# SIMULATED EVOLUTION ALGORITHM FOR MULTIOBJECTIVE VLSI NETLIST BI-PARTITIONING

*Sadiq M. Sait, Aiman H. El-Maleh, and Raslan H. Al-Abaji*

King Fahd University of Petroleum & Minerals Computer Engineering Department
Dhahran 31261, Saudi Arabia
{sadiq,aimane,raslan}@ccse.kfupm.edu.sa

## ABSTRACT

*In this paper, the Simulated Evolution algorithm (SimE) is engineered to solve the optimization problem of multi-objective VLSI netlist bi-partitioning. The multi-objective version of the problem is addressed in which, power dissipation, timing performance, as well as cut-set are optimized while Balance is taken as a constraint. Fuzzy rules are used in order to design the overall multi-objective cost function that integrates the costs of three objectives in a single overall cost value. Fuzzy goodness functions are designed for delay and power, and proved efficient. A series of experiments are performed to evaluate the efficiency of the algorithm. ISCAS-85/89 benchmark circuits are used and experimental results are reported and compared to earlier algorithms like GA and TS.*

## 1. INTRODUCTION

VLSI circuit design has various objectives. Until the beginning of this decade, two main objectives of VLSI circuit design were: the minimization of interconnect wire length and the improvement of timing performance. A large number of efforts targeting either one (especially cut-set) or both of the above objectives are reported in the literature[1, 2]. The power consumption of the circuit was not of main concern while trying to optimize the above two objectives. As different techniques are applicable and have been reported [3] at different steps of the VLSI design process, few performance-driven partitioning techniques at physical level design exist in literature. Furthermore, to our knowledge, no effort has been reported that targets the optimization of three objectives simultaneously (power, delay, cut-set).

In this work, the above problem is addressed in the partitioning step at the physical level. The Simulated Evolution algorithm is tailored for the multiobjective optimization of Partitioning. This paper is organized as follows.

In the next section, the SimE algorithm is introduced. In Section 3 the problem is presented and the cost functions are formulated. Section 4 presents the goodness functions and the design implementation details of the algorithm, and then experimental results are reported and discussed in section 5.

## 2. SIME ALGORITHM

In this section, Simulated Evolution is summarized. The pseudocode of SimE is given in [4]. SimE operates on a single solution, termed as *population*. Each population consists of elements. In case of partitioning problem, these elements are cells to be moved. In the *Evaluation* step, *goodness* of each element is measured. Goodness of an element is a single number between ë0í and ë1í, which is a measure of how near is the element from its optimal

location. After that comes *Selection* which is the process of selecting those individuals which are unŶt (badly placed) in the current solution. For that purpose, goodness of each individual is compared with a random number (in the range [0,1]), if the goodness is less than the random number then it is selected. The whole algorithm is repeated until some stopping criteria is met. Stopping criteria may be the number of iterations for which there is no further improvement in the overall cost, or total number of iterations are completed. At the end of *Repeat* loop, the algorithm returns the best solution found. Higher value of goodness means that the element is near its optimal location. For SOP, the goodness can be calculated as follows,

$$g_i = \frac{O_i}{C_i} \qquad (1)$$

**ALGORITHM** *Simulated_Evolution*$(B, \Phi_i, SC)$
$\quad B \quad$ = Bias Value. $\qquad \Phi \quad$ = Complete Solution.
$\quad \Phi_i \quad$ = Initial Solution. $\quad SC$ = Stopping Criterion.
$\quad e_i \quad$ = Individual link in $\Phi$.
$\quad O_i \quad$ = Lower bound on cost of $i^{th}$ link.
$\quad C_i \quad$ = Current cost of $i^{th}$ link in $\Phi$.
$\quad g_i \quad$ = Goodness of $i^{th}$ link in $\Phi$.
$\quad S \quad$ = Queue to store the selected links.
$\quad Allocate(e_i, \Phi_i)$ allocates $e_i$ in partial solution $\Phi_i$.
**Repeat**

```
EVALUATION:    ForEach e_i ∈ Φ DO
               Begin
                    Evaluate g_i
               End
SELECTION:     ForEach e_i ∈ Φ DO
               Begin
                    If random(1) > min(g_i + B, 1)
                    Begin
                         S = S ∪ e_i;
                         Φ = Φ − e_i;
                    End
               End
               sort(S)
ALLOCATION:    ForEach e_i ∈ S DO
               Begin
                    Allocate(e_i, Φ_i)
               End
```

**Until** *SC is satisfied*
**Return** (Best solution)
**End** *Simulated_Evolution*

Figure 1: Structure of the simulated evolution algorithm.

In *Selection*, an individual having high goodness measure still has a non-zero probability of assignment to set $P_s$. It is this element of non-determinism that gives **SimE** the capability of escaping local minima. After the selection process, the *allocation*

operator is used to place cells in $P_s$ to new locations. The choice of allocation function is problem speciÝc.

## 3. PROBLEM FORMULATION AND COST FUNCTIONS

This work addresses the problem of VLSI netlist partitioning with the objectives of optimizing power consumption, timing performance (delay), and cut-set while considering the Balance constraint (same as area constraint as unit area is assumed for every gate). Formally, the problem can be stated as follows:

Given a set of modules $V = \{v_1, v_2, ..., v_n\}$, the purpose of partitioning is to assign the modules to a speciÝed number of clusters $k$ (two in our case) satisfying prescribed properties. In general, a circuit can have multi-pin connections (nets) apart from two-pin. Our task is to divide $V$ into 2 subsets (blocks) $V_0$ and $V_1$ in such a way that the objectives are optimized, subject to some constraints.

**Cutsize** The cutsize cost function can be also be written as follows :

$$Minimize \quad f = \sum_{e \in \psi} w(e) \qquad (2)$$

where $\psi \subset E$ denotes the set of off-chip wires. The weight $w(e)$ on the edge $e$ represents the cost of wiring the corresponding connection as an external wire.

**Delay** In the general delay model where gate delay $d(v)$ and constant inter-chip wire delay are considered, $d_c \gg d(v)$ where $d_c$ is actually due to the off-chip capacitance denoted as $C_{off}$. Let the delay of node $v_i \in V$ be $d(v_i)$ and the delay of net $e_k \in E$ which is cut be $d_c$. Given a partition $\Phi : (V_A; V_B)$, the path delay $d(p_{ij})$ between nodes $v_i$ and $v_j$ is the sum of the node delays $d(v_i) \in V(p_{ij})$ and the delay of nets which are cut, that is :

$$Minimize \quad d(p_{ij}) = \sum_{v_i \in V(p_{ij})} d(v_i) + d_c \times ncut(p_{ij}) \qquad (3)$$

**Power** The average dynamic power consumed by a CMOS logic gate in a synchronous circuit is given by:

$$P_i^{average} = 0.5 \frac{V_{dd}^2}{T_{cycle}} C_i^{load} N_i \qquad (4)$$

where $C_i^{load}$ is the load capacitance, $V_{dd}$ is the supply voltage, $T_{cycle}$ is the global clock period, and $N_i$ is the number of gate output transitions per clock cycle. $N_i$ is calculated using the symbolic simulation technique of [5] under a zero delay model. $C_i^{load}$ in Eq. 4 consists of two components: $C_i^{basic}$ which accounts for the load capacitances driven by a gate before circuit partitioning, and the extra load $C_i^{extra}$ which accounts for the additional load capacitance due to the external connections of the net after circuit partitioning. Then, the total power dissipation of any circuit $\zeta$ is:

$$P_\zeta = \beta \frac{V_{dd}^2}{T_{cycle}} \sum_{i \in \zeta} (C_i^{basic} + C_i^{extra}) N_i \qquad (5)$$

where $\beta$ is a constant that depends on technology. When a circuit partitioning corresponds to a physical partitioning, $C_i^{extra}$ of a gate that is driving an external net is much larger than $C_i^{basic}$.

**Area or Balance Constraint** The balance constraint is given as follows:

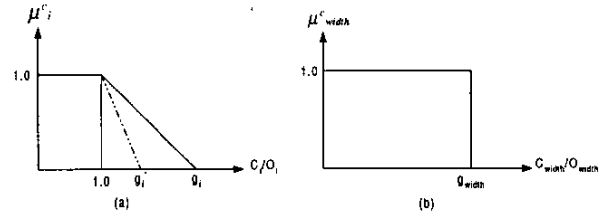$$\frac{|\beta_1 - \beta_2|}{\phi} \leq \alpha \qquad (6)$$



Figure 2: Membership function(s).

where $\beta_i$ is the number of cells in partition $i$ and $\phi$ is the total number of cells in the circuit, and the balance factor $\alpha$ ($0.5 < \alpha < 1.0$).

### 3.1. Overall Fuzzy Cost Function:

In order to solve the multiobjective partitioning problem, linguistic variables are deÝned as: cut-set, power dissipation, delay and balance. The following fuzzy rule is used to combine the conÐicting objectives

**IF** a solution has
    *Small cut-set* **AND**
    *Low power consumption* **AND**
    *Short delay* **AND**
    *Good Balance*
    **THEN** it is an *GOOD* solution.

The above rule is translated to *and-like* OWA fuzzy operator [6] and the membership $\mu(x)$ of a solution $x$ in fuzzy set *good solution* is given as:

$$\mu_{pdcb}^c(x) = \beta^c \times min(\mu_p^c(x), \mu_d^c(x), \mu_c^c(x), \mu_b^c(x)) +$$
$$(1 - \beta^c) \times \frac{1}{4} \sum_{j=p,d,c,b} \mu_j^c(x) \qquad (7)$$

where $\mu^c(x)$ is the membership of solution $x$ in fuzzy set of acceptable solutions, $\mu_{pdcb}^c(x)$ is the membership value in the fuzzy sets of 1 within acceptable *power*î, 1within acceptable *delay*î, 1within acceptable cut-setî and 1within acceptable *balance*î respectively. $\beta^c$ is the constant in the range $[0, 1]$, the superscript $c$ represents the cost. In this paper, $\mu^c(x)$ is used as the aggregating function. 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 *Low power consumption, Short delay, Small cut-set*, are shown in Fig. 2(a) We can vary the preference of an objective $j$ in the overall membership function by changing the value of $g_j$ which represents the relative acceptable limits for each objective whrere $g_j \geq 1.0$. Fig. 2(b) represents the membership functions for fuzzy set *good Balance*. $O_i$ is the estimate of lower bound on the cost of an individual $i$, and $C_i$ is the actual cost of $i$. $O_i$îs are independent of iteration, therefore, these are estimated only in the beginning. Whereas, $C_i$ has to be calculated in every iteration for every element.

## 4. PROPOSED SCHEME AND IMPLEMENTATION DETAILS

In this section, several important implementation details of SimE algorithm are discussed. The description is combined with fuzzy

logic implementation. In the proposed algorithm, interconnect power dissipation, overall circuit delay and Cutset are used as objectives and Balance as a constraint, which makes it a MOP. In order to solve this MOP, fuzzy logic that provides a convenient method to combine possibly conDicting objectives is used. It is clear that fuzzy logic can be applied at different stages of the SimE algorithm. These stages are *evaluation* and *allocation*. In the proposed algorithm, fuzzy logic is applied to the evaluation stage. In the *evaluation* stage, a new strategy for evaluating the goodness of a cell is employed.

### 4.1. Proposed Fuzzy goodness Evaluation Scheme

In this stage of the algorithm, goodness of individual cell is computed. $gd$, $gp$ and $gc$ are deYned to be Delay, power and cut goodness of a cell, respectively. A fuzzy membership for each goodness function is then derived in order to get the overall goodness of the cell in its partition.

**Cut Goodness** Taking into account the hypergraph representation of the circuit, the goodness function $gc$ is deYned and computed as follows:

$$gc_i = \frac{d_i - w_i}{d_i} \tag{8}$$

where $gc_i$ is the goodness of cell $i$, $V_i = \{v_1, v_2, ..., v_k\}$ is the set of nets connected to cell $i$, $U_i$ is a subset of $V_i$ containing the connected nets to cell $i$ that are cut; the cardinality of $V_i$ is expressed as $d_i$. The net $e_k$ is said to be cut if and only if cell $u \in e_k$ and cell $e \in v_k$ and $Block(u) \neq Block(v)$. The number of nets connected to $i$ and having the status as cut is expressed as $w_i$. The cut goodness is simply the number of uncut nets over the total nets connected. Since $gc_i$ is between 0 and 1, we can take the fuzzy membership $\mu_c$ as equal to the goodness $\mu_c = gc_i$. An example of goodness calculation is shown in Fig. 3; the goodness of cell 5 is calculated as follows: $gc_5 = \frac{3-2}{3} = 0.33$.

**Power Goodness** The power goodness $gp_i$ of a cell is deYned as a measure of how well placed is the cell in its present block according to power consumption and can be computed as follows:

$$gp_i = \frac{\sum_{j=1}^{k} S_j\ (j \in V_i) - \sum_{j=1}^{k} S_j\ (j \in U_i)}{\sum_{j=1}^{k} S_j\ (j \in V_i)} \tag{9}$$

$S_j$ is the switching probability of the cell that drives the net. The goodness is equal to the sum of the switching probabilities of the cells that are driving the uncut nets over the sum of the switching probabilities of the cells that are driving all nets connected. In this way a cell is placed in the partition where the sum of the switching probability of the cut nets is optimized. Results show that this goodness function gives high quality solutions with less power dissipation. Since $0 \leq gp_i \leq 1$ we can take the fuzzy membership $\mu_p = gp_i$. An example of power goodness calculation is shown in Fig. 3; the goodness of cell 5 is calculated as follows: $gc_5 = \frac{0.7-0.4}{0.7} = 0.428$.

The power and cutset objectives are possibly conDicting. Hence it is possible to Ynd alternative solutions for a speciYc circuit. For example, there may exist a solution with high number of cuts and low power consumption (because the nets cut have less switching probability) and another with lower cuts and higher power consumption.

**Delay Goodness** In our problem, we deal with multi-pin nets, which makes it hard to design a suitable and simple delay goodness function. We propose the following delay goodness.

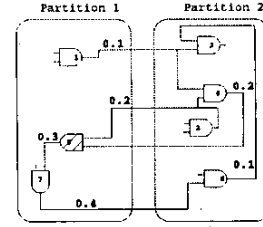$$gd_i = \frac{|K_i| - |L_i|}{|K_i|} \tag{10}$$
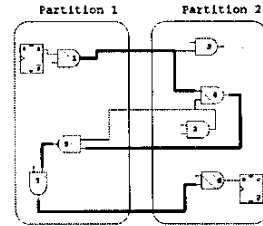


Figure 3: Power and Cut Goodness Calculation.



Figure 4: Delay Goodness Calculation.

where $gd_i$ is the delay goodness of cell $i$. We consider the set of all *critical paths* passing through $i$ and deYne the set $K_i$ as the set of all cells connected to these paths. We also deYne $L_i$ as a subset of $K_i$, containing those cells which are connected to all critical paths passing through $i$ and are not in the same block as $i$. This goodness function will tend to drive the cells that are connected by the critical path to the same block, thus minimizing the delay along the path. A cell is considered good in its block if the majority of cells connected to all critical paths passing through it are also placed in the block. An example for delay computation is given in Fig. 4. To calculate $gd_4$, we Yrst compute $|K_4| = 5$ for the critical path $\{1,4,5,7,6\}$ which is the only one connected to cell 4. $|L_4| = 3$ which are cells $\{1, 5, 7\}$. This gives $gd_4 = \frac{5-3}{5} = 0.4$. However, $gd_5 = 0.6$, and hence is better placed according to the delay consideration.

### 4.2. Proposed Fuzzy Evaluation Scheme and Selection

With the classical goodness of cut only, it is possible that a cell having a high goodness with respect to cut may not be selected even though it is badly placed with respect to circuit delay and power. In order to overcome this problem, it is necessary to include power and delay in the goodness measure along with cut goodness. Also, it is not desirable to select all the cells even if they all have a low goodness value. In this case, it is desirable to select those cells which are far from their lower bounds as compared to other cells in the design. For this purpose, the following fuzzy rule is proposed.

**Rule R1:**     (*as compared to other cells*)
    IF cell $i$ is
        *near its optimal Cut-set goodness*
        AND
        *near its optimal power goodness*
        AND
            *near its optimal net delay goodness*
            OR
            $T_{max}(i)$ *is much smaller than* $T_{max}$
    **THEN** it has a high goodness.

| Circuit | Simulated Evolution SimE | | | | | Genetic Algorithm | | | | | Tabu Search | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | D (ps) | Cut | P(sp) | $\mu(x)$ | Best(s) | D (ps) | Cut | P(sp) | $\mu(x)$ | Best(s) | D (ps) | Cut | P(sp) | $\mu(x)$ | Best(s) |
| S298 | 197 | 11 | 837 | 0.95 | 62 | 233 | 19 | 1013 | 0.79 | 43 | 197 | 24 | 926 | 0.81 | 21 |
| S386 | 393 | 28 | 1696 | 0.74 | 152 | 356 | 36 | 1529 | 0.75 | 151 | 386 | 30 | 1426 | 0.76 | 77 |
| S641 | 886 | 16 | 1738 | 0.98 | 966 | 1043 | 45 | 2355 | 0.83 | 1540 | 889 | 59 | 2281 | 0.85 | 818 |
| S832 | 400 | 39 | 3132 | 0.691 | 257 | 444 | 45 | 3034 | 0.68 | 276 | 446 | 50 | 2731 | 0.682 | 80 |
| S953 | 476 | 48 | 2473 | 0.93 | 249 | 526 | 96 | 2916 | 0.69 | 182 | 466 | 99 | 2518 | 0.734 | 225 |
| S1196 | 415 | 78 | 5488 | 0.82 | 398 | 396 | 123 | 5443 | 0.76 | 373 | 301 | 106 | 4920 | 0.801 | 134 |
| S1238 | 350 | 77 | 5960 | 0.73 | 205 | 475 | 127 | 5713 | 0.72 | 365 | 408 | 79 | 4597 | 0.75 | 160 |
| S1488 | 612 | 83 | 5892 | 0.7 | 716 | 571 | 104 | 5648 | 0.71 | 1183 | 528 | 98 | 5529 | 0.72 | 405 |
| S1494 | 502 | 71 | 6250 | 0.81 | 802 | 614 | 102 | 5474 | 0.70 | 1040 | 585 | 101 | 5339 | 0.71 | 427 |
| S2081 | 325 | 13 | 706 | 0.94 | 89 | 302 | 26 | 787 | 0.73 | 32 | 225 | 17 | 770 | 0.79 | 16 |
| S3330 | 394 | 46 | 8431 | 0.98 | 812 | 571 | 299 | 10358 | 0.75 | 2074 | 533 | 295 | 10298 | 0.79 | 994 |
| S5378 | 554 | 161 | 14094 | 0.95 | 465 | 587 | 573 | 18437 | 0.74 | 2686 | 590 | 430 | 16527 | 0.79 | 1100 |
| S9234 | 831 | 196 | 25672 | 0.98 | 3853 | 1313 | 1090 | 38149 | 0.72 | 5949 | 1052 | 918 | 34055 | 0.81 | 2821 |
| S13207 | 1014 | 313 | 35014 | 0.98 | 3129 | 1399 | 1683 | 45611 | 0.74 | 8097 | 843 | 1332 | 41114 | 0.79 | 3690 |
| S15850 | 1189 | 416 | 40716 | 0.96 | 1850 | 1820 | 2183 | 51747 | 0.74 | 10206 | 1411 | 1671 | 47480 | 0.831 | 5130 |

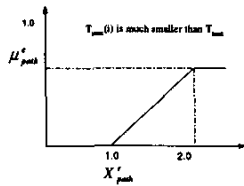Table 1: Comparison between SimE, GA and TS



Figure 5: Membership function for $T_{max}(i) \ll T_{max}$.

Where $T_{max}$ is the delay of the 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. The membership function is illustrated in Fig. 5. In our implementation, the Biasless Selection scheme proposed by Khan et al in [7] is used. The selection bias $B$ is totally eliminated and a cell is selected if $Random > goodness_i$.

## 5. EXPERIMENTAL RESULTS

Table 1 present the results obtained from running the SimE, GA, and TS (implementation details in [8]). Results suggest that the quality of solutions obtained by SimE outperformed both GA and TS in terms of quality of solution and time of execution.Fig. 6 shows the performance of SimE versus TS and GA with respect to time, it can be seen clearly that SimE outperforms both algorithm, in quality and runtime.

## 6. CONCLUSIONS

In this paper, the SimE iterative algorithm for multiobjective VLSI Partitioning is proposed. For that purpose goodness functions for delay and power are designed. Fuzzy logic was used in two levels for the Ýnal cost function as well as the Ýnal goodness funtion. It is used to integrate three objectives namely power, delay, and cutset into a scalar cost value. It is observed that SimE out performs GA and TS in terms of Ýnal solution costs and execution time.

**Acknowledgment:** The authors thank King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia, for support, under project #:COE/ITERATE/221

## 7. REFERENCES

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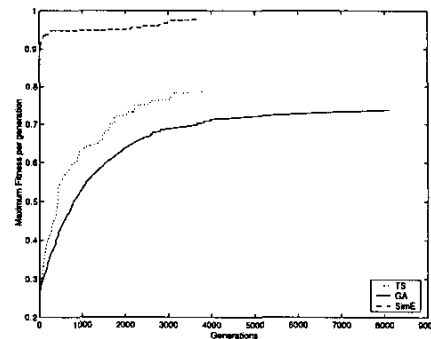


Figure 6: Multi objective SimE, GA and Ts performance for the circuit S13207 against time.

*Europe*, 1995.

[2] K. Shahookar and P. Mazumder. VLSI Cell Placement Techniques. *ACM Computing Surveys*, 2(23):143ñ220, June 1991.

[3] M. Pedram. CAD for Low Power: Status and Promising Directions. *IEEE International Symposium on VLSI Technology, Systems and Applications*, pages 331ñ336, 1995.

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

[5] A. Ghosh, S. Devadas, K. Keutzer, and J. White. Estimation of Average Switching Activity in Combinational and Sequential Circuits. *Design Automation Conference*, pages 253ñ259, 1992.

[6] R. R. Yager. On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decisionmaking. *IEEE Transaction on Systems, MAN, and Cybernetics*, 18(1), January 1988.

[7] Junaid A. Khan, Sadiq M. Sait, and Mahmood R. Minhas. Fuzzy Biasless Simulated Evolution for Multiobjective VLSI Placement. *IEEE CEC 2002, Hawaii USA*, 12-17 May 2002.

[8] R. H. Al-Abaji. Evolutionary Techniques for Multi-objective VLSI Netlist Partitioning. Masterís thesis, King Fahd University of Petroleum and Minerals, Dhahran, Kingdom of Saudi Arabia, May 2002.