7659, England, UK
Journal of Information and Computing Science
Vol. 2, No. 1, 2007, *pp*. 66-70

Multi-level Sequential Circuit Partitioning for Delay Minimization of VLSI Circuits

K. Somasundaram [+]

Department of Mathematics, Amrita Viswa Vidyapeetham, Coimbatore-641 105, India.

(*Received July 13, 2006, Accepted October 5, 2006*)

Abstract. Sequential graph partitioning algorithms have been developed to fulfill the requirements of emerging multi-phase problems in circuit delay models. In this paper we propose a heuristic algorithm for *k*-partition, which minimizes the circuit delay and cut size. Experimental results with MCNC benchmark circuits have shown that the delay in the circuit can be reduced by marginally in comparison with the other algorithms [2,3,11].

Keywords: Graph Partition, Circuit Partition and Delay minimization.

1. Introduction

Graph partitioning is one of the important and well-known problems in circuit design and testing. The objective is to divide the circuits into blocks such that each component falls within prescribed sizes and the complexity of connection between these components is reduced. Many of the VLSI physical design problems assumed to minimize the area of a chip occupied by the wires and cells; it can be modeled by embedding a graph in a grid. Hence, developing a good partitioning algorithm for an undirected graph is critical. In general, graph partitioning problem is an NP-complete. However, many algorithms have been developed that can find a reasonably good partitioning for cut size minimization for various types of graphs. Hendrickson & Leland [6] and Karypis & Kumar [8,9,10] introduced a new class of multilevel graph partitioning techniques. Several authors introduced matrix partitioning, in particular sparse matrix partitioning by Riyavong [19] and Karypis & Kumar [7]. These multilevel schemes provided excellent graph partitioning but have moderate computational complexity. Though these multilevel algorithms are quite fast compared to spectral methods, parallel formulations of multilevel algorithms are needed. Savage & Wloka [18] have extensively studied a graph embedding heuristic based on parallel Mob heuristic for graph partitioning. Kernighan and Lin [12] developed a heuristic algorithm (K-L algorithm) in polynomial time for two way uniform partitioning of a graph with 2*n* vertices. In this paper we describe a new efficient algorithm that aims at finding a good *k*-partition of circuits and minimizes the total cut size and the circuit delay during a testing process. Cong and Wu [3] proposed a global clustering based multi-level partitioning algorithm for performance optimization. They compute a delay minimal *k*-way partitioning first, and then gradually reduce the cut size while keeping the circuit delay by de-clustering and refinement. For cut size minimization, hMetis [11] significantly improved the solution quality based on a very efficient Multi-level optimization method. However, its coarsening step explores only the very local connectivity information. Hence, it cannot guarantee to produce small delay in general. Cong et al., [2] proposed a performance driven clustering/partitioning with retiming algorithm. Given an area bound for each cluster, PRIME [2] can compute a quasi-optimal solution if node duplication is allowed. For circuit partition without node duplication, a heuristic version of PRIME can still achieve the smallest delay compared with other partition algorithms. Muthukumar and Selvaraj [15] gave a comparison of heuristic algorithms for variable partitioning in circuit implementation. Cherng and Chen [4] have shown a new multi-level bipartitioning algorithm MLP based partitioning process, which integrates a clustering technique and an iterative improvement. The clustering algorithm is used to reduce the partitioning complexity and to improve the performance of partitioning. We can identify a few problems for all previous timing driven partitioning approaches: (i) Unrealistic delay models: It is common to use the general-delay model, which considers delay 1 for all gates, delay 0 for interconnects inside a partition, and a constant delay for interconnects

[+] *E-mail address*: s_sundaram@ettimadai.amrita.edu

between partitions [1,3,13,16]. (ii) Unrealistic simplifications: For instance, circuits are mapped to two-input gates only [3]. (iii) The run times for even moderate-sized circuits are too long. In this paper we use the second method as in [3].

The paper is organized as follows. Section 2 describes relevant notation, definitions for the graph partition problem and circuit delay model. Section 3 describes details of *k*-partition algorithms. Section 4 gives the circuit delay minimization using *k*-partition. Section 5 shows the excremental results with MCNC benchmark circuits. Discussion and conclusion are given in section 6.

2. Preliminaries

A *k*-partition problem is to partition the vertices of a graph into *k* roughly equal parts each with *m* vertices such that the number of edges connecting the vertices in different parts is minimized. That is, for a given weighted graph $G = (V, E)$, let c_{ij} be the weight or cost of an edge e_{ij} in E and k be the number of partitions of G. The *k*-partition of G is to find a set of disjoint subsets of $V_1, V_2,\ldots,V_k$ such that $\bigcup_{n=1}^{k} V_n = V$, for all e_{ij} in E, $v_i \neq v_j$, and $C = \Sigma\, c_{ij}$ is minimized. Here, C is called the cut cost of the partition.

The simplest partitioning which still contains all the significant features of larger problems is that of finding a minimal cost partition of a given graph with n vertices. By iterative (divide-conquer) procedure of K-L algorithm for 2-partitioning, one can obtain *k*-partitioning. But, in this case we cannot obtain the optimum cut size. The K-L algorithm has a time complexity of $O(n^2 \log n)$.

We can generate a *k*-partition, each with *m* elements. Starting with a random partitioning of *k* sets of *m* vertices each, the K-L algorithm is applied for two way partitioning procedure on each pair of partitions. Since there are $\binom{k}{2}$ pairs to consider, the time complexity for one pass through all pairs for the $O(n^2)$-procedure is $O(k^2n^2)$. Hence, the generalization of this procedure leads to non-polynomial time.

In this paper we have proposed an algorithm for *k*-partitioning in heuristic nature and it will minimize the circuit delay. The K-L algorithm for 2-partitioning is briefed as follows. Let S be a set of $2n$ vertices, with an associated cost matrix $C = (c_{ij})$, $i, j = 1, 2,\ldots,2n$. Starting with any arbitrary partitions A and B of S, it is tried to reduce the initial external cost $C = \Sigma\, c_{ij}$ by series of interchanges of subsets of A and B; the subsets are chosen by an algorithm described in [12]. We define for each $a \in A$, an external cost $E_a = \sum_{y \in B} c_{ay}$ and the internal cost $I_a = \sum_{x \in A} c_{ax}$. Similarly, E_b and I_b are defined for each $b \in B$. Let $D_s = E_s - I_s$ for all $s \in S$. The gain in the cut cost is obtained as $\Delta_i = D_{a_i} + D_{b_i} - 2c_{a_i b_i}$, for all $a_i \in A$, $b_i \in B$. Now, a 'p' is found such that the partial sum $\sum_{i=1}^{p} \Delta_i$ is maximal, and the corresponding a_p and b_p are interchanged to B and A respectively. Since we construct a sequence of gains Δ_i, $i = 1,2,\ldots,n$, and find the maximum partial sum, the process does not terminate immediately when some Δ_i is negative. This means that the process can sequentially identify sets for which the exchange of only a few elements would actually increase the cost, while the interchange of the entire sets produces a net gain.

Any given partitioning problem is to decompose a given circuit into *k* blocks for a given *k* with balanced area. Retiming is a well-known sequential circuit optimization technique for delay optimization by moving flipflops without changing the circuit behavior. A retiming hypergraph is a directed graph $G(V,E)$ and c_{ij} is the set of edge e_{ij} weights representing the numbers of flipflops on edges. The *general delay model* is used in this paper, which assumes that each gate v has a delay h_v, intra-block delay (local interconnect delay within each block) of h and inter-block delay (interconnect delay between blocks) of H, where $h < H$. The size of the each gate v is a_v. we ignore the setup and hold times as well as the size of flipflops as assumed in [2,4]. We ignore the size of flipflops, since it is difficult to predict which partition flipflop will be in before retiming. The circuit delay is a measure as the longest combinational path delay from a primer input of a flipflop output to a primary output or a flipflop input. Our objective is to perform a multi-level partitioning with retiming for the minimum delay, while reducing the cut size as much as possible.

3. *k*-Partitioning

We call our algorithm as Sequential Partitioning Algorithm (SPA). The basic idea of SPA is given as follows: Consider a graph G with a vertex set V with km vertices. Let V_1 be the initial partitioning with m elements. Let $U = \{u_i\}$ be the collection of dummy vertices, which are to be added with G such that $c_{v_i u_j}$ and $c_{u_j v_i}$ are zero for all $v_i \in V$, $u_i \in U$. Let $A_1 = V_1 \cup U$. Using K-L algorithm, the two way partitioning, A_1 and $B_1 = V - V_1$ are improved. A new set of bipartition A_1 and B_1 of $n-m$ vertices is obtained. By discarding the dummy vertices in A_1, we get one partition of m vertices. Let $V_2 = B = V - V_1$ be the next set with m vertices. Applying the above procedure for k times, we get sets A_i of vertices, $i = 1,2,\ldots,k$, such that the total cut cost $C = \Sigma\, c_{ij}$ is minimum. Initially we add $n-m$ dummies, and perform the procedure on it. The resulting partition will assign the dummy elements to the two subsets so as to minimize the external cost and at this point the dummies are discarded, leaving a partition into two subsets that satisfy the size constraints.

Let U be the set of all dummy vertices u_i. Then $D_{u_i} = 0$, for all $u_i \in U$, also we have $\Delta_i = D_{u_i} + D_{b_i} - 2c_{u_i b_i} = 0$, for all $u_i \in U$ and $\Delta_i = D_{a_i} + D_{u_i} - 2c_{a_i u_i} = 0$, for all $u_i \in U$. Hence dummy vertices may not have any impact on the optimality of the problem.

Initially, the computation of the D values is an n^2 procedure, since at most $(n - m)$ vertices are taken into another set of $(n - m)$ elements. The time required for updating the D values is proportional to number of values to be updated. Hence total updating time is proportional to n^2. The selection of pairs a_i and b_i in A and B for exchange can be done by sorting the D values, and selecting the maximum values of D's since maximum gain Δ_i is obtained from maximum of D_i's. The time required for this procedure is approximately k times $n^2 \log n$.

4. Delay Optimization

The multi-level sequential partition is to form a number of clusters for a circuit under a given area bound. Under the general delay model with node delay h_v, intra-block delay h and inter-block delay H, our objective is to minimize the circuit delay after optimal retiming. For minimizing the delay, the hypergraph G of a circuit, $h(e)$ is the edge weight and $d(e)$ is the edge delay. We minimize the delay by clustering the hypergraph and make a k-partition. The total delay in a circuit and delay after the partitioning the circuit is very significantly. The dummy node in the hypergraph corresponds to the nil gates, it does not perform any operation. The total cut cost corresponding the circuit delay is $C = \Sigma\, c_{ij}$. The optimal clustering solution, we can focus on cut size minimization and delay minimization.

The partitioning algorithm that can minimize the objective cut cost C, the number of wire cuts and the objective H, the number of times most critical paths are cut. However, the objectives are dissimilar objectives, which means that optimizing C alone does not necessarily imply that H is also optimized and vice-versa. Hence we formulate an objective function $Z = \alpha \dfrac{C}{C_0} + (1-\alpha)\dfrac{H}{H_0}$, where C_0 and H_0 are respectively the optimal values of cut size and delay respectively, and the α is the performance vector lies in [0, 1]. The objective function Z gives the linear combination of the cut size and delay in line segment between C and H. Minimization of Z computes a partition such that it optimizes both the cut size and delay. The delay objective component is expressed as a combination of all factors. Hence $H = \beta \sum_{i=1}^{|E_k|} c_i q_i + (1-\beta) \sum_{j=1}^{k} H_j k_j^r$, where the parameter β is an averaging value in [0, 1], c_i is the edge weight of edge e_i, q_i is either 0 or 1 depending on whether edge e_i is cut or not, $|E_k|$ is the number of edges that form the k paths, H_j is the current delay of the j^{th} critical path, k_j is the number of times that the j^{th} critical path has been cut so far, and r is used for even finer tuning. During the partition phase, when net is cut, it is assigned a certain delay that will be used to recompute all delays on the paths that include that net.

5. Experimental Results

The K-L algorithm runs $O(n^2 \log n)$ time but the above procedure will take the complexity as k times of $O(n^2 \log n)$. Hence the algorithm will remain as in polynomial time of heuristic nature. We hve implemented

our algorithm in C on Intel Pentium IV 400 Mhz PC with 1 GB memory.

If there are n vertices in a graph initially then this algorithm will generate as many n-m dummy vertices, on i^{th} stage there are $n–im$, i=1,2,…,(k-1), dummy vertices will be generated and this will be straightly decreasing on each stage, which is shown in Figure-1. The algorithm leads to a simple space complexity due to generation of dummy vertices in the initial stage. Though the space complexity in an algorithm will not have any impact, we can distribute the vertices into some processors based on their incidence relationship.

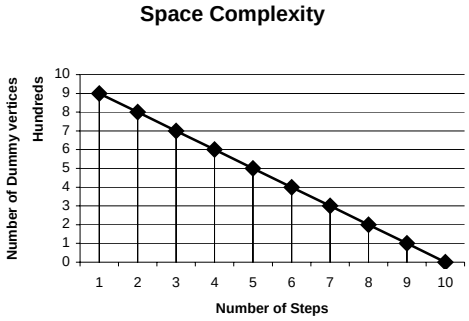

Figure 1 : Space complexity for generation of dummy vertices

Table 1. Comparison SPA with PRIME, hMetis, MLPR and HPM [3].

16 way partition							
			PRIME	hMetis	MLPR	HPM	SPA
circuit	Number of nodes	φ_{lb}	φ	φ	φ	φ	φ
S1423	530	57	67	120	68	104	102
S838.1	376	1519	29	43	28	40	39
S5378	1578	1113	27	35	25	31	31
S9234.1	1368	21	30	30	29	31	30
S1196	509	24	43	53	43	46	42
S13207.1	3376	32	45	55	41	51	50
S15850.1	4016	37	65	57	51	67	67
S38417	9897	27	39	38	30	38	37
S38584.1	13551	29	34	39	29	31	26
average		28.3	43.8	50	38.3	48.8	46.4

In our circuit test, we assume each two-input gate is an area of 1 unit and a delay of 1 unit. The local interconnect delay within a partition is 0 and the inter-partition delay is 5 as used in [2,3]. We used MCNC benchmark circuits for delay minimization. We compared our algorithm with other algorithms for bi-partitioning and multi-partitioning. We observed that we could satisfyingly minimize the delay using SPA. Table 1 shows the circuit delay results with various MCNC benchmark circuits.

6. Conclusion

We have described a new formulation of the k-partitioning graph algorithm(SPA). It is shown that this algorithm is able to minimizes the circuit delay. Our experimental analysis shows that, SPM is better able to find a good delay minimization. We believe our k-partition algorithm can be adapted to other problems in various stages in logic synthesis and physical delay for performance optimization.

7. Acknowledgements

This work was support in part by a grant from the office of Vice chancellor, Amrita Vishwa Vidyapeetham AVVP/05/KSS/Maths/C/IFRP/09.

8. References

[1] C. Ababei, N..Selvakkumaran, K. Bazargan and G. Karypis. Multi-Objective Partitioning for Cutsize and Path-Based Delay Minimization. *ICCAD Conference*. pp181-185, 2002.

[2] J. Cong, H. Li, and C. Wu. Simultaneous Circuit Partitiong/Clustering with Retiming for performance Optimization. In *Proc. ACM/IEEE Design Automation Conference.* pp460-465, 1999.

[3] J. Cong, C. Wu, Global Clustering-Based Performance-Driven Circuit Partitioning, *Proc. ISPD*, pp149-154, 2002.

[4] J. S. Cherng and S. J. Chen. An Efficient Multi-Level Partitioning Algorithm for VLSI Circuits. In *Proc. IEEE conference on VlsI Design.* pp70-79, 2003.

[5] S. H. Gerez, *Algorithms for VLSI Design Automation*. John Wiley &Sons, 1999.

[6] B. Hendrickson and R. Leland. A Multilevel Algorithms for Partitioning Graphs. *Technical report* SAND 93-1301.

[7] G. Karypis and V. Kumar. A Parallel Algorithm for Multilevel Graph Partitioning and Sparse Matrix Ordering. *Technical Report.* TR 95-036, 1995.

[8] G. Karypis and V. Kumar. Analysis of Multilevel Graph Partitioning. *Technical Report* TR 95-035, Department of Computer Science. University of Minnesota, 1995.

[9] G. Karypis and V. Kumar. A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM journal of Scientific computing.* 1998, **20**(1):359-392.

[10] G. Karypis and V. Kumar. Multilevel k-way Partitioning Scheme for Irregular Graphs, *Journal of Parallel and Distributive computing*. 1998, **48**(1):96-129.

[11] G. Karypis, R. Aggarwal, V. Kumar and S. Shekhar. Multilevel Hypergraph Partitioning: Applications in VLSI Domin, In *ACM / IEEE Design Automation Confeence.* pp526-529, 1997.

[12] B. W. Kernighan and S. Lin. An Efficient Heuristic Procedure for Partitioning Graphs. *The Bell System Technical Journal.* 1970.

[13] J. Minami, T. Koide, S. Wakabayashi. An Iterative Improvement Circuit Partitioning Algorithm under Path Delay Constrains. *IEICE Trans. Fundamentals.* Dec. 2000.

[14] R. Murgai, R.K. Brayton, and A. Sangiovanni-Vincentelli. On Clustering for Minimum Delay / Area, *In IEEE International conference on CAD.* pp6-9, 1991.

[15] V. Muthukumar and H. Selvaraj. Comparison of Heuristic Algorithms for Variable Partitioning in Circuit Implementation. In *Proc. IEEE Design Automation Conference.* pp256-260, 2003.

[16] S. L. Ou, M. Pedram. Timing-driven Partitioning Using Iterative Quadratic Programming, at http://atrak.usc.edu/~massoud/, 2001.

[17] S.M. Sait and H. *Youssef, VLSI Physical Design Automation*. IEEE Press. 1995.

[18] J. E. Savage and M. G. Wloka. Parallelism in Graph Partitioning. *Journal of Parallel and Distributed Computing.* **13:**257-272,1991.

[19] S. Riyavong. Experiments on Sparse Matrix Partitioning. *CERFACS working Note* WN/PA/03/32, CERFACS, France.

[20] H. Yang and D. F. Wong. Circuit Clustering for Delay Minimization under Area and Pin Constrains. *In ED&TC*, pp65-70, 1995.