Performance Driven Multiway Partitioning*

Jason Cong and Sung Kyu Lim
UCLA Department of Computer Science, Los Angeles, CA 90095
{cong,limsk}@cs.ucla.edu

Abstract— Under the interconnect-centric design paradigm, partitioning is seen as the crucial step that defines the interconnect [1]. To meet the performance requirement of today's complex design, performance driven partitioners must consider the amount of interconnect induced by partitioning as well as its impact on performance. In this paper, we provide new performance driven formulation for cell move based top-down multiway partitioning algorithms with consideration of the local and global interconnect delay. In our "constrained acyclic partitioning" formulation, cell moves are restricted to maintain acyclicity in partitioning solution to prevent cyclic dependency among cells in different partitions. In our "relaxed acyclic partitioning" formulation, acyclic constraints are relaxed to give partitioners capability of minimizing cutsize and delay. Our new acyclic constraint based performance driven multiway partitioning algorithm FLARE obtains (i) 4% to 13% better delay compared to the state-of-the-art cutsize minimization based hMetis [10] at almost no increase in cutsize, and (ii) 84% better cutsize compared to the state-of-the-art delay minimization based PRIME [2] at an expense of 16% increase in delay.

I. Introduction

Many sources including NTRS (National Technology Roadmap for Semiconductor) predict that 80% or more of the critical path delay will be directly linked to interconnect in deep submicron geometries. Thus, addressing interconnect issues in all steps involved in VLSI design process has become an essential goal. Under the interconnect-centric design paradigm, partitioning is seen as the crucial step that defines the interconnect [1]. To meet the performance requirement of today's complex design, performance driven partitioners must consider the amount as well as performance-related quality of the interconnect induced by partitioning. Cutsize minimization helps to lower the possibility of critical paths crossing partition boundary multiple times, thus improving performance. In addition, a proper model of delay estimation for partitioning has direct impact on delay minimization. Many proposed cutsize oriented partitioners do not consider delay, while many proposed delay oriented partitioners do not consider cutsize. As a result, there is a strong need for a performance driven partitioner that considers both cutsize and delay and provides smooth cutsize/delay tradeoff.

The performance driven circuit decomposition algorithms can be grouped into two categories: bottom-up clustering and top-down partitioning algorithms. Performance driven bottom-up circuit clustering problem is to group components in a circuit into clusters subject to an upper bound on the total area and/or total I/O pins in each cluster. The objective is to minimize the delay of the circuit. In [11], the authors proposed an efficient labeling based clustering algorithm to achieve the minimum delay for combinational circuits under simplistic delay model. [15, 13, 18] extend this work to consider more general delay model. Pan et al. [16] proposed a polynomial-time clustering algorithm for sequential circuits with retiming that achieves quasi-optimal delay under general delay model. The current state-of-the-art is established by PRIME [2] that provides significant space and time complexity improvement of [16] while maintaining quasi-optimal delay solutions. However, these methods face one or both of the following limitations: (i) they produce much worse cutsize compared to conventional top-down partitioning, which in turn translates into more routing area and congestion problem, (ii) it is hard to control area balance among partitions and sometimes fail to obtain exact number of partitions.

Performance driven top-down circuit partitioning problem has been studied actively especially during recent years. The problem is to divide a circuit into predetermined number of partitions while maintaining the area of each partition within user specified range. Unlike the conventional top-down partitioning algorithms [8, 10] that minimize cutsize only, the primary objective of performance driven algorithms is to minimize the delay of the circuit. Shih et al. [17] proposed an algorithm to guarantee that the delay between registers satisfies the timing constraint. Hwang and Gamal [9] showed that logic replication from one partition to another can improve cutsize and delay. Liu et al. [14] proposed an efficient algorithm to combine logic replication and retiming for bipartitioning. However, these algorithms may suffer a long runtime for large circuits and do not guarantee any optimality.

In this paper, we provide new performance driven for-

*This research is partially supported by the MARCO/DARPA Gigascale Silicon Research Center (GSRC) and Fujitsu Laboratories of America under the California MICRO Program.

mulation for cell move based top-down multiway partitioning algorithms with consideration of the local and global interconnect delay induced by the partitioning. In our *constrained acyclic partitioning* formulation, cell moves are restricted to maintain acyclicity in partitioning solution to prevent cyclic dependency among cells in different partitions. In our *relaxed acyclic partitioning* formulation, acyclic constraints are relaxed to give partitioners capability of minimizing cutsize and delay. Our new acyclic constraint based performance driven multiway partitioning algorithm `FLARE` obtains (i) 4% to 13% better delay compared to the state-of-the-art cutsize minimization based `hMetis` [10] at almost no increase in cutsize, and (ii) 84% better cutsize compared to the state-of-the-art delay minimization based `PRIME` [2] at the expense of 16% increase in delay.

The remainder of the paper is organized as follows. Section II presents problem formulation. Section III presents partitioning with acyclic constraints. Section IV provides experimental results. Section V concludes the paper with our ongoing research.

II. Problem Formulation

A sequential gate-level circuit netlist NL can be represented as a directed retiming graph $G(V, E)$ where V is the set of nodes representing gates in the circuit, and E is the set of edges representing the connections between gates. A directed edge $e(u, v)$ denotes the connection from gate u to gate v. A fan-in set of vertex v is defined as $FI(v) = \{u | u \in V \text{ and } e = (u, v) \in E\}$, and fan-out set of vertex v is similarly defined as $FO(v) = \{u | u \in V \text{ and } e = (v, u) \in E\}$. A set of primary inputs is defined as $PI = \{v | v \in V \text{ and } FI(v) = \emptyset\}$, and a set of primary outputs is defined as $PO = \{v | v \in V \text{ and } FO(v) = \emptyset\}$.

A balanced duplication free K-way partitioning $B = \{B_1, B_2, \cdots, B_K\}$ of given $G(V, E, W)$ satisfies the following conditions:

- $B_i \cap B_j = \emptyset$ for $i \neq j$ and $B_1 \cup B_2 \cup \cdots \cup B_K = V$
- $\alpha_i \leq |B_i| \leq \beta_i$ for given α_i and β_i, $1 \leq i \leq K$

We build *dependency graph* $D(G, B)$ [4] as follows; each vertex in $D(G, B)$ represents a partition in B, and a directed edge (B_i, B_j) exists if there exists an edge $e = (x, y) \in E$ such that $x \in B_i$ and $y \in B_j$. We call partitioning solution B *acyclic* if its dependency graph $D(G, B)$ is a directed acyclic graph.

We measure *delay* for given partitioning solution B of a sequential circuit, denoted $\phi(B)$, for performance evaluation of B. In *general delay model* [15, 16, 2], each node v has a delay of $d(v)$, and each edge $e(u, v)$ has a delay of $d(e)$ defined as follows;

$$d(e) = \begin{cases} D & \text{if } e = (u, v) \in E,\ u \in B_i,\ v \in B_j,\ i \neq j \\ 0^1 & \text{if } e = (u, v) \in E \text{ and } u, v \in B_i \end{cases}$$

The delay of a path $p = (u \rightarrow v)$ from $u \in V$ to $v \in V$, denoted $d(p)$, is defined to be the sum of $d(e)$ and $d(g)$ along p. $\phi(B)$ is the longest path delay among all combinational paths of one of the following types; $PI \rightarrow PO$, $PI \rightarrow FF$, $FF \rightarrow PO$, or $FF \rightarrow FF$. The *delay ratio* corresponds to $d(e)/d(g)$, which is equivalent to D in case we assume (i) $d(g) = 1$, and (ii) e connects vertices in different partitions. The delay ratio serves as a first-order approximation of how big global interconnect delay is compared to local interconnect delay.

III. Partitioning with Acyclic Constraints

We present our multiway partitioning algorithm `FLARE` that simultaneously considers cutsize and delay minimization under new acyclic constraint based formulation. In the constrained acyclic partitioning formulation, cell moves are restricted to maintain acyclicity in partitioning solution to prevent cyclic dependency among cells in different partitions. In the relaxed acyclic partitioning formulation, acyclic constraints are relaxed to give partitioners capability of optimizing delay. `FLARE` performs relaxed acyclic formulation based `xLR` bipartitioning algorithm on top of two-level cutsize oriented `ESC` [6] cluster hierarchy. This is then used as the bipartitioning engine for pairwise movement based multiway partitioning framework `PM` [5].

A. Constrained Acyclic Partitioning

Assuming topological ordering of V in $G(V, E)$ is from partition B_f to B_t, we define the *backward edge set* $V = \{e | e = (x, y) \in E,\ x \in B_t,\ y \in B_f\}$. Then $B = (B_f, B_t)$ is acyclic iff $V = \emptyset$. We define *A-counter* for each vertex $x \in V$, denoted $a(x)$, as follows;

$$a(x) = \begin{cases} |\{y | y \in FO(x) \text{ and } y \in B_f\}| & \text{if } x \in B_f \\ |\{y | y \in FI(x) \text{ and } y \in B_t\}| & \text{if } x \in B_t \end{cases}$$

The acyclic constraint will be violated, i.e. $V \neq \emptyset$, if x is moved to the other partition when $a(x) > 0$. In other words, we can only move a cell x with $a(x) = 0$ under acyclic constraint. This is an additional constraint imposed on cell moves other than the conventional area balance constraint. An illustration of A-counters is shown in Figure 1-(a). If δ denotes the maximum degree among vertices in V, the computation of $a(x)$ takes $O(\delta)$ since it requires to examine all its neighbors. In addition, A-counters range from 0 to δ.

A naive way to incorporate A-counter based acyclic constraints into `FM` would be to find next legal cell move by searching down the bucket to obtain the first maximum gain cell x that has $a(x) = 0$. However, this is a very inefficient approach since $O(\delta)$ computation of A-counter is required for *all* cells examined before we find the first

[1] Local interconnect delay can be estimated and its average can be lumped into the gate delay $d(v)$ for simplicity.

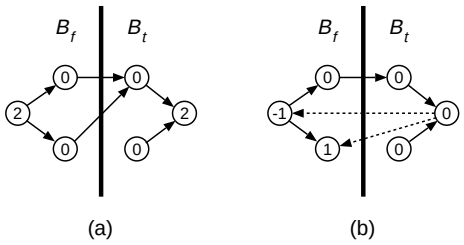

Fig. 1. Illustration of A-counters and R-counters. Numbers in each circle denotes the counters, and backward edges are shown in dotted lines. Topological ordering is from partition B_f to B_t. (a) A-counters that represent number of backward edges introduced upon cell move, (b) R-counters that represent reduction of backward edges upon cell move.

legal cell move. In order to maintain the linear complexity of single pass of FM under acyclic constraints, we provide two schemes. First, we precompute A-counters for all vertices before the cell move begins and incrementally update them upon each cell move. Second, in order to avoid searching down the bucket to obtain unconstrained cells, we use *hybrid bucket* that arranges unconstrained cells in LIFO order by inserting at the head of the list, while arranging constrained cells in FIFO order by inserting at the tail. The hybrid bucket structure also maintains counters at the header to keep track of how many unconstrained cells remain at each bucket.

B. Relaxed Acyclic Partitioning

One major drawback of partitioning under acyclic constraints is the restriction of cell moves due to the existence of acyclic constraints imposed on cells. We sacrifice cutsize by ignoring many beneficial cell moves in order to maintain acyclicity in the partitioning solution. As it will become evident from related experiments, cutsize degradation has negative effect on delay minimization. However, if we treat acyclicity to be an object to be optimized instead of a constraint to be satisfied at all times, we can optimize both cutsize and delay. Assuming topological ordering of V in $G(V, E)$ is from partition B_f to B_t, we use the size of backward edge set $|V| = |\{e|e = (x, y) \in E,\ x \in B_t,\ y \in B_f\}|$ to represent the *degree* of acyclic constraint violation. Then, we try to minimize $|V|$ instead of requiring V to be $\emptyset$.

We extend the definition of A-counter in previous section to formulate the reduction in $|V|$. Assuming B_f to B_t topological ordering of V, we define *R-counter*, denoted $r(x)$, for each vertex $x \in V$ as follows;

$$r(x) = \begin{cases} |\{y|y \in FI(x) \text{ and } y \in B_t\}| - a(x) & \text{if } x \in B_f \\ |\{y|y \in FO(x) \text{ and } y \in B_f\}| - a(x) & \text{if } x \in B_t \end{cases}$$

$r(x)$ represents the reduction in $|V|$ if x is moved to the other partition. An illustration of R-counters is shown in Figure 1-(b). Since acyclic constraint is relaxed, every cell move is legal if it does not violate the conventional area balance constraint. In addition, we can use the conventional LIFO bucket structure to manage cell gains. The computation and incremental update of R-counters can be done in a similar way as A-counters.

Note that R-counters do not represent real hyperedge cutsize reduction that FM [8] tries to minimize. Therefore, we incorporate R-counters into FM based cutsize gain formulation in the following way; if $g(x)$ denotes conventional FM cutsize reduction gain, our new *hybrid gain* becomes;

$$h(x) = \alpha \cdot g(x) + \beta \cdot r(x)$$

where α and β serve as weighting constants. $h(x)$ represents real reduction in both hyperedge cutset and backward edge set if $\alpha = \beta = 1$, which is our empirical choice. We perform cell moves based on $h(x)$ and update $g(x)$ and $r(x)$ together while visiting neighboring cells of x. Note that the partitioner performs the conventional cutsize oriented moves if $r(x) = 0$, which in turn indicates that $r(x)$ component has an effect of altering the conventional move sequence towards better delay result. It is possible to obtain cyclic partitioning result, i.e. $|V| > 0$, at the termination of partitioning. However, relaxed acyclic partitioning enables the partitioner to optimize cutsize without any restriction and at the same time prevent critical paths in the given circuit from being cut multiple times if not once in (B_f, B_t).

C. Summary of FLARE Algorithm

Our performance driven multiway partitioning algorithm FLARE first builds two-level cluster hierarchy with ESC [6] bottom-up clustering algorithm. Then, we adopt *two-phase* top-down partitioning scheme to integrate clustering into partitioning: we perform our performance driven xLR partitioning first on the clustered circuit for global optimization and then on the declustered circuit for local refinement. We develop xLR algorithm by incorporating two types of enhancement to FM [8] algorithm, (i) LR [3] for cutsize minimization, (ii) relaxed acyclic partitioning based R-counter $r(x)$ discussed in Section III-B for simultaneous cutsize and delay minimization. We provide comprehensive experimental justification for each enhancement in xLR in the following Section IV-B. Lastly, the two-phase partitioning algorithm based on the combination of xLR and ESC is used as the bipartitioning engine for pairwise movement based multiway partitioning framework PM [5]. Interested readers are referred to [5] for more details on PM framework.

An overview of FLARE algorithm is shown in Figure 2. FLARE maintains (i) undirected graph U for executing ESC clustering algorithm, (ii) directed graph G_0 and G_1 from which relaxed acyclic constraint based $r(x)$ is computed, and (iii) hypergraph H_0 and H_1 from which conventional cutsize gain $g(x)$ is computed (line 1 to 3). The cluster-

```
FLARE(NL, K)
Input: sequential circuit netlist NL, # of block K
Output: partition B, cutsize c(B), and delay φ(B)
1.  obtain undirected graph U from NL;
2.  obtain directed graph G0 from NL;
3.  obtain hypergraph H0 from NL;
4.  C = ESC(U);
5.  obtain G1 and H1 from C;
6.  remove FF from G0;
7.  T = topological sorting of (G1);
8.  B1 = split T into K parts;
9.  while (gain > 0)
10.     obtain block pairing with PM;
11.     compute g(x) from H1;
12.     compute r(x) from G1;
13.     while (exists legal cell move)
14.         perform cell move;
15.         update g(x) and r(x);
16.     update B1;
17.     project B1 to B;
18.     compute g(x) from H0;
19.     compute r(x) from G0;
20.     while (exists legal cell move)
21.         perform cell move;
22.         update g(x) and r(x);
23.     update B;
24. compute c(B) from H0;
25. compute φ(B) from G0;
26. return B, c(B), and φ(B)
```

Fig. 2. Overview of performance driven partitioning algorithm FLARE. FLARE performs relaxed acyclic formulation based xLR bipartitioning algorithm on top of two-level cutsize oriented ESC [6] cluster hierarchy. This is then used as the bipartitioning engine for pairwise movement based multiway partitioning framework PM [5].

ing solution C from ESC algorithm is used to contract G_0 and H_0 to obtain G_1 and H_1 (line 4 and 5). Under the assumption that all cycles in G involve flip-flops, we convert G_0 into directed acyclic graph by removing flip-flop set FF from G_0 (line 6). We obtain the initial acyclic K-way partition B_1 from topological sorting of vertices in G_1 (line 7 to 8). Our relaxed acyclic partitioning algorithm xLR is applied on the clustered netlist (line 11 to 16) and then on the original netlist (line 18 to 23) according to two-phase partitioning framework.

IV. Experimental Result

A. Experimental Setting

We implemented our algorithms in C++/STL and tested on SUN ULTRA SPARC60 at 360Mhz. We obtained the latest binary executable of hMetis [10] (v1.5.3) from the website for the evaluation. The benchmark set consists of 7 ISCAS circuits and 4 large scale industrial designs provided by our industrial sponsor. Table I shows the characteristics of these circuits. We report cutsize, delay, and runtime from 16-way partitioning results. All algorithms mentioned in this section obtain 16 partitions by recursively applying bipartitioning, except for LR/ESC-PM [6] and FLARE that obtain 16 partitions simultaneously using PM [5] framework. The bipartitioning area balance skew is set to [.45, .55], which is equivalent to [$0.45^4 = 0.041$, $0.55^4 = 0.092$] for the 16-way partitioning. Runtimes are measured in seconds, and cutsizes are based on Cost-1 metric that counts the number of hyperedges that span more than single partition. We assume that all gates have unit area and unit delay, while primary inputs, primary outputs and flip-flops have no area and no delay. We apply retiming [12] on all algorithms used in our experiments as a post delay refinement process. We use $D = 5$ for the current $0.18\mu m$ technology throughout the entire experiments unless specified otherwise.

TABLE I
Benchmark circuit characteristics. #GA, #PI, #PO, #FF, and #net respectively denote the total number of gates, primary inputs, primary outputs, flip-flops, and nets in each circuit.

name	#GA	#PI	#PO	#FF	#net
s9234	1290	28	39	135	1492
s5378	1443	35	49	163	1690
s13207	3146	59	152	486	3843
s15850	3784	76	150	515	4525
bigkey	8599	228	197	224	9248
s38584	13209	38	304	1423	14974
clma	30552	61	82	33	30728
ind1	29780	2630	3242	603	36255
ind2	26060	2772	6242	1755	36829
ind3	52197	2801	3070	2001	60069
ind4	101531	4155	4547	8333	118566

B. Overall Comparison

We use two existing cutsize oriented cell move based partitioning algorithms FM [8] and LR [3] to evaluate our constrained and relaxed acyclic partitioning formulation. FM is the standard cell move based algorithm, whereas LR is an enhancement of FM based on modified gain function. The naming convention of algorithms we test is of [c|x][FM|LR] form. The prefix [c|x] respectively denotes constrained and relaxed acyclic constraint partitioning. The objective of adding acyclic constraints is to improve delay at the expense of cutsize increase, while relaxing acyclic constraint is to release restriction on cell moves to minimize both cutsize and delay. Indeed, we observe the corresponding trends from Table II. We note (i) constrained acyclic partitioners cFM and cLR improve delay while degrading cutsize, (ii) relaxed acyclic partitioners xFM and xLR improve bad cutsize results by cFM and cLR while maintaining delay results.

Table III reveals the overall cutsize and delay comparison among (i) cutsize oriented FM, hMetis, and

TABLE II

CUTSIZE, DELAY, AND RUNTIME REDUCTION TRENDS ON 16-WAY PARTITIONING RESULT OF CONSTRAINED AND RELAXED ACYCLIC PARTITIONING ALGORITHMS. THE CUTSIZE RESULT IS BASED ON COST-1 METRIC, AND THE DELAY RATIO D IS SET TO 5. TIME DENOTES TOTAL RUNTIME INCLUDING PARTITIONING AND RETIMING IN SECONDS.

	FM based Partitioning						LR based Partitioning					
	FM		cFM		xFM		LR		cLR		xLR	
ckt	cut	dly	cut	dly	cut	dly	cut	dly	cut	dly	cut	dly
s9234	148	35	532	44	351	50	153	35	512	44	172	35
s5378	201	33	778	42	453	38	213	30	743	42	302	32
s13207	238	65	1610	65	679	61	204	55	1412	65	405	65
s15850	305	67	1692	68	711	58	263	61	1322	61	471	61
bigkey	78	22	5577	37	946	28	37	20	4547	15	127	20
s38584	475	52	7262	59	1972	51	314	49	5232	39	569	38
clma	1031	89	7576	110	2342	98	583	77	6571	110	835	90
ind1	1842	474	5146	416	3120	417	1145	427	4182	411	1542	421
ind2	1535	55	11341	71	3088	66	1195	56	9343	71	1689	58
ind3	4415	785	15989	599	6739	667	2440	720	11341	593	3444	667
ind4	6887	124	47991	109	13239	124	3105	104	34546	107	4463	96
TOTAL	17155	1801	105494	1620	33640	1658	9652	1634	79751	1558	14019	1583
TIME	3372		1234		2246		3864		1453		2754	

LR/ESC-PM, (ii) delay oriented PRIME [2] and our FLARE algorithm. FLARE performs relaxed acyclic formulation based xLR bipartitioning algorithm on top of two-level cutsize oriented ESC [6] cluster hierarchy. This is then used as the bipartitioning engine for pairwise movement based multiway partitioning framework PM [5]. First of all, FLARE obtains (i) better delay compared to the state-of-the-art cutsize oriented hMetis [10] at almost no increase in cutsize, and (ii) significantly better cutsize compared to the state-of-the-art quasi-optimal delay oriented PRIME at the expense of 16% increase in delay. Secondly, the conventional cutsize minimization partitioning hMetis improves both the cutsize and delay of FM. This illustrates the side-effect of cutsize minimization objective on delay minimization. However, we show that the delay of hMetis can still be further improved by FLARE while maintaining comparable cutsize quality.

The global *vs* local interconnect delay ratio D is expected to increase as the technology advances into deeper sub-micron. Then, the delay advantage of FLARE over hMetis is expected to increase from 4% to 13% when D increases from 5 (estimated for $0.18\mu m$ technology) to 16 (estimated for $0.07\mu m$ technology). Interested readers are refered to our technical report for more details [7].

V. CONCLUSION AND ONGOING WORK

In this paper, we provided two new performance driven formulations for cell move based top-down multiway partitioning algorithms for sequential circuits; constrained and relaxed acyclic partitioning. The objective of adding acyclic constraints was to improve delay at the expense of cutsize increase, while relaxing acyclic constraint was to release restriction on cell moves to minimize cutsize and delay. We develop an efficient multiway partitioning algorithm FLARE that simultaneously considers cutsize and delay minimization under new acyclic constraint based formulation. Our ongoing study includes performance driven mincut placement based on our new performance driven formulation.

REFERENCES

[1] J. Cong. An interconnect-centric design flow for nanometer technologies. In *Proc. of Int'l Symp. on VLSI Technology, Systems, and Applications*, pages 54–57, 1999.

[2] J. Cong, H. Li, and C. Wu. Simultaneous circuit partitioning/clustering with retiming for performance optimization. In *Proc. Design Automation Conf.*, 1999.

[3] J. Cong, H. P. Li, S. K. Lim, T. Shibuya, and D. Xu. Large scale circuit partitioning with loose/stable net removal and signal flow based clustering. In *Proc. Int. Conf. on Computer-Aided Design*, pages 441–446, 1997.

[4] J. Cong, Z. Li, and R. Bagrodia. Acyclic multi-way partitioning of boolean networks. In *Proc. Design Automation Conf.*, pages 670–675, 1994.

[5] J. Cong and S. K. Lim. Multiway partitioning with pairwise movement. In *Proc. Int. Conf. on Computer-Aided Design*, pages 512–516, 1998.

[6] J. Cong and S. K. Lim. Edge separability based circuit clustering with application to circuit partitioning. In *Proc. Asia South Pacific Design Automation Conf.*, 2000.

[7] J. Cong, S. K. Lim, and C. Wu. Performance-driven multi-level and multi-way partitioning. Technical Report 990046, CS Dept. of UCLA, Oct. 1999.

TABLE III

CUTSIZE AND DELAY COMPARISON AMONG VARIOUS PARTITIONING ALGORITHMS INCLUDING (I) CUTSIZE ORIENTED FM [8], hMetis [10], AND LR/ESC-PM [6], (II) DELAY ORIENTED PRIME [2], AND OUR FLARE ALGORITHM. RATIO IS RELATIVE TO CUTSIZE AND DELAY RESULTS OF FLARE. THE CUTSIZE RESULT IS BASED ON COST-1 METRIC, AND THE DELAY RATIO D IS SET TO 5. TIME DENOTES TOTAL RUNTIME INCLUDING CLUSTERING (IF ANY), PARTITIONING, AND RETIMING IN SECONDS.

	Cutsize Minimization						Delay Minimization			
	FM		hMetis		LR/ESC-PM		PRIME		FLARE	
ckt	cut	dly	cut	dly	cut	dly	cut	dly	cut	dly
s9234	148	35	145	30	146	35	359	35	139	35
s5378	201	33	193	29	202	28	327	27	185	29
s13207	238	65	203	55	174	50	414	45	172	50
s15850	305	67	226	57	201	59	762	53	218	63
bigkey	78	22	38	12	37	10	761	12	44	10
s38584	475	52	294	38	302	39	1860	34	283	39
clma	1031	89	332	84	333	80	3748	97	372	87
ind1	1842	474	831	388	825	386	5675	319	848	373
ind2	1535	55	830	61	923	64	4789	50	996	60
ind3	4415	785	2000	626	2020	627	11412	478	1936	587
ind4	6887	124	2236	104	2221	100	17175	81	2163	93
TOTAL	17155	1801	7328	1484	7384	1478	47282	1231	7356	1426
RATIO	2.33	1.26	0.99	1.04	1.00	1.04	6.43	0.86	1.00	1.00
TIME	3372		3694		3135		9586		3069	

[8] C. Fiduccia and R. Mattheyses. A linear time heuristic for improving network partitions. In *Proc. Design Automation Conf.*, pages 175–181, 1982.

[9] J. Hwang and A. El Gamal. Min-cut replication in partitioned networks. *IEEE Trans. on Computer-Aided Design*, pages 96–106, 1995.

[10] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning : Application in VLSI domain. In *Proc. Design Automation Conf.*, pages 526–529, 1997.

[11] E. L. Lawler, K. N. Levitt, and J. Turner. Module clustering to minimize delay in digital networks. *IEEE Trans. on Computer-Aided Design*, pages 47–57, 1969.

[12] C. E. Leiserson and J. B. Saxe. Retiming synchronous circuitry. *Algorithmica*, pages 5–35, 1991.

[13] L. Liu, M. Kuo, C. K. Cheng, and T. C. Hu. Performance-driven partitioning using a replication graph approach. In *Proc. Design Automation Conf.*, pages 206–210, 1995.

[14] L. T. Liu, M. T. Kuo, C. K. Cheng, and T. C. Hu. A replication cut for two-way partitioning. *IEEE Trans. on Computer-Aided Design*, pages 623–630, 1995.

[15] R. Murgai, R. K. Brayton, and A. Sangiovanni Vincentelli. On clustering for minimum delay/area. In *Proc. Int. Conf. on Computer-Aided Design*, pages 6–9, 1991.

[16] P. Pan, A. K. Karandikar, and C. L. Liu. Optimal clock period clustering for sequential circuits with retiming. *IEEE Trans. on Computer-Aided Design*, pages 489–498, 1998.

[17] M. Shih, E. S. Kuh, and R. S. Tsay. Performance-driven system partitioning on multi-chip modules. In *Proc. Design Automation Conf.*, pages 53–56, 1992.

[18] H. Yang and D. F. Wong. Circuit clustering for delay minimization under area and pin constraints. In *Proc. European Design and Test Conf.*, pages 65–70, 1995.