Performance Driven Multi-level and Multiway Partitioning with Retiming*

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

Chang Wu
Aplus Design Technologies, Inc.
Los Angeles, CA, USA
changwu@aplus-dt.com

Abstract

In this paper, we study the performance driven multiway circuit partitioning problem with consideration of the significant difference of local and global interconnect delay induced by the partitioning. We develop an efficient algorithm `HPM` (Hierarchical Performance driven Multi-level partitioning) that simultaneously considers cutsize and delay minimization with retiming. `HPM` builds a multi-level cluster hierarchy and performs various refinement while gradually decomposing the clusters for simultaneous cutsize and delay minimization. We provide comprehensive experimental justification for each step involved in `HPM` and in-depth analysis of cutsize and delay tradeoff existing in the performance driven partitioning problem. `HPM` obtains (i) 7% to 23% better delay compared to the state-of-the-art cutsize driven `hMetis` [11] at the expense of 19% increase in cutsize, and (ii) 81% better cutsize compared to the state-of-the-art delay driven `PRIME` [2] at the expense of 6% increase in delay.

1 Introduction

Circuit partitioning divides a given circuit into a collection of smaller subcircuits while satisfying the given area and/or pin constraints. Due to substantial advances in VLSI technology, designers are facing enormous increase in system complexity. As a result, the divide-and-conquer methodology is indispensable in order to make the VLSI layout problem tractable. The conventional objective of partitioning is to minimize the number of connections among the subcircuits. The problem must be partitioned into smaller subproblems with minimal amount of interconnections so that each subproblem can be solved effectively and efficiently. However, 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 another essential goal. Under the new interconnect-centric design paradigm, partitioning is seen as the crucial step that defines the local and global interconnects [1] as illustrated in Figure 1. To meet the performance requirement of today's complex design, partitioners must consider the amount of interconnect induced by partitioning (measured by its cutsize) as well as its impact on performance (measured by its delay). 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 driven partitioners do not consider delay, while many proposed delay driven 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.

*This research is partially supported by the MARCO/DARPA Gigascale Silicon Research Center (GSRC) and NSF Young Investigator Award MIP-9357582.

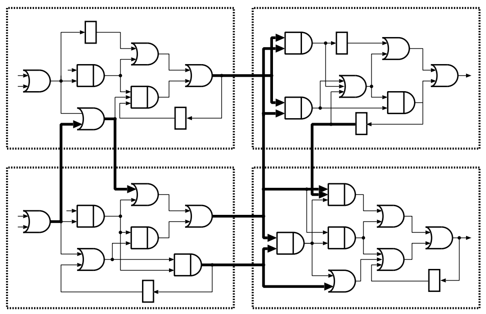


Figure 1: Role of partitioning under the interconnect-centric paradigm. Global interconnects are shown in thick lines.

Most of the performance driven circuit partitioning algorithms can be grouped into two categories; bottom-up clustering and top-down partitioning. The performance driven bottom-up clustering problem is to group gates into clusters under the upper bound of area and/or pin constraints so that the delay of the circuit is minimized. In [12], the authors proposed an efficient labeling based clustering algorithm to achieve the minimum delay for combinational circuits under simplistic delay model. [16, 18, 20] extend this work to consider more general delay model. Pan et al. [17] 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 [17] 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 the conventional cutsize driven partitioning, which in turn translates into more routing area and congestion problem, (ii) it is hard to control area balance among blocks and sometimes fail to obtain exact number of blocks.

The performance driven top-down 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. The primary objective of the problem is to minimize the delay of the circuit. Many of the proposed methods adopt two popular techniques for improving delay; retiming and logic replication. Shih et al. [19] proposed an algorithm to satisfy the timing constraints between registers. Hwang and Gamal [10] showed that logic replication from one block to another can improve cutsize and delay. Liu et al. [15, 14] proposed an efficient algorithm to combine logic replication and retiming for bipartitioning. Cong and Lim [6] provide cell move based formulation for simultaneous cutsize and delay minimization. However, these algorithms may suffer a long runtime for large circuits and do not guarantee any optimality.

In this paper, we study the performance driven multiway circuit partitioning problem with consideration of the significant difference of local and global interconnect delay induced by the partitioning. We develop an efficient algorithm `HPM` (Hierarchical Performance driven Multi-level partitioning) that simultaneously considers cutsize and delay minimization with retiming. `HPM` builds a multi-level cluster hierarchy and performs various refinement while gradually decomposing the clusters for simultaneous cutsize and delay minimization. During the clustering phase of `HPM`, a delay driven clustering [2] builds the initial cluster structure to ensure the best possible subsequent retiming, which is then extended by a cutsize driven multi-level clustering [5]. During the refinement phase of `HPM`, simultaneous cutsize and delay driven partitioning [6] is performed. We adopt the existing multiway partitioning framework [4] to overcome the limitation of recursive bipartitioning approach. Lastly, the delay result is further improved by retiming [13]. We provide comprehensive experimental justification for each step involved in `HPM` and in-depth analysis of cutsize and delay tradeoff existing in the performance driven partitioning problem. `HPM` obtains (i) 7% to 23% better delay compared to the state-of-the-art cutsize driven `hMetis` [11] at the expense of 19% increase in cutsize, and (ii) 81% better cutsize compared to the state-of-the-art delay driven `PRIME` [2] at the expense of 6% increase in delay.

The remainder of the paper is organized as follows. Section 2 provides formulation of the performance driven multiway partitioning problem with retiming. Section 3 presents `HPM` algorithm. Section 4 provides our experimental results. Section 5 concludes the paper with our ongoing research.

2 Preliminaries

2.1 Motivation

Most of existing algorithms on performance driven clustering and partitioning consider only combinational circuits. They do not consider retiming by assuming that the positions of flipflops are fixed [12, 16, 14, 20, 6]. Retiming is a very important sequential circuit optimization technique for delay minimization by repositioning flipflops. Traditionally, retiming is performed during logic synthesis and suffers from the problem of inaccurate routing delay estimation. On the other hand, if we perform retiming after partitioning, retiming can be performed under more accurate routing delay estimation. However, the capability of retiming is limited by the partitioning solutions as illustrated in Figure 2. The two partitioning solutions in Figures 2 have the same cutsize of 1 and delay of 4, where node delay is assumed to be 1 and inter-block delay is 2. Separate step of retiming can be performed for both solutions. For the solution (a), retiming cannot help to reduce the delay. However, retiming can reduce the delay of the solution (b) from 4 to 3. This indicates that if we consider retiming during partitioning, we tend to achieve better results with smaller delay.

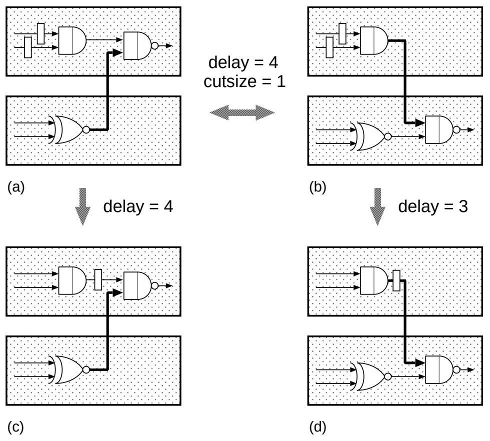


Figure 2: Advantage of simultaneous partitioning and retiming for delay minimization. Critical path is shown in thick lines.

2.2 Problem Formulation

A sequential gate-level circuit is represented as a hypergraph $H(V, E_H)$, where V is the set of nodes and E_H is the set of hyper-edges. Each node represents a gate in the circuit. Each hyper-edge represents a subset of nodes connected by one net in the circuit. Correspondingly, a circuit is also represented as a directed retiming graph $G(V, E_G, W)$ where V is the set of nodes representing gates in the circuit, E_G is the set of edges representing the connections between gates, and W is the set of edge weights. A directed edge $e(u,v)$ denotes the connection from gate u to gate v, and $w(e)$ denotes the number of FF's on the connection. A fan-in set of vertex v is defined as $FI(v) = \{u | u \in V \text{ and } e = (u,v) \in E_G\}$, and fan-out set of vertex v is similarly defined as $FO(v) = \{u | u \in V \text{ and } e = (v,u) \in E_G\}$. 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 $B = \{B_1, B_2, \cdots, B_K\}$ with retiming solution $R(V) = \{r(v) \mid \forall v \in V\}$ of given $G(V, E_G, W)$ 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, $1 \leq i \leq K$
- $w^r(e(u,v)) = w(e(u,v)) + r(v) - r(u) \geq 0$
- $r(v) = 0$ for all $v \in PI$ and $v \in PO$

We measure *cutsize* for given partitioning solution B of a sequential circuit, denoted $c(B)$, for area estimation of B. Cutsize is defined to be the total number of hyper-edges that connect vertices in different partitions. We measure *clock period* (or alternatively named *delay*) for given partitioning solution B of a sequential circuit, denoted $\phi(B)$, for performance evaluation of B. In "general delay model" [16, 17, 2], each node v has a delay of $d(v)$, and each edge $e(u,v)$ has a delay of $d(e) = D$ if u and v are in different partitions, and

$d(e) = 0$[1] otherwise. The delay of a path $p = (u \to v)$ from $u \in V$ to $v \in V$, denoted $d(p)$, is defined to be the sum of $d(e)$ and $d(v)$ for edges e and nodes v on p. A zero-weight path p is a path with $w(p) = 0$. The delay $\phi(B)$ induced by partitioning B is the longest path delay among all combinational paths, i.e., $\phi(B) = \max_p\{d(p)|w(p) = 0\}$. The *delay ratio* corresponds to $d(e)/d(v)$, which is equivalent to D in case we assume $d(v) = 1$ and e connects vertices in different partitions. This serves as a first-order approximation of how big global interconnect delay is compared to local interconnect delay. Our objective is to minimize both $\phi(B)$ and $c(B)$.

3 HPM Algorithm

We present HPM (Hierarchical Performance driven Multi-level partitioning) algorithm in this section. HPM builds a multi-level cluster hierarchy and performs various refinement while gradually decomposing the clusters for simultaneous cutsize and delay minimization.

3.1 Overview of HPM Algorithm

HPM consists of two phases, namely clustering and refinement. During the first step of clustering phase of HPM, a delay driven clustering algorithm PRIME [2] builds the initial cluster structure with consideration of subsequent retiming. Then, retiming is performed on top of clustering result. Next, a cutsize driven clustering algorithm ESC [5] adds more level to the cluster hierarchy in the second step. During the refinement phase of HPM, a simultaneous cutsize and delay driven partitioning algorithm xLR [6] is performed while decomposing clusters at each level. We adopt the existing multiway partitioning framework PM [4] to overcome the limitation of recursive bipartitioning approach. PM computes a matching of blocks to be generated and simultaneously applies xLR bipartitioner on top of every block pair. Finally, the delay result is further improved by retiming [13] on top of partitioning result.

The description of HPM algorithm is shown in Figure 3. Let C^i denote the cluster hierarchy at level i, C^0 being the original circuit. C^1 is computed using PRIME clustering, followed by RETIMING on C^1. ESC computes C^i for $2 \leq i \leq h$ in bottom-up manner, where h denotes the height of cluster hierarchy. PM obtains matching M of blocks for pairwise multiway refinement at each level i. Then, xLR bipartitioning is applied on top of each block pair to refine partition B^i in top-down manner. Finally, RETIMING is applied on B^0 to obtain the final partitioning result B, and the B along with its cutsize and delay are returned. HPM requires two parameters, namely cluster size limit L and delay ratio D. Section 4.2 and Section 4.4 demonstrate the impact of these parameters on cutsize and delay minimization.

3.2 Delay Driven Clustering with Retiming

In the first step of the clustering phase of HPM, we use the state-of-the-art delay driven clustering algorithm PRIME [2]. This is the first and crucial step of the entire HPM algorithm for delay optimization. We consider retiming during the cluster formation to obtain better quality result. To get a minimal delay for clustering and retiming, we binary search in a range of $[lb, ub]$, where lb is a lower-bound on the delay computed by assuming every edge has delay of 0, ub is an upper-bound computed by assuming every edge has delay of D (the global interconnect delay). For each target delay, we use label computation (to be explained later) to check its feasibility, i.e., if there exists a clustering and retiming solution with delay of no more than the target value. After getting the minimum delay, we can obtain a clustering solution and perform optimal retiming on the clustered circuit to get the minimum delay.

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

```
HPM(NL, L, D)
Input: netlist $NL$, area bound $L$, and delay ratio $D$
Output: partitioning $B$, cutsize $c(B)$, and delay $\phi(B)$
1.  $C^0 = NL$;
2.  $C^1 =$ PRIME($L, D$);
3.  $C^1 =$ RETIMING($C^1$);
4.  for ($i = 2$ to $h$)
5.      $C^i =$ ESC($C^{i-1}$);
6.  for ($i = h$ downto 1)
7.      $M =$ PM($B^i$);
8.      $B^{i-1} =$ xLR($C^i, B^i, M$);
9.  $B =$ RETIMING($B^0$);
10. return $B$, $c(B)$, and $\phi(B)$;
```

Figure 3: Description of HPM algorithm.

Let $G(V, E_G, W)$ be the retiming graph of the original circuit. Each gate v has delay $d(v)$, and each edge $e(u, v)$ has delay $d(e) = D$ if u and v are in different clusters, and $d(e) = 0$ otherwise. The *edge length*, denoted *length(e)*, of edge $e(u, v)$ is defined to be $-\phi \cdot w(e) + d(v)$ for a target delay ϕ. The *path length*, denoted *length(p)*, of a path p is $\sum_{e \in p} length(e)$. Intuitively, the length of a path represents the node delay on the path less the delay which can be reduced with retiming. Let $d_C(e)$ denote the delay of edge e in a clustered circuit C. Accordingly, the *length* of an edge e in C, denoted $length_C(e)$, is $length(e) + d_C(e)$. The path length of a path p in C, denoted $length_C(p)$, is $\sum_{e \in p} length_C(e)$. The l-value $l_C(v)$ of a node v in C is the maximum path length from primary inputs to v in C. Based on the retiming theory in [13], the authors of [17] showed that:

Theorem 1 *In a clustered circuit C of a sequential circuit with a target clock period ϕ, if there is a primary output whose l-value is greater than ϕ, C cannot be retimed to ϕ or less. On the other hand, if the l-values of all primary outputs are less than or equal to ϕ, C can be retimed to a delay less than $\phi + D$.*

Accordingly, we can check if the delay of a clustered circuit C is larger than ϕ or not by computing node l-values. For a target delay ϕ, let *node label* of node v, denoted $l^{opt}(v)$, be the minimum $l_C(v)$ among all clustered circuits C. Based on Theorem 1, to check if there exists a feasible clustering solution for a given ϕ, we can compute node labels and check if $l^{opt}(v) \leq \phi$ for every primary output v. With binary search, we can compute a clustering with retiming solution with a delay of less than $\phi_{min} + D$, where ϕ_{min} is the minimum delay. Notice, however, that in order to achieve a delay of less than $\phi_{min} + D$ node duplication is needed which can increase the circuit size significantly. In our work, we form a duplication-free clustering solution at the cost of larger delay.

3.3 Cutsize Driven Clustering

In the second step of the clustering phase of HPM, we use a cutsize driven multi-level clustering algorithm ESC (Edge

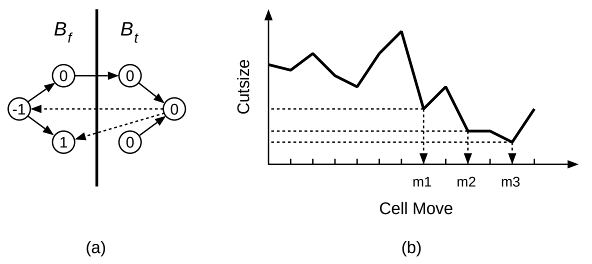


Figure 4: (a) Illustration of cost function $r(x)$ used in xLR. (b) Illustration of MRP (Multiple Rollback Point) scheme.

Separability based Clustering) [5]. ESC is an efficient graph search based bottom-up clustering algorithm. Unlike existing algorithms that are based on local connectivity information of the netlist such as edge weights, ESC exploits more global connectivity information *edge separability* to guide clustering process. For given edge $e = (x, y)$ in an edge weighted undirected graph $U(V, E_U, W_U)$, edge separability of e is defined as the minimum cutsize among the cuts separating x and y in U. Thus, computing the edge separability for a given edge $e = (x, y)$ is equivalent to finding the x-y mincut. Direct computation of edge separability for all edges in U requires max-flow computation for $|E_U|$ times, which is extremely time-consuming. ESC provides an efficient way to estimate separability of *all* edges in U in $O(|V| \log |V|)$ time without using any flow computation. ESC is a bottom-up clustering algorithm, where clusters grow from the contraction of edges. Each vertex belongs to its own cluster initially, and clusters grow from greedy merging of edges based on the estimation of edge separability computed by ESC. ESC clustering algorithm can be applied repeatedly to build multi-level cluster hierarchy. Then partitioning can apply on each level of the hierarchy while propagating partitioning information, starting from the top to bottom.

3.4 Cutsize and Delay Driven Refinement

During the refinement phase of HPM, we use xLR partitioning [6] for simultaneous cutsize and delay minimization. xLR is an extension of cutsize driven partitioning algorithm LR [3] to consider both cutsize and delay during cell moves. xLR tries to avoid having cyclic dependency among partitions so that critical paths do not cross the partition boundary multiple times. Since xLR adopts cost function that reflects cutsize and delay into the cell move gain formulation, xLR serves as an excellent choice for refinement after the decomposition of clusters. Assuming topological ordering of V in $G(V, E_G, W)$ is from partition B_f to B_t, xLR uses 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, xLR tries to minimize $|V|$ instead of requiring V to be $\emptyset$. The cell move gain based on this concept can be represented in terms of $r(x)$ as illustrated in Figure 4-(a). We refer interested readers to [6] for more details on the algorithm and experimental results that demonstrate its effectiveness on cutsize and delay minimization.

In addition, we examine the property of cell move based algorithms and propose a scheme named MRP (Multiple Rollback Point). The rational behind MRP scheme is to choose a partition with good delay that does not compromise cutsize at the end of single pass of cell moves. MRP scheme can prevent the time-consuming evaluation of delay upon *each* cell move in order to find a solution with good cutsize and delay partition. In conventional FM algorithm, only single solution that minimizes the cutsize during each pass is kept. As depicted in Figure 4-(b), FM returns m_3 as the best partition observed so far. In MRP scheme. however, we keep multiple partitions that maintain good cutsize from which we select the one that gives the best delay result. During each pass, we maintain a priority queue to maintain k best-cutsize partitions with *unique* cutsize value (m_1, m_2, and m_3 in Figure 4-(b)). At the end of pass, we compute delay of each kept partition to select the best-delay one. The reason we decide to keep unique-cutsize solutions is to sample various local minimum points for good cutsize/delay result.

4 Experimental Result

4.1 Experimental Setting

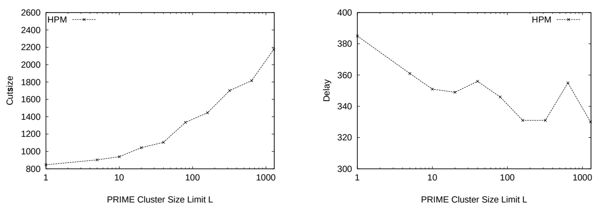
We implemented our algorithms in C++/STL, compiled with gcc v2.4, and tested on SUN ULTRA SPARC60 at 360Mhz. We obtained the latest binary executable of hMetis [11] (v1.5.3) for the evaluation. The benchmark set consists of 7 ISCAS circuits and 4 large scale industrial designs provided by our industrial sponsor. Detailed statistics of the circuits can be found in our technical report [7]. We report cutsize, delay, and runtime from 16-way partitioning results obtained by recursively applying bipartitioning for all algorithms except for HPM, where HPM generates the blocks simultaneously. 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 reports the number of hyper-edges that span more than single block.

We assume that all gates have unit area and unit delay, while primary inputs, primary outputs and flip-flops have no area[2] and no delay. We apply retiming [13] 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 according to Table 2 and $L = 10$ according to Section 4.2 throughout the entire experiments unless specified otherwise. We use (i) FM [9] and hMetis [11] for the de facto standard and state-of-the-art conventional cutsize driven partitioning algorithms, and (ii) PRIME [2], FLARE [6], and HPM for the state-of-the-art delay driven partitioning algorithms for an extensive analysis on cutsize/delay tradeoff.

4.2 Choice of Cluster Size

Figure 5 shows the impact of PRIME cluster size limit L on cutsize and delay. Figure 5-(a) reveals the relation L *vs* cutsize, and Figure 5-(b) reveals the relation L *vs* delay. We observe that cutsize degrades but delay improves as L increases. PRIME tends to form more unbalanced clusters in terms of size distribution as L increases, giving subsequent partitioner higher chance of getting stuck in a local cutsize minima. As L increases, however, delay tends to improve due to higher flexibility in forming good quality clusters. We observe that $L = 10$ establishes a good tradeoff between cutsize and delay.

[2]We do not consider the area of flip-flops because their positions are not yet determined until the final step of retiming. It is possible that we violate partitioning area balance constraint if the area of flip-flops is required to be non-zero. In such a case, we may need to perform a post-process to balance partition area.



(a) Size Limit *vs* Cutsize (b) Size Limit *vs* Delay

Figure 5: Impact of PRIME cluster size limit L on cutsize and delay result of HPM. Benchmark circuit ind1 is used.

4.3 Overall Comparison

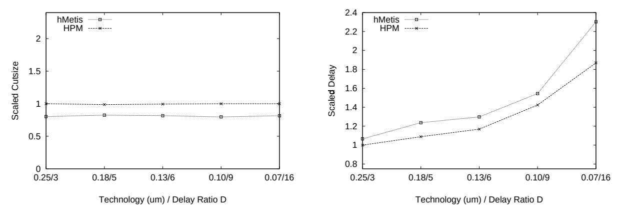
Table 1 reveals the overall cutsize and delay comparison among all partitioning algorithms mentioned in this paper. First of all, Table 1 shows the impact of retiming on delay. The columns "bfr" and "aft" respectively denote delay results before and after retiming. We observe that delay results always improve upon retiming regardless of partitioning objectives. In addition, the margin of improvement is larger for FM compared to other algorithms. Note that cutsize is not affected by the retiming process since moving a flip-flop f across partition boundary will not change the cut-state of a net n that is incident to f. These retiming results are the best possible ones since PRIME, the first step of the clustering phase of HPM, ensures the optimality as discussed in Section 3.2.

We summarize our observations in what follows. First, HPM obtains (i) 14% better delay compared to the state-of-the-art cutsize driven hMetis [11] at the expense of 19% increase in cutsize, and (ii) 81% better cutsize compared to the state-of-the-art delay driven PRIME [2] at the expense of 6% increase in delay. In addition, the delay advantage of HPM over hMetis is expected to increase for the future technology as shown in Section 4.4. Second, the conventional cutsize driven partitioning hMetis improves both the cutsize and delay of FM. This illustrates the side-effect of cutsize minimization objective on delay minimization. However, experimental results indicate that the delay of hMetis can still be further improved by HPM while maintaining comparable cutsize quality. Third, HPM significantly improves the cutsize of PRIME with a surprising margin of 81.4%. In addition, HPM obtains delay result that is only 6.2% worse than quasi-optimal PRIME. Runtime is also improved by 67.9%.

4.4 Impact of Delay Ratio

Table 2 shows the Elmore delay computation of local *vs* global interconnect for 5 technology generations based on NTRS '97 parameters [8]. It is used to determine the delay ratio D for each technology generation. The wire length l and wire width w are chosen according to [8], and the driver resistance R_d and loading capacitance C_L are $10\times$ and $100\times$ the minimum device for local and global interconnect, respectively.

Figure 6 shows the impact of delay ratio increase on cutsize and delay. As shown in Table 2, D is expected to increase as the technology advances into deeper sub-micron. The state-of-the-art conventional cutsize driven partitioning algorithm hMetis and performance driven delay driven HPM are used to illustrate the prediction. Figure 6-(a) illustrates D *vs* cutsize, and Figure 6-(b) illustrates D *vs* delay. First, we observe that cutsize is not affected by the delay ratio change. hMetis obtains 16% better cutsize for the 5



(a) Delay Ratio *vs* Cutsize (b) Delay Ratio *vs* Delay

Figure 6: Impact of delay ratio increase on cutsize and delay for hMetis *vs* HPM. The computation of corresponding D is shown in Table 2.

current and future technologies compared to HPM, but this trend stays unaffected by the delay ratio change. However, it is clear from Figure 6-(b) that delay of both hMetis and HPM is affected by change in D. The rate of delay increase of hMetis is faster than that of HPM, and 13.5% delay advantage of HPM over hMetis for the current $0.18\mu m$ technology is expected to increase to 23.2% for $0.07\mu m$ technology.

5 Conclusion and Ongoing Work

We studied the performance driven circuit partitioning problem and presented an efficient algorithm HPM that simultaneously considers cutsize and delay minimization. HPM considers local and global interconnect delay and provides wide variety of cutsize/delay tradeoff points. HPM builds a multi-level cluster hierarchy and performs various refinement while gradually decomposing the clusters for simultaneous cutsize and delay minimization. Our experiments demonstrate the effectiveness as well as justification for each step involved in HPM. Finally, HPM obtains very competitive cutsize and delay results compared to the state-of-the-art hMetis and PRIME. Our ongoing studies include (i) the impact of cell duplication during PRIME on cutsize and delay, and (ii) geometric embedding based performance driven partitioning.

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

[5] J. Cong and S. K. Lim. Edge separability based circuit clustering with application to circuit partitioning. In *Proc. IEEE/ACM Asia South Pacific Design Automation Conf.*, pages 429–434, 2000.

[6] J. Cong and S. K. Lim. Performance driven multiway partitioning. In *Proc. IEEE/ACM Asia South Pacific Design Automation Conf.*, 2000.

	Cutsize Minimization						Delay Minimization							
	FM			hMetis			PRIME		FLARE			HPM		
ckt	cut	bfr	aft	cut	bfr	aft	cut	dly	cut	bfr	aft	cut	bfr	aft
s9234	219	176	35	145	72	30	359	35	139	73	35	186	73	30
s5378	328	123	35	193	50	29	327	27	185	51	29	214	43	26
s13207	409	206	55	203	96	55	414	45	172	94	50	232	91	60
s15850	512	307	72	226	121	57	762	53	218	122	63	302	117	60
bigkey	292	89	20	38	25	12	761	12	44	24	10	43	25	10
s38584	849	178	47	294	71	38	1860	34	283	68	39	354	75	39
clma	1659	514	101	332	176	84	3748	97	372	170	87	414	180	78
ind1	2330	1565	481	831	671	388	5675	319	848	640	373	934	627	341
ind2	1920	269	65	830	102	61	4789	50	996	112	60	1075	92	43
ind3	5032	2660	787	2000	1084	626	11412	478	1936	991	587	2352	999	533
ind4	7478	198	127	2236	174	104	17175	81	2163	183	93	2673	157	87
TOTAL	21028	6285	1825	7328	2642	1484	47282	1231	7356	2528	1426	8779	2479	1307
RATIO	2.39	2.54	1.40	0.83	1.07	1.13	5.39	0.94	0.84	1.02	1.09	1.00	1.00	1.00
TIME	3372			3694			9586		3182			3073		

Table 1: Cutsize and delay comparison among various partitioning algorithms. "bfr" and "aft" respectively denote delay results before and after retiming. RATIO is relative to HPM results. TIME denotes the sum of total CPU time elapsed for all benchmark circuits.

	Local Interconnect					Global Interconnect				
tech (μm)	0.25	0.18	0.13	0.10	0.07	0.25	0.18	0.13	0.10	0.07
l (mm)	1.25	0.9	0.65	0.5	0.35	17.3	18.4	20.7	22.8	24.9
w (μm)	0.25	0.18	0.13	0.10	0.07	1.20	0.89	1.80	1.83	1.97
R_d (Ω)	1620	1710	2210	2340	2210	162	171	221	234	221
C_L (fF)	2.82	2.34	1.35	0.72	0.66	28.2	23.4	13.5	7.2	6.6
R_w (Ω)	365	340	405	460	475	187	182	101	110	95
C_w (fF)	121	67.3	31.8	25.2	15.4	2924	2348	1779	1962	2423
delay (ps)	224	131	80	67	40	757	623	487	569	653
D	1	1	1	1	1	3.4	4.8	6.1	8.5	16.3

Table 2: Elmore delay computation of local *vs* global interconnect for 5 technology generations based on NTRS '97 parameters. l, w, R_d, C_L, R_w, and C_w respectively denote wire length, wire width, driver resistance, loading capacitance, total wire resistance, and total wire capacitance.

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

[8] J. Cong and D. Z. Pan. Interconnect estimation and planning for deep submicron designs. In *Proc. ACM Design Automation Conf.*, pages 507–510, 1999.

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

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

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

[12] 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.

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

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

[15] L. Liu, M. Shih, N. C. Chou, C. K. Cheng, and W. Ku. Performance-driven partitioning using retiming and replication. In *Proc. Int. Conf. on Computer-Aided Design*, pages 296–299, 1993.

[16] 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.

[17] 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.

[18] R. Rajaraman and D. F. Wong. Circuit clustering for delay minimization. In *Proc. ACM Design Automation Conf.*, pages 309–314, 1993.

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

[20] 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.