for a particular library, therefore we cannot reduce C_j^g. Furthermore, C_i^r are related to the corresponding interconnect wire-lengths l_i. Hence, the cost due to the overall power in VLSI circuits can be termed as:

$$\text{Cost}_{power} = \sum_{i \in M} S_i l_i \tag{3}$$

Estimation of Delay: Let a path π consist of nets $\{v_1, v_2, ..., v_k\}$, then its path delay T_π is expressed by

$T_\pi = \sum_{i=1}^{k}(CD_i + ID_i)$, where CD_i is the switching delay of the cell driving net v_i and ID_i is the interconnect delay of net v_i. The overall circuit delay is equal to T_{π_c}, where π_c is the most critical path in the layout. CD_i is constant and only ID_i depends on placement. This delay is estimated as: $ID_i = (LF_i + R_i^r) \times C_i$ where LF_i is a load factor of the driving block (independent of layout) and R_i^r is the interconnect resistance of net v_i. The cost function due to timing performance in the placement problem can be expressed as:

$$\text{Cost}_{delay} = T_{\pi_c} \tag{4}$$

Layout Width: In our work, layout width is considered as a constraint. The upper limit on the layout width is defined as: $Width_{max} = (1 + a) \times Width_{opt}$, where $Width_{max}$ is the maximum allowable width of the layout, $Width_{opt}$ is the minimum possible layout width obtained by adding the widths of all cells and dividing it by the number of rows in the layout. The parameter a denotes how wide the layout can be as compared to the optimal one.

2. Fuzzy Cost Measure

In order to select the best solution found so far, it is required to develop some cost measure representing all objectives. In this work, a goal directed search approach is adopted, where the best placement is one that satisfies as much as possible a user specified vector of fuzzy goals.

In order to combine three parameters and one constraint, the following fuzzy rule is suggested.

Rule R1: IF a solution is within *acceptable wire-length* AND *acceptable power* AND *acceptable delay* AND *within acceptable layout width* **THEN** it is an acceptable solution.

2.1. Controlled Fuzzy Operators (CFO)

In order to combine memberships in different fuzzy subsets some operator is needed. Two different operators have been used for placement in previous works, i.e., pure fuzzy operators (min and max operators) and ordered weighted averaging operators (OWA) [4, 5]. OWA was reported to be better than pure fuzzy operators. However, they suffer from the problem of selecting β (the averaging factor), which may be different for different problem instants. To solve this problem two new fuzzy operators are proposed in this work which are given below:

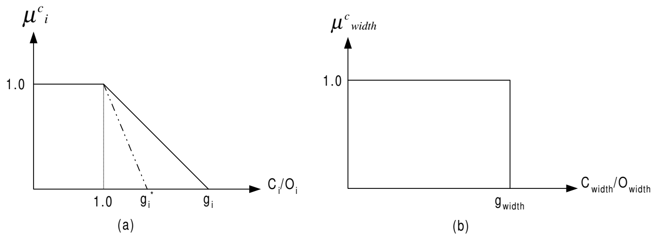

Figure 1: Membership functions within acceptable range.

2.1.1. Fuzzy Controlled AND Operator (FCAO):

The FCAO is analogous to t-norm operator and defined as :

$$\mu = 1 - \sum_{i=1}^{N} \bar{w_i}\bar{\mu_i} \tag{5}$$

where, μ is the membership in the fuzzy set of overall good solutions, μ_i is the membership in the fuzzy subset of good solutions with respect to the i^{th} objective, $\bar{\mu} = 1 - \mu$, $\bar{w_n} = \frac{\bar{\mu_n}}{\sum_{i=1}^{N}\bar{\mu_i}}$ and N is the total number of objectives. Combining all these, the FCAO operator may also be represented as:

$$\mu = 1 - \frac{\sum_{i=1}^{N}\bar{\mu_i}^2}{\sum_{i=1}^{N}\bar{\mu_i}} \tag{6}$$

2.1.2. Fuzzy Controlled OR Operator (FCOO):

The FCOO is analogous to s-norm operator. In the same manner as FCAO, FCOO is defined as:

$$\mu = \frac{\sum_{i=1}^{N}\mu_i^2}{\sum_{i=1}^{N}\mu_i} \tag{7}$$

It has to be noted that FCAO is applied on membership values in complementary fuzzy subsets, whereas FCOO is applied directly on the membership values in the fuzzy subsets.

Using controlled fuzzy operators (CFO), the above fuzzy rule translates to the following:

$$\mu_{pdl}^c(x) = 1 - \frac{\bar{\mu}_p^c(x)^2 + \bar{\mu}_d^c(x)^2 + \bar{\mu}_l^c(x)^2}{\bar{\mu}_p^c(x) + \bar{\mu}_d^c(x) + \bar{\mu}_l^c(x)} \tag{8}$$

$$\mu^c(x) = min(\mu_{pdl}^c(x), \mu_{width}^c(x)) \tag{9}$$

where $\mu^c(x)$ is the membership of solution x in fuzzy set of acceptable solutions, $\mu_{pdl}^c(x)$ is the membership in fuzzy set of "acceptable power AND acceptable delay AND acceptable wire-length", whereas $\mu_p^c(x)$, $\mu_d^c(x)$, $\mu_l^c(x)$, and $\mu_{width}^c(x)$ are the individual membership values in the fuzzy sets *within acceptable wire-length,*

```
ALGORITHM Simulated_Evolution(B, Φ_initial, StoppingCondition)
NOTATION
B= Bias Value.   Φ= Complete solution.
m_i= Module i.   g_i= Goodness of m_i.
ALLOCATE(m_i, Φ_i)=Function to allocate m_i in partial solution Φ_i
Begin
Repeat
  EVALUATION:
        ForEach m_i ∈ Φ evaluate g_i;
        /* Only elements that were affected by moves of previous */
        /* iteration get their goodnesses recalculated*/
  SELECTION:
        ForEach m_i ∈ Φ DO
              begin
                 IF Random > Min(g_i + B, 1)
                 THEN
                     begin
                         S = S ∪ m_i; Remove m_i from Φ
                     end
              end
      Sort the elements of S
  ALLOCATION:
        ForEach m_i ∈ S DO
              begin
                 ALLOCATE(m_i, Φ_i)
              end
Until   Stopping Condition is satisfied
Return Best solution.
End (Simulated_Evolution)
```

Figure 2: Structure of the simulated evolution algorithm.

power, delay, and layout width respectively. The superscript c represents "cost". The solution that results in maximum value of $\mu^c(x)$ is reported as the best solution found by the search heuristic.

The membership functions for fuzzy sets *within acceptable power, delay and wire-length* are shown in Figure 1(a), whereas the constraint *within acceptable layout width* is given as a crisp set as shown in Figure 1(b).

Since layout width is a constraint, its membership value is either 1 or 0 depending on $goal_{width}$ (in our case $goal_{width} = 1.25$) and it is combined with other objectives using min operator. However, for other objectives, by increasing or decreasing the value of $goal_i$ one can vary its preference in the overall membership function. The lower bounds ($O_i s$ for $i \in \{l, p, d, width\}$) are computed as: $O_l = \sum_{i=1}^{n} l_i^*$, $O_p = \sum_{i=1}^{n} S_i l_i^*, \forall v_i \in \{v_1, v_2, ..., v_n\}$; $O_d = \sum_{j=1}^{k} CD_j + ID_j^* \;\; \forall v_j \; in \; path \; \pi_c$; and $O_{width} = Width_{opt}$; where k is the number of nets in π_c. Superscript * indicates lower bound on an objective.

3. Fuzzy Simulated Evolution (FSE)

The general Simulated Evolution algorithm [6], is presented in Figure 2. In order to apply simulated evolution one has to design a suitable goodness measure, a cost function, and an appropriate allocation operator. These three together have the most impact on the behavior of the SE algorithm. Due to the multiobjective

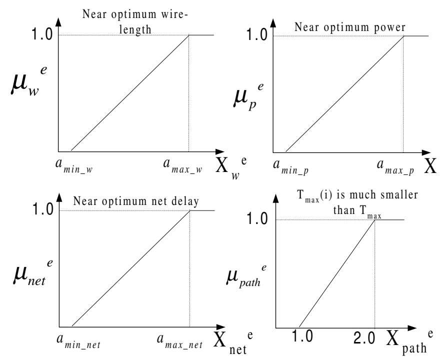

Figure 3: Membership functions used in fuzzy evaluation.

nature of the placement problem, the goodness measure, cost function, and the allocation operator should take into consideration all objectives.

Fuzzy Goodness Evaluation: Following the generation of an initial solution, the goodness of each cell in its current location is determined. A designated location of a cell is considered good if it results in short wire-length for its nets, reduced delay, and reduced power. These conflicting requirements can be expressed by the following fuzzy logic rule.

Rule R2: IF cell i is *near its optimal wire-length* AND *near its optimal power* AND (*near its optimal net delay* OR $T_{max}(i)$ *is much smaller than* T_{max}) **THEN** it has a high goodness.

where T_{max} is the delay of most critical path in the current iteration and $T_{max}(i)$ is the delay of the longest path traversing cell i in the current iteration.

With the AND and OR fuzzy operators implemented as CFO, rule **R2** evaluates to the expression below:

$$g_i = \mu_i^e(x) = 1 - \frac{\bar{\mu}_{iw}^e(x)^2 + \bar{\mu}_{ip}^e(x)^2 + \bar{\mu}_{id}^e(x)^2}{\bar{\mu}_{iw}^e(x) + \bar{\mu}_{ip}^e(x) + \bar{\mu}_{id}^e(x)} \quad (10)$$

where

$$\mu_{id}^e(x) = \frac{\mu_{inet}^e(x)^2 + \mu_{ipath}^e(x)^2}{\mu_{inet}^e(x) + \mu_{ipath}^e(x)} \quad (11)$$

g_i is the goodness of cell i, and $\mu_{id}^e(x)$ represents the membership in fuzzy set of *good timing performance.* It is obtained after applying FCOO to $\mu_{inet}^e(x)$ and $\mu_{ipath}^e(x)$.

$\mu_{ipath}^e(x)$ is included in the computation of $\mu_{id}^e(x)$ because if a cell is not on the critical path then it must have high goodness with respect to the delay objective.

If a cell i drives the net v_i, $\{v_1, v_2, ..., v_k\}$ is the set of nets connected to cell i and v_p is the net driven by the predecessor cell of cell i on the longest path traversing cell i, then base values $X_{iw}(x)$, $X_{ip}(x)$, $X_{inet}(x)$ for fuzzy sets near optimal wire-length, power, net delay and $X_{ipath}(x)$ for fuzzy set "$T_{\max}(i)$ much smaller than $T_{\max}$" are computed as given in Equations 12-15,

$$X_{iw}(x) = \frac{\sum_{j=1}^{k} l_j^*}{\sum_{j=1}^{k} l_j} \quad (12)$$

$$X_{ip}(x) = \frac{\sum_{j=1}^{k} S_j \ bl_j^*}{\sum_{j=1}^{k} S_j \ l_j} \quad (13)$$

$$X_{inet}(x) = \frac{ID_i^* + ID_{ip}^*}{ID_i + ID_{ip}} \quad (14)$$

$$X_{ipath}^{e}(x) = \frac{T_{max}}{T_{max}(i)} \quad (15)$$

where ID_{ip} is interconnect delay of the net driven by the predecessor cell of cell i on the current longest path traversing cell i. Membership functions of these base values are shown in Figure 3. In Figure 3, the values of a_{min} and a_{max} depend on the statistical nature of the base values. It is observed that these base values are normally distributed. Therefore, we can say that around 95% cells have base values in the range $[\bar{X}_i - 2\sigma_i, \bar{X}_i + 2\sigma_i]$, where $\bar{X}_i$ is the mean value of $X_i(x)$ and σ_i is the standard deviation of $X_i(x)$ for $i = w, p, net$. a_{min} and a_{max} are therefore computed as:

$$a_{min\_i} = \bar{X}_i - 2\sigma_i \ \ and \ \ a_{max\_i} = \bar{X}_i + 2\sigma_i \quad (16)$$

The values of a_{min} and a_{max} are computed in the beginning and then recomputed again when the size of selection set is around 90 percent of the initial value.

Selection: In this stage of the algorithm, some cells are selected probabilistically depending on their goodness values. Bias concept in selection step, present in the original SE algorithm [6], is the major drawback of the heuristic. It is not easy to select the bias value because it varies from problem to problem. Also in the case of placement it varies from circuit to circuit. To overcome this problem, another selection scheme is proposed. According to this scheme, a random number is generated in the range $[0, M]$ and compared with g_i. If the generated random number is greater than g_i, then the cell is selected for allocation. The value of M is calculated as follows:

$$M = \bar{G} + 2\sigma_g \quad (17)$$

where $\bar{G}$ and σ_g are the average goodness and standard deviation of goodness values of cells in the current iteration. Value of M is calculated in the beginning and updated only once, when the size of selection set is 90% of its initial size [7].

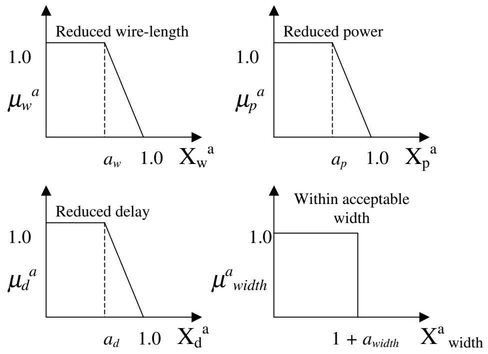

Figure 4: Membership functions used in allocation.

Allocation: In the allocation stage, the selected cells are to be placed in the best available locations. We have considered selected cells as movable modules and remaining cells as fixed modules. Selected cells are sorted in descending order of their goodness with respect to their partial connectivity with unselected cells. One cell from the sorted list is selected at a time and its location is swapped with other movable cells in the current solution. The swap that results in the maximum gain is accepted and the cell is removed from the selection set.

The goodness of the new location is characterized by the following fuzzy rule:

Rule R3: IF a swap results in *reduced overall wire-length* AND *reduced overall power* AND *reduced delay* AND *within acceptable layout width* **THEN** it gives good location.

The above rule is interpreted as follows.

$$\mu_{i\_pwd}^a(l) = 1 - \frac{\bar{\mu}_{ip}^a(l)^2 + \bar{\mu}_{iw}^a(l)^2 + \bar{\mu}_{id}^a(l)^2}{\bar{\mu}_{ip}^a(l) + \bar{\mu}_{iw}^a(l) + \bar{\mu}_{id}^a(l)} \quad (18)$$

$$\mu_i^a(l) = min(\ \mu_{i\_width}^a(l), \ \mu_{i\_pwd}^a(l) \ ) \quad (19)$$

the superscript a is used here to represent allocation. $\mu_i^a(l)$ is the membership of cell i at location l in the fuzzy set of good location. $\mu_{i\_pwd}^a(l)$ is the membership in the fuzzy set of "reduced wire-length and reduced power and reduced delay". $\mu_{iw}^a(l)$, $\mu_{ip}^a(l)$, $\mu_{id}^a(l)$, and $\mu_{iwidth}^a(l)$ are the membership in the fuzzy sets of reduced wire-length, reduced power, reduced delay and within acceptable width respectively. Notice that the third AND operator in the above fuzzy rule is implemented as a pure min because the width constraint has to be always satisfied.

If a cell i swaps its location with cell j then the base values are computed as shown in Equations 20-23:

$$X_{iw}^a(l) = \frac{(\sum_{m=1}^{k_i} l_{im} + \sum_{m=1}^{k_j} l_{jm})_n}{(\sum_{m=1}^{k_i} l_{im} + \sum_{m=1}^{k_j} l_{jm})_{n-1}} \quad (20)$$

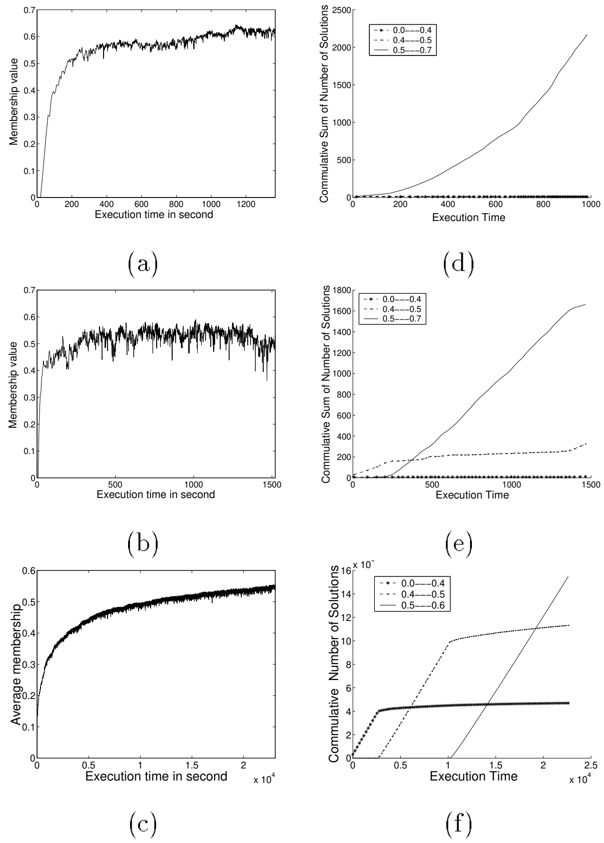

Figure 5: (a), (b) and (c) show Membership values versus execution time for FSE with CFO, FSE with OWA and GA respectively. (d), (e) and (f) show cumulative number of solutions visited in a specific membership range versus execution time for FSE with CFO, FSE with OWA and GA respectively.

$$X_{ip}^{a}(l) = \frac{(\sum_{m=1}^{k_i} S_{im} l_{im} + \sum_{m=1}^{k_j} S_{jm} l_{jm})_n}{(\sum_{m=1}^{k_i} S_{im} l_{im} + \sum_{m=1}^{k_j} S_{jm} l_{jm})_{n-1}} \quad (21)$$

$$X_{id}^{a}(l) = \frac{(ID_i + ID_{ip} + ID_j + ID_{jp})_n}{(ID_i + ID_{ip} + ID_j + ID_{jp})_{n-1}} \quad (22)$$

$$X_{i\_width}^{a}(l) = \frac{Width_n}{Width_{opt}} \quad (23)$$

where, subscripts n and $n-1$ show the iteration numbers, $\{v_{i1}, v_{i2}, ..., v_{ik_i}\}$ is the set of nets connected to cell i, $Width_n$ is the actual width at n^{th} iteration.

Membership functions for these base values are shown in Figure 4. The values of a_w, a_p, a_d and a_{width} depend upon priority on the optimization level of the respective objective. Typical values for a_w, a_p and a_d are in the range $[0.75, 0.95]$, whereas a_{width} is in the range $[0.2, 0.5]$. In our case we have set $a_w = 0.75$, $a_p = 0.75$, $a_d = 0.85$ and $a_{width} = 0.25$.

Stopping Condition The algorithm terminates when no further improvement is observed in the best solution found so far in terms of Equation 9.

4. GA Based Optimization

For comparison purposes, we have also implemented **genetic algorithm (GA)** [8] with $\mu^c(x)$ as **fitness** value. **Roulette wheel** selection scheme [8], is used. **Partially mapped crossover (PMX)** is used to generate new offsprings. For the selection of next generation, **Extended Elitism Random Selection** scheme is used, where half of the chromosomes in the next population are the best among offspring and current population and half are selected randomly. A variable **mutation** is used that depends upon the standard deviation of fitness in the current population. Stopping criteria is the maximum number of generations.

5. Experiments and Results

We have applied FSE with CFO, FSE with OWA and GA on eleven different ISCAS-85 and ISCAS-89 benchmark circuits. For FSE, execution is aborted when no improvement is observed in the last 500 iterations. For GA, the stopping criteria is 10,000 generations.

Table 1 compares the quality of final solution generated by FSE with CFO and OWA, and GA. The circuits are listed in order of their size (122-1753 modules). From the results, it is clear that GA is better than FSE for smaller circuits but with extremely larger execution time (undesirable), whereas for circuits with large number of cells FSE outperforms GA. In most of the cases, CFO as compared to OWA gives solution with larger reduction in the cost of few objectives with slight increase in the cost of others (desirable). In the cases where CFO gives slightly inferior solution, the execution time is also low. In general, CFO has given either same or better quality solution than OWA without any problem of selection of β (averaging factor in OWA).

In order to compare improvement in the quality of solution versus time, we plot the current membership values of the solution obtained by FSE CFO and OWA (Figure 5-(a) and (b)). The average fitness (membership) values in a current population obtained by GA versus execution time are plotted in Figure 5-(c). These plots are for test case S1196. It can be observed that the quality of solution improves rapidly in FSE based search as compared to GA. This behavior was observed for all test cases

Figures 5(d), (e) and (f) track with time the total number of solutions found by FSE with CFO and OWA, and GA for various membership ranges. Note however that FSE exhibited much faster evolutionary rate than GA. For example, after about 50 seconds,

Table 1: Layout found by FSE with CFO, FSE with OWA and GA. "L", "P" and "D" represent the wire-length, power, and delay costs and "T" represents execution time in seconds.

Circuit	FSE with CFO				FSE with OWA				GA			
	L (μm)	P (μm)	D (ps)	T (s)	L (μm)	P (μm)	D (ps)	T(s)	L (μm)	P (μm)	D (ps)	T (s)
S2081	2639	431	114	92	2693	462	112	43	2426	388	113	2341
S298	5130	946	142	84	4989	1013	133	104	4062	838	130	2922
S386	7097	1707	196	314	7088	1640	197	152	6824	1665	181	3945
S832	23959	5253	399	365	24705	5827	390	1643	21015	4787	232	7206
S641	11499	2691	689	680	13906	3321	702	618	18320	4365	736	21982
S953	31846	4883	237	942	32340	5242	245	1278	32031	5156	230	11221
S1238	40726	12696	383	585	39629	12377	371	1168	52679	15473	410	16208
S1196	38041	11606	351	999	42426	12745	364	1521	51804	15259	370	23070
S1494	55010	13628	699	2979	56961	14071	719	3378	71021	17497	803	26032
S1488	56875	13835	706	5357	57091	13887	710	3529	69792	17346	784	21434
C3540	164156	58002	686	10013	164897	58055	734	18318	310996	109850	924	43232

almost all new solutions discovered by FSE with CFO have a membership more than 0.5 in the fuzzy subset of good solutions with respect to all objectives, and almost none were found with lower membership values. In contrast, for GA, it is only after 10,000 seconds that the first solution with membership more than 0.5 was found (see Figure 5). This behavior was observed for all test cases.

6. Conclusion

In this paper, we have proposed Fuzzy Simulated Evolution Algorithm for low power high performance VLSI standard cell placement. Fuzzy logic is used to overcome the multi-objective nature of the problem. In FSE, fuzzy logic is employed at evaluation and allocation stages and in choice of the best solution from the set of generated solutions. In GA, fuzzy logic is used in the fitness evaluation. We have also proposed new controlled fuzzy operators. The proposed fuzzy operators are compared with OWA operators. Also FSE is compared with GA.

It is observed that FSE perform much better than GA in terms of execution time. FSE performs better than GA in terms of the final solution in bigger circuits, whereas GA is better for smaller circuits but with considerably higher execution time. Furthermore, quality of solution improved more rapidly in FSE based search as compared to GA. In addition, CFO gives solutions with better or same quality than OWA without the need for the selection of any parameter like β. It also exhibits better evolutionary rate as compared to OWA.

Acknowledgment:
The authors thank King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, for support. Project Code: COE/ITERATE/221

7. References

[1] J. Cong, L. He, C. Koh, and P. Madden. Performance Optimization of VLSI Interconnect Layout. *Integration, the VLSI Journal*, 21:1–94, 1996.

[2] Sadiq M. Sait and Habib Youssef. VLSI Physical Design Automation: Theory and Practice. *McGraw-Hill Book Company, Europe*, 1995.

[3] Srinivas Devadas and Sharad Malik. A Survey of Optimization Techniques Targeting Low Power VLSI Circuits. *32nd ACM/IEEE DAC*, 1995.

[4] Ronald R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transaction on Systems, MAN, and Cybernetics*, 18(1), January 1988.

[5] Sadiq M. Sait, Habib Youssef, and Ali Hussain. Fuzzy Simulated Evolution Algorithm for Multiobjective Optimization of VLSI Placement. *IEEE Congress on Evolutionary Computation*, pages 91–97, July 1999.

[6] R. M. Kling and P. Banerjee. ESP: Placement by Simulated Evolution. *IEEE Transaction on Computer-Aided Design*, 3(8):245–255, March 1989.

[7] Junaid A. Khan. Performance Driven, Low Power, Standard VLSI Cell Placement Using Simulated Evolution. Master's thesis, Department of Electrical Engineering, King Fahd University of Petroleum and Minerals, Saudi Arabia, January 2001.

[8] Sadiq M. Sait and Habib Youssef. *Iterative Computer Algorithms with Applications in Engineering: Solving Combinatorial Optimization Problems.* IEEE Computer Society Press, California, December 1999.