Circuit Bipartitioning Using Genetic Algorithm

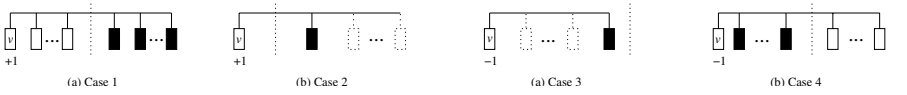
Jong-Pil Kim and Byung-Ro Moon

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

Abstract. In this paper, we propose a hybrid genetic algorithm for partitioning a VLSI circuit graph into two disjoint graphs of minimum cut size. The algorithm includes a local optimization heuristic which is a modification of Fiduccia-Matheses algorithm. Using well-known benchmarks (including ACM/SIGDA benchmarks), the combination of genetic algorithm and the local heuristic outperformed hMetis [3], a representative circuit partitioning algorithm.

The Fiduccian-Matheyses algorithm (FM) [2] is a representative iterative improvement algorithm for a hypergraph partitioning problem. FM improves an initial solution through short-sighted moves based on the gain. Thus, the quality of the FM is not stable. Kim and Moon [4] introduced lock gain as a primary measure for choosing the node 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.

To apply the lock gain to the hypergraph bisection, considerable modification is necessary for the lock gain calculation method, because lock gain was originally designed for the general graphs [4]. 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.



Positive lock gain Negative lock gain

If all nodes on the right side are locked and there is no locked node on the left side (Case 1) or if there exists locked nodes on the right side and there is no free node on the left side (Case 2) then $l_e(v) = 1$. On the other hand, if all nodes are on the left side and at least one node is locked (Case 3) or if all nodes on the left side except v are locked and there is no locked node on the right side

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2724, pp. 2408–2409, 2003.

Table 1. Bipartition Cut Sizes of GA and hMetis

Circuits	GA			hMetis200		
	Average[1]	CPU[2]	Best	Average[3]	CPU[2]	Best
Test02	**88.16**	17.20	88	89.81	12.12	88
Test03	**58.00**	5.59	58	59.90	11.74	58
Test04	**51.15**	8.90	51	54.77	10.02	54
Test05	72.05	27.85	71	**71.33**	19.22	71
Test06	**63.12**	4.89	63	64.05	12.30	64
Prim1	53.87	1.14	53	**53.00**	6.33	53
Prim2	**149.02**	19.45	146	177.76	34.39	146
19ks	**110.22**	21.44	110	110.81	25.89	110
Industry2	186.97	211.25	183	**181.15**	233.35	180

1. The average cut size of 100 runs
2. CPU seconds on Pentium III 1GHz
3. The average cut size of 100 runs, each of which is the best of 200 runs of hMetis

(Case 4) then $l_e(v) = -1$. Then, 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.

We tested the proposed algorithm on 9 benchmarks including ACM/SIGDA benchmarks [1]. We compare the performance of the hybrid GA which uses the lock-gain based FM as a local optimization engine against a well-known partitioner hMetis [3]. Because the GA took roughly 200 times more than a single run of hMetis, it is not clear how critical the genetic search is to the performance improvement. Thus, hMetis200, that is a multi-start version of hMetis with 200 runs, was compared.

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

Acknowledgments. This work was partly supported by Optus Inc. and Brain Korea 21 Project. The RIACT at Seoul National University provided research facilities for this study.

References

1. BENCHMARK. http://vlsicad.cs.ucla.edu/~cheese/benchmarks.html.
2. C. Fiduccia and R. Mattheyses. A linear time heuristics for improving network partitions. In *19th IEEE/ACM Design Automation Conference*, pages 175–181, 1982.
3. 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.
4. 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.