ENHANCING PERFORMANCE OF ITERATIVE HEURISTICS FOR VLSI NETLIST PARTITIONING

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

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

ABSTRACT

In this paper we, present a new heuristic called PowerFM which is a modification of the well-known Fidducia Mattheyeses algorithm for VLSI netlist partitioning. PowerFM considers the minimization of power consumption due to the nets cut. The advantages of using PowerFM as an initial solution generator for other iterative algorithms, in particular Genetic Algorithm (GA) and Tabu Search (TS), for multiobjective optimization is investigated. A series of experiments are conducted on ISCAS-85/89 benchmark circuits to evaluate the efficiency of the PowerFM algorithm. Results suggest that this heuristic would provide a good starting solution for multiobjective optimization using iterative algorithms.

1. INTRODUCTION

In recent years, the focus of portable devices has shifted from low throughput devices (e.g., watches, calculators) to high performance devices like notebook computers, cellular phones, etc. Minimizing power is the primary concern for these battery-powered products as for such products longer battery life translates to extended use and better marketability. Exploring the tradeoffs between power, performance, and other objectives during synthesis and physical design is thus demanding more attention.

The optimization for power consumption can be performed at various levels of VLSI design including behavioral level, architectural level, logic level, and physical level. Another compelling reason for the desire of low power consumption is the increasing density of VLSI circuits. The present technology allows integration of tens of millions of transistors on a single chip and the still advancing technology is allowing further high integration. The excessive power consumption of high density circuits results in heating and thus becoming a hindrance towards high integration and hence the feasible packaging of circuits [1, 2]. Also, circuits are operating at much higher clock frequency than before. Therefore, the power dissipation which is a function of clock frequency, is getting significantly prominent. This phenomenon is offering an obstacle in further increase of clock frequency. Due to these reasons, there is an emerging need for minimizing the power requirement of VLSI circuits. For the partitioning phase, two low-power oriented techniques based on Simulated Annealing (SA) algorithm have recently been presented in [3]. An enumerative optimal delay partitioning algorithm targeting low power is proposed by Vainshav et al. in [4].

1.1. FM Partitioning Heuristic

The FM heuristic is a modification of the Kernighan-Lin group migration method for circuit partitioning. In the FM algorithm, all nodes initially in the free set are arranged into a bucket array data structure, in which each bucket contains nodes with the same gain. For each move, the node with the highest gain is considered as the primary candidate to be moved from its current block (From block) to its complementary block (To block). The candidate node must satisfy the balance criterion, used to control the size of subcircuits. If the candidate node does not meet the balance criterion, the node with the next highest gain is selected from the free nodes subset and moved. The moved node is locked and eliminated from the bucket array. The move is completed by modifying the gains of all nodes connected to the critical nets. At the end of a pass, all cells are freed and the process is repeated until we reach a position where no further gain can be achieved. The best partition encountered during the pass is taken as the output of the pass. The number of cells to move is given by the value of k which yields maximum positive gain G_k, where $G_k = \sum_{i=1}^{k} g_i$. Only the cells given by the best sequence, that is $c_1, c_2, ..., c_k$, are permanently moved to their complementary blocks. Then all cells are freed and the procedure is repeated from the beginning. A general description of the heuristic is given in Fig. 1. The best candidate node is defined according to the highest cut-gain associated with moving a node from one subcircuit to another. They are measured using the *net-cut model* [5]. A net is called a *cut net* if it belongs to the current cut set; otherwise, the net is referred to as a *nocut* net. A net is called critical if it is a *cut net* that, as a result of moving a single node, can become a *nocut* net, or vice versa.

```
ALGORITHM  $FM$
  Begin

Step1: Compute gains of cells;
  Step2: $i = 1$;
         Select 'base cell' and call it $c_i$;
         If no base cell Then Exit EndIf;
         A base cell is the one which
            $(i)$ has maximum gain;
            $(ii)$ Satisfies balance criterion;
               IF $tie$ Then use size criterion or
                              Internal connections;
               EndIf;
  Step3: Lock cell $c_i$;
         update gains of cells of those affected critical nets;
  Step4: IF free_cells $\neq \phi$
            Then $i = i + 1$;
            select next base cell $c_i$;
         IF $c_i \neq \phi$ then Goto step 3;
  Step5: Select best sequence of moves $c_1$,$c_2$,...,$c_k$ $(1 \leq k \leq i)$
         such that $G = \sum_{j=1}^{k} g_i$ is maximum;
         ($g_i$ is the gain for cell $c_i$)
         IF tie Then choose subset that achieve a superior balance;
         IF $G \leq 0$ Then Exit;
  Step6: Make all $k$ moves permanent;
            Free all cells;
            Goto Step 1
  End.
```

Figure 1. Fiduccia-Mattheyeses bipartitioning algorithm [5].

The basic concept of min-cut gain calculation provided with the net-cut model can be explained as follows. Let node i_0 be connected to n critical cut nets and to m critical nocut nets. The gain associated with the reassignment of a node i_0 is defined as the difference:

$$G_{i_0} \triangleq n - m \tag{1}$$

In this work, an extension to the FM algorithm which considers optimizing power as the main objective of Partitioning is presented.

2. 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 specified 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 written as follows :

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

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 :

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

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

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

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 [6] 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 ζ is:

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

where β 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:

$$\frac{|\beta_1 - \beta_2|}{\phi} \leq \alpha \tag{6}$$

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

2.1. Overall Fuzzy Cost Function

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

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

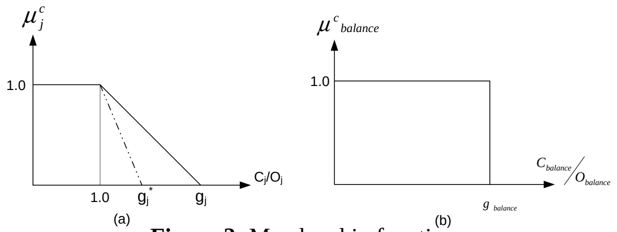

Figure 2. Membership functions

The above rule is translated to *and-like* OWA fuzzy operator [7] and the membership $\mu(x)$ of a solution x in fuzzy set *good solution* is given as:

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

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 " within acceptable *power*", "within acceptable *delay*", "within acceptable cut-set" and "within acceptable *balance*" respectively. β^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's 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.

3. POWERFM HEURISTIC

The PowerFM is a modification of the FM algorithm which seeks minimization of the power consumption due to the cut. All concepts of the FM are maintained, the major difference is that we are calculating the gain due to the sum of the switching probabilities of the cut nets. Also some other necessary modifications are done in some parts of the Algorithm that we will discuss in what follows.

3.1. Power Gain Calculation

The power gain for a cell i is calculated using Eqn. 8. X_i is the set of critical cut nets. U_i is the set of critical uncut net.

```
ALGORITHM $Compute\,Cell\,gains$;
Begin
    For each free cell $i$ Do
    $g(i) \leftarrow 0$ ;
    F $\leftarrow$ $From\,block$ of cell $i$
    T $\leftarrow$ $To\,block$ of cell $i$
        FOR each net $n$ on cell $i$ DO
            If $F(n) = 1$
            Then $g(i) = g(i) + (C_{off} \times Sw\,prob\,of\,driving\,net)$
            (Cell i is the only cell in the From block connected to net n.)
            If $T(n) = 0$
            Then $g(i) = g(i) - (C_{off} \times Sw\,prob\,of\,driving\,net)$
            (All of the cells connected to net n are in the From block.)
        EndFor
    EndFor
End.
```

Figure 3. Procedure to compute gains of free cells.

$$Pgain(i) = C_{off}\left(\sum_{j \in X_i} S_j - \sum_{j \in U_i} S_j)\right) \qquad (8)$$

In each pass, the gain of every free cell is updated according to the Compute Gain Algorithm shown in Fig. 3. Let $F(n)$ be the number of cells connected to net n in the From block (current block) of the moved cell i. Let $T(n)$ be the number of cells connected to net n in the To block (destination block) of the moved cell i. When computing the gain we consider only the *critical nets*; A net is critical if it has a cell which if moved will change its cutsate. That is if and only if $F(n)$ is 1, or $T(n)$ is 0.

The algorithm is simple and it checks if the net is critical and if $F(n) = 1$ then moving cell i will increase the gain by $C_{off} \times Sw\,prob\,of\,driving\,net$, and if $T(n) = 0$ then moving the cell i will decrease the gain by $C_{off} \times Sw\,prob\,of\,driving\,net$.

3.2. GA and TS

Genetic Algorithm is an elegant search technique that emulates the process of natural evolution as a means of progressing towards the optimal solution. The algorithm starts with a set of initial solutions called *population* that is generated randomly. In each iteration (*known as generation in GA terminology*), all the individual chromosomes in the population are evaluated using a *fitness function*. Then, in the *selection* step, two of the above chromosomes at a time are selected from the population. The individuals having higher fitness values are more likely to be selected. After the selection step, different operators namely *crossover, mutation* act on the selected individuals for evolving new individuals called *offsprings*.

Tabu Search starts from an initial feasible solution and carries out its search by making a sequence of random moves or perturbations. A Tabu list is maintained which stores the attributes of a number of previous moves. This list prevents taking the search process back to recently visited states. In each iteration, a subset of neighbor solutions is generated by making a certain number of moves and the best move (the move that resulted in the best solution) is accepted, provided it is not in the Tabu list. Otherwise, if the said move is in the Tabu list, it is accepted only if it leads to a solution better than the best solution found so far (aspiration criterion). Thus, the aspiration criterion can override the Tabu list restrictions. The solution encoding and initialization steps are similar to those described above for GA. These two multiobjective optimization iterative algorithms (GA and TS) for VLSI Partitioning were proposed in [8], [9], [10].

4. EXPERIMENTAL RESULTS

A series of experiments were performed on ISCAS-85/89 benchmark circuits, the results are analyzed and reported in this section. Table 1 shows a comparison of results of TS and GA when the initial solution is chosen from random or provided by PowerFM for both GA and TS algorithms. Table 1 shows also the results obtained from PowerFM when used on its own. P_{avg} refers to the average power of the results obtained from 100 runs of the PowerFM. The notation in Table1 is as follows: $D(ps)$ stands for Delay and it is measured in pico-seconds, Cut is the number of nets cut, $P(sp)$ is the power dissipation measured in terms of switching probability, $T(s)$ is the total time taken by the whole run for PowerFM.

When starting from random solution it was observed that TS outperforms GA in terms of final solution costs and execution time. These two algorithms are complex, and relatively take more execution time than PowerFM. The idea of using the PowerFM as a starting solution for iterative algorithms is relevant because PowerFM proved to be an extremely fast algorithm compared to GA and TS (at least 100 times faster), with reasonable performance. This will save a lot of time for algorithms like GA and TS where the converging rate is slow. Furthermore results, showed that GA and TS were able to improve solutions provided by PowerFM. Fig. 4 and Fig. 5 show the performance of GA and TS respectively when applied to the circuit s1488 when starting from an initial solution provided by PowerFM. In Fig. 4 and Fig. 5, (a) shows the number of nets cut, (b) shows the longest path delay of the circuit in pico-seconds, (c) shows the power dissipation, (d) shows the cell difference between the two partitions, (e) shows the average generation fitness, (e) shows the Best solution fitness. Both GA and TS show an improvement in terms of the overall quality of solution.

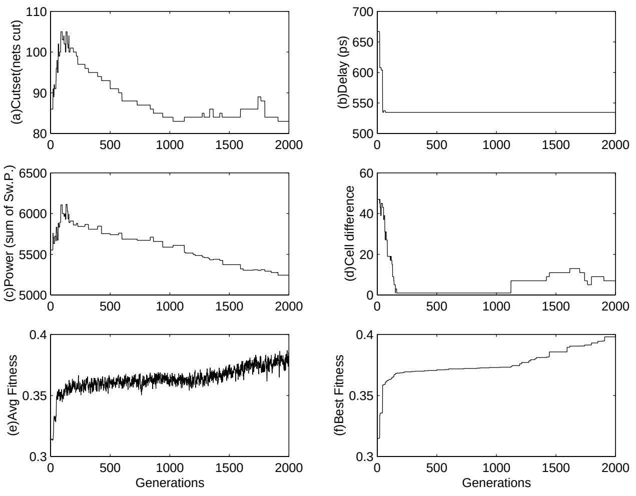

Figure 4. Genetic Algorithm starting from PowerFM for circuit s1488.

It can be noted that for most of the circuits when using a starting solution provided by PowerFM the results are better than when starting with random solution in terms of quality of solution. An important point to notice also is that although when starting from random TS performed better than GA; when starting from PowerFM GA proves to be more efficient than TS. This is due to the fact that GA starting from a good solution has the ability to inherent the good characteristics and improve on it and proved to be able to benefit more than TS when starting from a good solution provided from PowerFM. This is noted when the results of TS (starting from PowerFM) and GA (starting from PowerFM) in Table 1 are compared; it can be seen that GA is better for large circuits (s3330 ... s15850) in terms of power and cutset. The results proved that it is beneficial to use PowerFM as a starting solution for multiobjective GA and TS. Moreover looking at the results of PowerFM alone, it comparably provided

Table 1. Start from PowerFM versus Random Start for GA and TS.

	GA Random Start			GA Start From PowerFM			TS Random Start			TS Start From PowerFM			PowerFM				
Circuit	D(ps)	Cut	P(sp)	D(ps)	Cut	P(sp)	D(ps)	Cut	P(sp)	D(ps)	Cut	P(sp)	D(ps)	Cut	P(sp)	T(s)	P_{avg}
S298	233	19	1013	191	10	921	197	24	926	189	10	849	301	20	732	0.05	828
S386	356	36	1529	345	31	1401	386	30	1426	333	27	1264	434	29	1511	0.39	1673
S641	1043	45	2355	861	43	2343	889	59	2281	844	48	2476	1221	44	1667	0.61	1773
S832	444	45	3034	441	42	3032	446	50	2731	431	40	3135	527	51	2855	1.97	3338
S953	526	96	2916	465	89	3012	466	99	2518	430	85	2999	902	120	2191	0.60	2422
S1196	396	123	5443	390	86	4921	301	106	4920	335	77	4823	612	68	4116	1.81	5289
S1238	475	127	5713	461	91	5702	408	79	4597	401	74	5190	544	62	4281	1.80	5358
S1488	571	104	5648	541	83	5248	540	110	6300	521	94	6005	724	70	5228	5.60	5787
S1494	614	102	5474	601	97	5123	585	101	5339	534	95	5058	630	80	5354	7.19	6022
S2081	302	26	787	260	15	740	225	17	770	244	12	704	335	7	565	0.11	586
S3330	571	299	10358	435	203	9296	533	295	10298	419	257	9288	593	226	9522	6.37	10180
S5378	587	573	18437	442	423	15356	590	430	16527	432	400	15319	574	363	14565	19.22	15453
S9234	1313	1090	38149	856	375	28305	1052	918	34055	835	705	31837	832	389	26784	92.50	29100
S13207	1399	1683	45611	951	750	39620	843	1332	41114	823	1310	40235	1286	929	37190	273	39155
S15850	1820	2183	51747	1350	851	43680	1411	1671	47480	1210	1332	45320	1464	919	42521	318.56	43238

impressive results in terms of Power and cutset considering that its main aim is to optimize power.

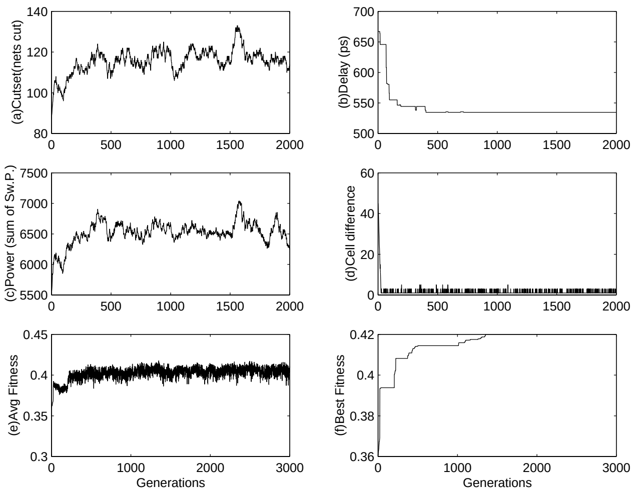

Figure 5. Tabu Search algorithm starting from PowerFM for circuit s1488.

5. CONCLUSIONS

In this paper, we proposed a new modification to the FM algorithm PowerFM which targets power optimization. The possibility to use the algorithm as a provider for initial solution for other iterative multiobjective algorithms in particular GA and TS was investigated. GA performed better than TS when starting from a solution rovided by PowerFM. PowerFM results where important due to its speed and good quality of the final solution. A series of experiments were performed, analyzed and reported to evaluate the efficiency of the algorithm. Results suggest that the algorithm proved to be efficient for optimizing power, and would provide a good starting solution for the multiobjective optimization using Genetic and Tabu search partitioning algorithms.

Acknowledgment

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

6. REFERENCES

[1] M. Pedram. CAD for Low Power: Status and Promising Directions. *IEEE International Symposium on VLSI Technology, Systems and Applications*, pages 331–336, 1995.

[2] U. Narayanan, G.I. Stamoulis, and R. Roy. Characterizing Individual Gate Power Sensitivity in Low Power Design. *12th International Conference on VLSI Design*, pages 625–628, January 1999.

[3] I.S. Choi and S.Y. Hwang. Circuit Partitioning algorithm for Low-Power Design Under Area Constraints Using Simulated Annealing. *IEE Proc. Cicrcuits Devices Systems*, 146(1):8–15, February 1999.

[4] H. Vaishnav and M. Pedram. Delay optimal partitioning targeting low power VLSI circuits. *IEEE Trans. on Computer Aided Design*, 18(6):298–301, june 1999.

[5] C. M. Fiduccia and R. M. Mattheyses. A Linear-Time Heuristic for Improving Network Partitions. pages 175–181, 1982.

[6] A. Ghosh, S. Devadas, K. Keutzer, and J. White. Estimation of Average Switching Activity in Combinational and Sequential Circuits. *Design Automation Conference*, pages 253–259, 1992.

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

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

[9] Sadiq M. Sait, Aiman El-Maleh, and Raslan Al-Abaji. Simulated evolution algorithm for multiobjective VLSI netlist bi-partitioning. *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS)*, V:457–460, May 2003.

[10] Sadiq M. Sait, Aiman El-Maleh, and Raslan Al-Abaji. General iterative heuristics for VLSI multiobjective partitioning. *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS)*, V:497–500, May 2003.