Simultaneous Delay and Power Optimization for Multi-level Partitioning and Floorplanning with Retiming

Mongkol Ekpanyapong and Sung Kyu Lim

School of Electrical and Computer Engineering,
Georgia Institute of Technology
{pop,limsk}@ece.gatech.edu

Abstract

Delay minimization and power minimization are two important objectives in the design of the high-performance, portable, and wireless computing and communication systems. Retiming is a very effective way for delay optimization for sequential circuits. In this paper we propose a unified framework for multi-level partitioning and floorplanning with retiming, targeting simultaneous delay and power optimization. We first discuss the importance of retiming delay and visible power as opposed to the conventional static delay and total power for sequential circuits. Then we propose GEO-PD algorithm for simultaneous delay and power optimization and provide smooth cutsize, wirelength, power and delay tradeoff. In GEO-PD, we use retiming based timing analysis and visible power analysis to identify timing and power critical nets and assign proper weights to them to guide the multi-level optimization process. In general, timing and power analysis are done at the original netlist while a recursive multi-level approach performs partitioning and floorplanning on the sub-netlist as well as its coarsened representations. We show an effective way to translate the timing and power analysis results from the original netlist to a coarsened sub-netlist for effective multi-level delay and power optimization. To the best of our knowledge, this is the first paper addressing simultaneous delay and power optimization in multi-level partitioning and floorplanning.

1. Introduction

Delay minimization and power minimization are two important objectives in the design of the high-performance, portable, and wireless computing and communication systems. Thus, a considerable research effort has been made in trying to find power and delay-efficient solutions to circuit design problems. One such procedure that is applied at the logic level is circuit partitioning and floorplanning.

Circuit partitioning aims to divide a given circuit to smaller sub-circuits so that it can be used in the next physical design process for hierarchical design approach. Traditionally, the objective of partitioning is to minimize the amount of interconnection among sub-circuits [10,2,5], which has direct impact on the final chip area. Delay has also been an important objective in partitioning [4,7,8,14], which aims to minimize the number of inter-partition connection on critical paths. A recent research [1,3] focused on simultaneous cutsize and delay optimization. Another recent study [9] addresses power optimization in clustering. After partitioning the given circuits into sub-circuits, floorplanning is applied to identify the dimension and location of the sub-circuits. Among several ways to perform floorplanning, partitioning based method has been one of the viable approaches. Most partitioning-based floorplanning algorithms attempt to minimize area and wirelength. A recent study [3] attempts to minimize wirelength and delay in multi-level partitioning based floorplanning. However, none of these existing works addresses simultaneous power and delay optimization for partitioning and floorplanning. Retiming [6] is logic optimization technique by shifting flip-flops (FFs) position for delay minimization. Recent studies [3,4,8,14] show how to perform partitioning and retiming simultaneously.

In this paper we propose a unified framework for multi-level partitioning and floorplanning with retiming, simultaneously optimizing delay and power. We first discuss the importance of retiming delay and visible power as opposed to the conventional static delay and total power for sequential circuits. Then we propose GEO-PD algorithm for simultaneous delay and power optimization and provide smooth cutsize, wirelength, power and delay tradeoff. In GEO-PD, we use retiming based timing analysis and visible power analysis to identify timing and power critical nets and assign proper weights to them to guide the multi-level optimization process. In general, timing and power analysis are done at the original netlist while a recursive multi-level approach performs partitioning and floorplanning on the sub-netlist as well as its coarsened representations. We show an effective way to translate the timing and power analysis results from the original netlist to a coarsened sub-netlist for effective multi-level delay and power optimization. Our experiments based on large scale ISCAS89 [12] and ITC99 [13] benchmark circuits reveal smooth tradeoff among cutsize, wirelength, delay, and power. To the best of our knowledge, this is the first paper addressing both delay and power optimization in multi-level partitioning and floorplanning.

The organization of this paper is as follows. Section 2 describes problem formulation. Section 3 is devoted to our algorithm GEO-PD. Section 4 presents our experimental result and analysis. Finally, the last section presents our conclusions.

2. Problem Formulation

Given a sequential gate-level netlist $NL(C, N)$, where C is the set of cells representing gates and flip-flops, and N is the set of nets connecting the cells, the purpose of the Power and Delay driven K-way Partitioning with Retiming (PDPR) problem is to assign cells in NL to K blocks while area constraint for each block is satisfied. The Power and Delay driven Floorplanning with Retiming (PDFR) problem is to find the location of blocks obtained by PDPR. Given a PDPR/PDFR solution B, let $\theta(B)$, $\omega(B)$, $\delta(B)$, $\phi(B)$, $\pi(B)$, and $\rho(B)$ respectively denote the cutsize, wirelength, static delay, retiming delay, visible power, and total power (all of them to be defined later).[1] The formal definitions of PDPR and PDFR problems are as follows:

PDPR Problem The Power and Delay driven K-way Partitioning with Retiming problem under the given area constraints $A = (L_i, U_i)$ has a solution $B = \{B_1, B_2, ..., B_K\}$, where B denotes the set of blocks. B is feasible if it satisfies the following conditions: i) $B_i \subset C$, $1 \le i \le K$, ii) $L_i \le |B_i| \le U_i$, $1 \le i \le K$, iii) $B_1 \cup B_2 \cup ... \cup B_K = C$, iv) $B_i \cap B_j = \varnothing$ for all $i \ne j$. The objective is to minimize $\theta(B) + \alpha \cdot \phi(B) + \beta \cdot \pi(B)$.

PDFR Problem The Power and Delay driven K-way Floorplanning with Retiming problem has a solution $B = \{B_1(x_1,y_1), B_2(x_2,y_2), ..., B_K(x_K,y_K)\}$, where B denotes the set of blocks, and (x_i,y_i) represents their geometric locations. We obtain the block information from PDPR. The objective is to minimize $\omega(B) + \alpha \cdot \phi(B) + \beta \cdot \pi(B)$.

2.1. Cutsize and Wirelength Objective

We model NL using a hypergraph $H=(V, E_H)$, where the vertex set V represents cells, and the hyperedge set E_H represents nets in NL. Each hyperedge is non-empty subset of V. The *cutsize* $\theta(B)$ of partitioning solution B is total number of hyperedges connecting vertices in different blocks.

The x-span of hyperedge h, denoted h_x, is defined as $h_x = \max_{c \in h}\{x_i \mid c \in B_i\} - \min_{c \in h}\{x_i \mid c \in B_i\}$. The y-span, denoted h_y, is calculated using the y-coordinates. The sum of x-span and y-span of each hyperedge h is the half-parameter of the bounding block (HPBB) of h and denoted $HPBB(h)$. The *wirelength* $\omega(B)$ of floorplanning solution B is the sum of HPBB of all hyperedges in H.

2.2. Delay Objective

For delay objective, we model NL using a directed graph $G = (V, E)$ where the vertex set V represents cells, and the directed edge set E represents the signal direction in NL. In the *geometric delay model*, each vertex v has delay $d(v)$ and each edge $e=(u,v)$ has delay $d(e)$. Let $s(e)$ denote the *cut-state* of e: $s(e)=1$ if e is cut, and $s(e)=0$ otherwise. In case of

[1] Our objective functions during PDPR are cutsize, retiming delay, and visible power, and our objective functions during PDFR are wirelength, retiming delay, and visible power. However, we measure and report the static delay and total power as well in this paper. Our algorithm can easily be modified to optimize these objectives as well. In addition, our experimental results in Section 4 demonstrate how much retiming can help to reduce delay for huge sequential circuits.

```
GEO-PD(NL,K)
insert all cells in NL to root node R in T (= partitioning tree)
insert R into Q (= FIFO queue)
while (leaf nodes in T < K)
     N = remove front element in Q
     GEO-PD-2way(N) (= bipartitioning on N)
     split cells in N into N1 and N2
     insert N1 and N2 into Q and T
     if (there are 2^j leaf nodes in T for j>1)
          GEO-PD-Kway(T)
return T

GEO-PD-2way(N)
NL' = sub-netlist containing cells in N
ESC(NL') (= multi-level clustering on NL')
h = height of the cluster hierarchy
B = random partitioning among clusters at level h
for (i = h downto 0)
     NL'(i) = coarsened NL' at level i
     while (gain)
          DELAY-WEIGHT(NL'(i))
          POWER-WEIGHT(NL'(i))
          total net weight = 1 + power weight + delay weight
          while (gain)
               move cells in NL'(i) to minimize weighted cutsize
               retrieve max gain moves and update B
     project B to level i-1
return B

GEO-PD-Kway(T)
B = derive initial partitioning for NL from leaf nodes in T
ESC'(NL) (= restricted clustering preserving K-way cutlines)
perform multi-level partitioning to minimize weighted wirelength
update T
```

Figure 1. Overview of the GEO-PD algorithm

partitioning, we assume $d(e) = k_d s(e)$, where k_d is given by the user (we use k_d=2 for our experiment). The rationale is that the delay of a long wire (= cut edge) is much greater than that of short wire (= uncut edge). In case of floorplanning, $d(e) = m(e) \cdot s(e)$, where $m(e) = |x_u - x_v| + |y_u - y_v|$. The delay of a path *p*, denoted *d*(*p*), is the sum of the delay of gates and edges along *p*. Then, the *static delay* $\delta(B)$ of partitioning and/or floorplanning solution *B* is $\max_{p \in G}\{d(p(u,v)) \mid u \in PI \text{ or } FF \ \& \ v \in PO \text{ or } FF\}$.

By employing the concept of retiming graph [6], we model *NL* using a directed graph $R = (V, E_R)$, where the edge weight $w(e)$ of $e=(u,v)$ denotes the number of flip-flops between gate *u* and *v*. The path weight can be calculated by $w(p)=\sum_{e \in p} w(e)$. Let $w^r(e)$ denote edge weight after retiming *r*, i.e. number of flip-flops on the edge after retiming. Then, $w^r(p)=\sum_{e \in p} w^r(e)$. A circuit is retimed to a delay ϕ by a retiming *r* if the following conditions are satisfies; (i) $w^r(e) \geq 0$ for each *e*, (ii) $w^r(p) \geq 1$ for each path *p* such that $d(p) > \phi$. We define the edge length of $e=(u,v)$ as $l(e) = -\phi \cdot w(e) + d(v) + d(e)$, and the path length of *p* as $l(p)= \sum_{e \in p} l(e)$. The *sequential arrival time* [8] of vertex *v*, denote $l(v)$, is maximum path length from PIs or FFs to *v*. If the sequential arrival time of all POs or FFs are less than or equal to ϕ, the target delay ϕ is called *feasible*. Let $D_g = \max\{d(v) \mid v \in V\}$. Then, the *retiming delay* $\phi(B)$ of a partitioning and/or floorplanning solution *B* is the minimum feasible $\phi + D_g$.

2.3. Power Objective

For power objective, we model *NL* as hypergraph $H=(V, E_H)$ as discussed in Section 2.1. The *total power consumption* $\rho(B)$ of partitioning/floorplanning solution *B* is calculated as follows:

$$P_t = \frac{1}{2} V_{dd}^2 \cdot f \cdot \sum_{v \in V} (C_g(v) + C_w(v)) \cdot SA(v)$$

where V_{dd} is supply voltage, f is global clock frequency, $C_g(v)$ and $C_w(v)$ represent the gate capacitance and wire capacitance seen by gate v, and $SA(v)$ is switching activity of v. $C_g(v)$ is the sum of the input capacitance of all sink gates driven by v. Let n_v denote the net whose driving gate is v. In case of partitioning, $C_w(v) = k_p \cdot s(n_v) \cdot C_g(v)$, where k_p is given by the user (we use k_p=2 for our experiment). The rationale is that the power consumption by the gate driving a long wire (=cut net) is much larger than that of short wire (= uncut net). In case of floorplanning, $C_w(v) = HPBB(n_v) \cdot C_g(v)$. Let VG be the set of *visible gates* that is defined as $VG = \{v \mid s(n_v) = 1\}$. Then, the *visible power consumption* $\pi(B)$ of partitioning and/or floorplanning solution B is calculated as follows:

$$P_v = \frac{1}{2} V_{dd}^2 \cdot f \cdot \sum_{v \in VG} (C_g(v) + C_w(v)) \cdot SA(v)$$

We note that the wire capacitance $C_w(v)$ is the only factor that changes based on a partitioning or floorplanning solution. In other words, the power consumed by non-visible gates is fixed regardless of partitioning or floorplanning results. Thus, we attempt to minimize the visible power in our algorithms.

3. GEO-PD Algorithm

3.1. Overview of GEO-PD Algorithm

An overview of the GEO-PD algorithm is shown in Figure 1. GEO-PD is a multi-level partitioner and floorplanner for simultaneous delay and power optimization. GEO-PD partitions and floorplans the given netlist *NL* into $K=n \times m$ dimension using a top-down recursive bipartitioning approach. If the location information of the blocks is ignored, GEO-PD gives a partitioning solution for PDPR problem; otherwise GEO-PD gives a floorplan solution for PDFR problem. `GEO-PD` consists of two subroutines: `GEO-PD-2way` recursively bipartitions *NL*, whereas `GEO-PD-Kway` refines these partitioning results occasionally as illustrated in Figure 2. `GEO-PD-2way` is performed on the sub-netlist, whereas `GEO-PD-Kway` is performed on the entire netlist. Initially, the partitioning tree *T* has only root node *R*, and all cells in *NL* are inserted into *R*. The FIFO (First In First Out) queue *Q* is used to support the recursive breadth-first cut sequence.

`GEO-PD-2way` first generates the sub-netlist from the given partition tree node and performs multi-level clustering on it. We use ESC clustering algorithm [2] for this purpose. An illustration of multi-level cluster hierarchy is shown in Figure 2. Then we obtain a random initial partitioning *B* among the clusters at the top level of the hierarchy. The subsequent top-down multi-level refinement is used to improve *B* in terms of delay and power. We perform retiming based timing analysis RTA [3] to identify timing critical nets. We also perform power analysis to identify power critical nets. Then we compute the delay and power weights for the nets in the sub-netlist for simultaneous delay and power optimization. The subsequent iterative improvement through cluster move tries to minimize the weighted cutsize. Finally we project the current solution to the next level coarser netlist for multi-level optimization. At the end of `GEO-PD-2way`, two new children nodes are inserted into *T* based on *B*.

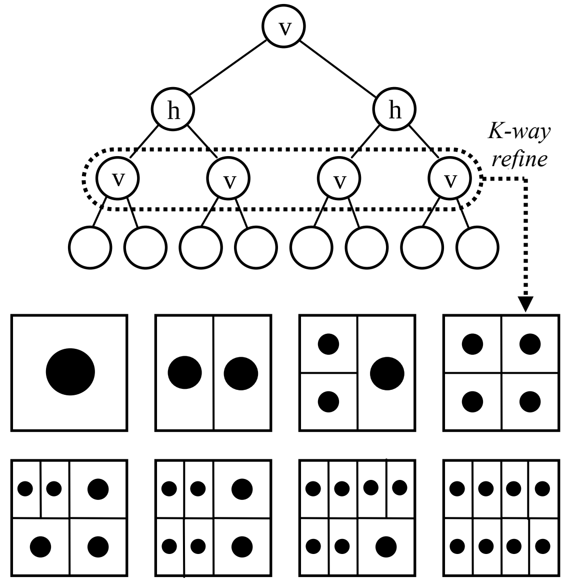

Figure 2. Illustration of partitioning tree and breadth-first cut sequence in GEO-PD algorithm. v and h denote vertical and horizontal cuts. A *K*-way refinement is performed when there are 2^j blocks ($j > 1$).

`GEO-PD-Kway` refinement is performed when we obtain 2^j partitions ($j > 1$) from `GEO-PD-2way` (4, 8, 16 partitions, etc). We first perform a restricted multi-level clustering, where grouping among cells in different partition is prohibited. This allows the partitioner to preserve the initial partitioning results. Then we again perform multi-level partitioning in the same way as in `GEO-PD-2way` for additional delay and power improvement. `GEO-PD-Kway` is applied onto the global netlist for more global level optimization.

3.2. Weight Computation

For simultaneous delay and power optimization, we first identify timing and power critical nets and assign proper weights to them to guide the optimization process. A net is *timing critical* if it lies along a critical path and *power critical* if it has high fanout with large wirelength and is driven by a gate with high switching activity. In GEO-PD, retiming delay and visible power are minimized through retiming based timing analysis [3] and visible power analysis. We use *sequential slack* [3] to compute how much time slack exists before timing violation occurs after retiming. These values are then used to compute the delay weights of the nets for retiming delay minimization. In case of power optimization, we use switching activity and gate/wire capacitance to compute power weights of the nets for visible power minimization. Both delay and power weights are added together, and GEO-PD performs multi-level partitioning to minimize the total weighted cutsize (for partitioning) or weighted wirelength (for floorplanning).

We note that the multi-level approach [2,5] is very effective in minimizing the weighted cutsize and wirelength. However, timing and power analysis is typically done at the original netlist while a recursive multi-level approach performs partitioning and floorplanning on the sub-netlist as well as its coarsened representations. Thus, it is crucial that we have an effective way to translate the timing and power analysis results from the original netlist to a coarsened sub-netlist.

3.2.1. Delay Weight Computation. Figure 3 shows `DELAY-WEIGHT(NL')`, our delay weight calculator. Before we perform retiming based timing analysis (RTA), we initialize the edge delay in *R* (= retiming graph) based on the current partitioning/floorplanning results. In case of partitioning, we set the delay of cut edges to k_d and uncut edges to 0 as discussed in Section 2.2. In case of floorplanning, we set the delay of edges to their Manhattan distances. Then, a Bellman-Ford variant RTA is performed from a given feasible delay to compute sequential slack. For each cluster *C* from the given coarsened sub-netlist `NL'`, we compute *C*(*R*), the set of all the nodes in *R* that are grouped into *C*. We use the minimum slack among all cells in *C*(*R*) as the slack for *C*. The reason we use the minimum slack value is since the critical path

```
DELAY-WEIGHT(NL')
set delay of edges in R (= retiming G)
perform RTA(R) (= timing analysis)
compute sequential slack for nodes in R
for each cluster C in NL'
   C(R) = all cells in R grouped into C
   slack(C) = min among cells in C(R)
X = top x% clusters with small slack
for each net N in NL'
   if (all clusters in N are in X)
      compute delay-weight(N) using Eqn1

POWER-WEIGHT(NL')
for each net Nv in NL'
   Nv' = corresponding net in NL
   compute HPBB(Nv')
   compute power-weight(Nv) using Eqn2
```

Figure 3. Overview of the delay and power weighting functions in GEO-PD algorithm

information is preserved regardless of multi-level clustering results (we have also performed experiments using average slack value instead of minimum. But the minimum slack method generated better delay results).

After the cluster slack computation is finished, we sort the clusters in a non-decreasing order of their slack values. We store the top x% (we use 3% in our experiment) into a set X. For each net that contains *only* the clusters in X, we use the following equation to compute the delay weight:

$$dwgt(n) = \alpha\left(1 - \frac{\min\{slack(v) \mid v \in n\}}{\max\{slack(w) \mid w \in NL'\}}\right)^{p1} \quad (1)$$

This equation gives higher weights to the nets that contain smaller minimum cluster slack, thus giving higher priority to the nets containing more timing critical clusters. Instead of requiring *all* clusters in a net to be timing critical, we tried another scheme where we give delay weights to the nets with 2 or more timing critical clusters. Our related experiment indicates that this approach produced worse results. Our extensive experiments indicate that α=25 and $p1$=1 are an excellent empirical choice.

3.2.2. Power Weight Computation. Figure 3 shows `POWER-WEIGHT(NL')`, our power weight calculator. As discussed earlier in Section 2.3, our goal is to minimize visible power consumption since the power consumed by non-visible gates is fixed regardless of partitioning or floorplanning results. Since we do not know a priori which nets will be cut after the partitioning, we compute the power weights assuming all nets are cut. Then our goal is to minimize the weighted cutsize or wirelength. For a net driven by a gate v, we use the following equation to assign power weight:

$$pwgt(n_v) = \beta\left(\frac{SA(v)[C_g(v) + C_w(v)]}{\max\{SA(u)[C_g(u) + C_w(u)] \mid u \in V\}}\right)^{p2} \quad (2)$$

where $SA(v)$, $C_g(v)$ and $C_w(v)$ respectively represent the switching activity, gate capacitance and wire capacitance seen by gate v. We use $C_w(v) = k_p \cdot C_g(v)$ for partitioning and $C_w(v) = HPBB(n_v) \cdot C_g(v)$ for floorplanning. This equation gives higher weights to the nets that have high fanout, larger wirelength, and source gate with high switching activity. In a multi-level approach, each net in the original netlist NL is transformed depending on the given sub-netlist NL' and its multi-level clustering information. For example, $n_a=\{a,b,c,d\}$ in NL becomes $n_{C1}=\{C1,C2\}$ if NL' contains a and b only and a is clustered into $C1$ and b into $C2$. In this case, we compute $HPBB(n_a)$ based on the location of $C1$, $C2$, c, and d, and use $SA(a)$ in our power weight equation. Our extensive experiments indicate that β=25 and $p2$=0.3 are an excellent empirical choice.

4. Experimental Results

Our algorithms are implemented in C++/STL, compiled with gcc v2.96, and run on Pentium III 746 MHz machine. The benchmark set consists of six big circuits from ISCAS89 [12] and four big circuits from ITC99 [13] suites. We

Table 1. Benchmark circuit characteristics.

ckt	gate	PI	PO	FF	Dr	Ds
b17o	22854	37	97	1414	38	44
b20o	11979	32	22	490	73	74
b21o	12156	32	22	490	73	74
b22o	17351	32	22	703	78	79
s5378	2828	36	49	163	32	33
s9234	5597	36	39	211	39	58
s13207	8027	31	121	669	50	59
s15850	9786	14	87	597	62	82
s38417	22397	28	106	1636	32	47
s38584	19407	12	278	1452	47	56

generate random switching activity values for these circuits since such information is not available.[2] We assume unit delay for all gates in the circuits. Table 1 shows the statistical information of benchmark circuits. We provide the number of gates, PI, PO, and FF for each circuit. Dr and Ds represent the lower bound on retiming delay and static delay, which are calculated by assigning zero delay to all edges and performing retiming and static timing analysis. We note that retiming can improve the delay results significantly. For example, delay can be reduced by 32% for s38417 with retiming, which makes retiming a very attractive choice for delay optimization. This explains why our GEO-PD algorithm focuses on retiming delay as opposed to static delay.

We conduct experiments using ESC [2], GEO [3], GEO-P and GEO-PD algorithms. ESC is a state-of-the-art cutsize driven multi-level algorithm, and GEO is a state-of-the-art simultaneous cutsize and delay driven multi-level algorithm. GEO-P is obtained by setting delay weights of GEO-PD to zero for power optimization only. Lastly, GEO-PD is a simultaneous power and delay driven multi-level algorithm. For partitioning (floorplanning) we report cutsize (wirelength), retiming delay, static delay, visible power and total power. Note that the delay and power results are based on block location in case of floorplanning. In case of partitioning, we use user specified parameters k_d=2 for the delay of cut edge and k_p=2 for the ratio between wire and gate capacitance as discussed in Section 2.3. We report 64 ways partitioning and 8×8 floorplanning results. We report the average runtime of each algorithm measured in second.

Table 2 shows the partitioning results among ESC, GEO, GEO-P, and GEO-PD. We first note that the delay improvement of GEO over ESC is not significant. In fact, the retiming delay results got worse by an average margin of 2%, whereas the static delay improved by 4%. GEO-P improves ESC by an average margin of 12% for visible power and 4% for total power at the cost of 24% increase in cutsize. Finally, GEO-PD obtains 2% worse retiming delay and 7% better visible power results than ESC at the cost of 8% increase in cutsize.

The delay improvement of GEO-PD is significantly more visible in floorplanning where the solution space is much larger than partitioning. Table 3 shows the floorplanning results among ESC, GEO, GEO-P, and GEO-PD. GEO has 10% better retiming delay than ESC at the cost of 16% increase in wirelength. GEO-P has 21% better visible power than ESC at the cost of 10% increase in wirelength. Finally, GEO-PD has 5% better retiming delay and 12% better visible power than ESC at the cost of 25% increase in wirelength. Table 4 reveals more details on how GEO-PD improves ESC results in floorplanning. In particular, GEO-PD improves the retiming delay of s38584 by 21%. The visible power improvement is as much as 31% for s9234. Moreover, the retiming delay and visible power improvement is consistent among all 10 circuits. In overall, GEO-PD reveals a smooth wirelength, delay, and power tradeoff curve and improves both delay and power results of ESC at the cost of increase in wirelength.

5. Conclusions

To the best of our knowledge, this is the first paper addressing both delay and power optimization in multi-level partitioning and floorplanning. In addition, we demonstrated the importance of optimizing the retiming delay and visible power as opposed to the conventional static delay and total power. We demonstrated how cutsize and wirelength have conflicting objectives against power and delay and proposed an effective algorithm GEO-PD for smooth delay, power, and wirelength tradeoff.

[2] The sis package [11] can compute the switching activity for sequential circuits, but it takes a prohibited amount of runtime even for a circuit with a few thousand gates.

Table 2. Comparison among ESC, GEO, GEO-P, and GEP-PD on 64 ways partitioning. Each algorithm reports cutsize, retiming delay (Dr), static delay (Ds), visible power (Pv) and total power (Pt).

	ESC					GEO					GEO-P					GEO-PD				
ckt	cut	Dr	Ds	Pv	Pt	cut	Dr	Ds	Pv	Pt	cut	Dr	Ds	Pv	Pt	cut	Dr	Ds	Pv	Pt
b17o	3418	59	79	3403	4888	3360	65	83	3404	4889	3842	59	78	3286	4810	3433	64	83	3331	4840
b20o	1808	57	94	1636	2425	1948	56	92	1664	2444	2201	58	94	1533	2357	1958	55	92	1622	2416
b21o	1811	57	96	1565	2389	1982	55	84	1656	2450	2334	54	90	1547	2377	1927	57	96	1587	2403
b22o	2251	57	93	2108	3311	2352	59	97	2161	3347	2712	60	98	2037	3263	2418	61	99	2108	3311
s5378	472	43	49	208	359	428	40	47	201	354	555	47	49	141	314	470	45	49	168	332
s9234	465	44	86	263	580	459	48	79	266	582	612	48	80	219	551	528	48	84	244	567
s13207	459	72	83	343	827	474	67	78	354	834	661	71	83	306	802	520	69	80	312	806
s15850	551	82	116	383	972	548	82	104	396	980	698	81	115	307	921	595	81	110	346	948
s38417	789	41	61	760	2179	829	41	59	760	2180	951	42	67	638	2098	858	41	59	645	2103
s38584	896	63	74	993	2369	1031	61	72	1102	2442	1019	63	74	850	2273	987	63	74	955	2344
Ratio	1.00	1.00	1.00	1.00	1.00	1.03	1.00	0.96	1.03	1.01	1.24	1.02	1.00	0.88	0.96	1.08	1.02	0.99	0.93	0.98
Time	111					1999					124					2054				

Reference

[1] C. Ababei, N. Selvakkumaran, K. Bazargan, and G. Karypis, “Multi-objective Circuit Partitioning for Cut size and Path-Based Delay Minimization,” *IEEE International Conference in Computer Aided Design*, page 181-185, 2002.

[2] J. Cong and S. K. Lim, “Edge separability based circuit clustering with application to circuit partitioning,” *to appear in IEEE Trans on Computer-Aided Design*, 2003.

[3] J. Cong and S. K. Lim, “Physical Planning with Retiming,” *IEEE International Conference in Computer Aided Design*, page 2-7, 2000.

[4] J. Cong, S. K. Lim, and C. Wu., “Performance driven multi-level and multiway partitioning with retiming,” *ACM Design Automation Conf.*, page 274-279, 2000.

[5] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, “Multilevel hypergraph partitioning: Application in VLSI domain,” *ACM Design Automation Conf.*, page 526-529, 1997.

[6] C. E. Leiserson and J. B. Saxe, “Retiming synchronous circuitry,” *Algorithmica*, page 5-35, 1991.

[7] R. Murgai, R. K. Brayton, and A. Sanglovanni Vicentelli, “On clustering for minimum delay/area,” *IEEE Int. Conf. On Computer-Aided Design,* page 6-9, 1991.

[8] 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.*

[9] H. Vishnu and M. Pedram, “Delay-Optimal Clustering Targeting Low-Power VLSI Circuits,” *IEEE Trans on Computer-Aided Design*, page 639-643, 1995.

[10] C. Fiduccia and R. Mattheyses, “A Linear Time Heuristic for Improving Network Partitions,” *ACM Design Automation Conf.*, page 175-181, 1982.

[11] E. M. Sentovich, et al, "SIS: A System for Sequential Circuit Synthesis", Department of EECS, University of California, Berkeley, Electronics Research Laboratory Memorandum No. UCB/ERL M92/41, 1992.

[12] http://www.cbl.ncsu.edu

[13] http://www.cad.polito.it/tools/9.html

[14] J. Cong, H. Li, and C. Wu, “Simultaneous Circuit Partitioning/Clustering with Retiming for Performance Optimization,” *ACM Design Automation Conf.*, page 460-465, 1999.

Table 3. Comparison among ESC, GEO, GEO-P, and GEP-PD on 8×8 floorplanning. Each algorithm reports wirelength, retiming delay (Dr), static delay (Ds), visible power (Pv) and total power (Pt).

	ESC					GEO					GEO-P					GEO-PD				
ckt	wire	Dr	Ds	Pv	Pt	wire	Dr	Ds	Pv	Pt	wire	Dr	Ds	Pv	Pt	wire	Dr	Ds	Pv	Pt
b17o	9629	70	101	5232	6717	10451	63	94	5697	7219	9982	63	100	4604	6128	10468	61	99	4938	6485
b20o	5772	72	107	3335	4125	6730	79	114	3660	4453	6450	71	107	3101	3925	7277	72	110	3145	3971
b21o	6357	79	127	3458	4282	6618	65	109	3468	4266	6703	75	117	2863	3693	7491	70	113	3235	4068
b22o	7243	77	118	4076	5279	7724	69	103	4473	5676	8570	83	137	3879	5106	8685	76	124	4211	5440
s5378	1502	60	77	384	535	1462	45	71	389	539	1539	57	65	234	407	1597	57	69	269	438
s9234	1425	50	91	427	744	1685	48	101	476	787	1510	52	101	292	623	1683	48	95	296	629
s13207	1525	91	106	747	1231	1925	77	96	900	1378	1803	91	106	536	1032	2367	91	102	634	1125
s15850	1587	99	143	584	1172	2085	90	129	814	1392	1720	96	136	395	1009	2236	100	140	517	1126
s38417	2032	41	71	1158	2578	2695	41	82	1483	2895	2524	43	81	963	2423	2819	41	67	1088	2535
s38584	2973	87	102	1950	3326	3663	68	80	2091	3432	3061	79	91	1619	3043	3546	69	84	1766	3140
Ratio	1.00	1.00	1.00	1.00	1.00	1.16	0.90	0.95	1.14	1.08	1.10	0.98	1.00	0.79	0.88	1.25	0.95	0.96	0.88	0.94
Time	104					2231					121					2257				

Table 4. Performance ratio between GEO-PD and ESC. The entries are computed by GEO-PD results divided by ESC results.

	GEO-PD vs ESC, 64way					GEO-PD vs ESC, 8×8				
ckt	cut	Dr	Ds	Pv	Pt	wire	Dr	Ds	Pv	Pt
b17o	1.00	1.08	1.05	0.98	0.99	1.09	0.87	0.98	0.94	0.97
b20o	1.08	0.96	0.98	0.99	1.00	1.26	1.00	1.03	0.94	0.96
b21o	1.06	1.00	1.00	1.01	1.01	1.18	0.89	0.89	0.94	0.95
b22o	1.07	1.07	1.06	1.00	1.00	1.20	0.99	1.05	1.03	1.03
s5378	1.00	1.05	1.00	0.81	0.92	1.06	0.95	0.90	0.70	0.82
s9234	1.14	1.09	0.98	0.93	0.98	1.18	0.96	1.04	0.69	0.85
s13207	1.13	0.96	0.96	0.91	0.97	1.55	1.00	0.96	0.85	0.91
s15850	1.08	0.99	0.95	0.90	0.98	1.41	1.01	0.98	0.89	0.96
s38417	1.09	1.00	0.97	0.85	0.97	1.39	1.00	0.94	0.94	0.98
s38584	1.10	1.00	1.00	0.96	0.99	1.19	0.79	0.82	0.91	0.94
Ave	1.08	1.02	0.99	0.93	0.98	1.25	0.95	0.96	0.88	0.94