A Hybrid Genetic Search for Circuit Bipartitioning

Jong-Pil Kim and Byung-Ro Moon
School of Computer Science & Engineering, Seoul National University
Shilim-dong, Kwanak-gu, Seoul, 151-742 Korea
{jpkim, moon}@soar.snu.ac.kr

Hypergraph partitioning is an important problem and has many applications including design automation of VLSI chips and multi-chip systems. In this paper, we proposed a hybrid genetic algorithm for partitioning a hypergraph into two disjoint graphs of minimum cut size.

The Fiduccia-Matheyses algorithm (FM) [1] is a representative iterative improvement algorithm for a hypergraph partitioning problem. But the quality of the FM is not stable. Kim and Moon [3] introduced lock gain as a primary measure for choosing the nodes to move. It uses the history of search more efficiently. Lock gain showed excellent performance for general graphs. We adapt the lock gain for hypergraphs within the framework of FM.

Lock gain was originally designed for the general graphs [3]. Thus, to apply the lock gain to the hypergraph bisection, some modification is necessary. We propose a new lock gain calculation method for the hypergraph bisection. Let's define $l_e(v)$ to be the lock gain of a node v due to the net e. $l_e(v)$ is obtained as the following. We assume that the node v is on the left side without loss of generality. L, R, L', and R' are defined to be #nodes on the left side, #nodes on the right side, #locked nodes on the left side, and #locked nodes on the right side, respectively.

If all nodes on the right side are locked and there is no locked node on the left side ($L > 0,\ \ L' = 0,\ \ R = R' > 0$), or if there exists locked nodes on the right side and there is no free node on the left side ($L = 1,\ \ L' = 0,\ \ R \geq R' > 0$), then $l_e(v) = 1$. On the other hand, if all nodes are on the left side and at least one node is locked ($L > L' > 0,\ \ R = R' = 0$), or if all nodes on the left side except v are locked and there is no locked node on the right side ($L - L' = 1,\ \ R > 0,\ \ R' = 0$), then $l_e(v) = -1$. The lock gain $l(v)$ of the node v is defined to be $l(v) = \sum_{e \in N(v)} l_e(v)$ where $N(v)$ is a set of nets to which the node v is connected.

Table 1: Bipartition Cut Sizes

Circuits	GA		hMetis150	
	Average[1]	CPU[2]	Average[3]	CPU[2]
Test02	**88.18**	13.58	89.93	10.81
Test03	**58.00**	4.35	59.96	10.31
Test04	**51.28**	8.48	54.85	8.82
Test05	71.88	27.84	**71.44**	17.23
Test06	**63.04**	5.73	64.16	10.08
Prim1	53.66	1.33	**53.02**	5.28
Prim2	**148.87**	24.12	178.39	30.70

1. The average cut size of 100 runs
2. CPU seconds on Pentium III 866MHz
3. The average cut size of 100 runs, each of which is the best of 150 runs of hMetis

We tested the proposed algorithm on 7 ACM/SIGDA benchmarks. We examine the performance of the hybrid GA which uses the lock-gain based FM as a local optimization engine against a well-known partitioner hMetis [2]. Because the GA took roughly 150 times more than a single run of hMetis, it is not clear how critical the genetic search is to the performance improvement. Thus, hMetis150, that is a multi-start version of hMetis with 150 runs, was compared.

Table 1 shows the performance of the GA. On the average, the proposed GA performed best in 5 graphs among 7.

References

[1] C. Fiduccia and R. Mattheyses. A linear time heuristics for improving network partitions. In *19th IEEE/ACM Design Automation Conference*, pages 175–181, 1982.

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

[3] Y. H. Kim and B. R. Moon. A hybrid genetic search for graph partitioning based on lock gain. In *Genetic and Evolutionary Computation Conference*, pages 167–174, 2000.

This work was supported by KOSEF through Statistical Research Center for Complex Systems at Seoul National University(SNU) and Brain Korea 21 Project. The RIACT at SNU provided research facilities for this study.