

Application of Semidefinite Programming to Circuit Partitioning Problem*

Changhui Choi
Department of Management Sciences
The University of Iowa
Iowa City, Iowa 52242, U.S.A.

Yinyu Ye
Department of Management Sciences
The University of Iowa
Iowa City, Iowa 52242, U.S.A.

June 6, 2000

Abstract

Graph partitioning is an important step in various applications including VLSI design. Even though partitioning problem in VLSI design is an NP-complete combinatorial optimization problem, most of the dominant partitioners are tend to rely on heuristics rather than solving this problem as an optimization problem. In this report we propose a new approach that

In this paper we apply Semidefinite Programming to circuit partitioning problems. Unlike other iterative improvement techniques, we translate hypergraph into weighted undirected graph and solve it as a graph partitioning problem. Semidefinite relaxation is used in translation, then vector solution is recovered using heuristic and randomized methods.

Key words. semidefinite programming, circuit partitioning, graph partitioning, dual potential reduction algorithm, semidefinite relaxation, hyper-graph.

*This work was partially supported by NSF grants DMI-9522507 and DMS-9703490.

1 Introduction

VLSI circuit partition is used to reduce the VLSI chip area by reducing the number of components and interconnections in the layout of large circuits to enable efficient parallel simulation of circuits and reduce time delays. In this process bisection is used to partition circuits into two parts or partition the smaller parts into even smaller parts by minimizing the connection between parts being partitioned while balancing the areas on each side.

Usually circuit C is represented by a hypergraph or netlist $H(V,E)$ with V as being a set of modules and E as a set of signal nets. Here is an example of simple circuit.

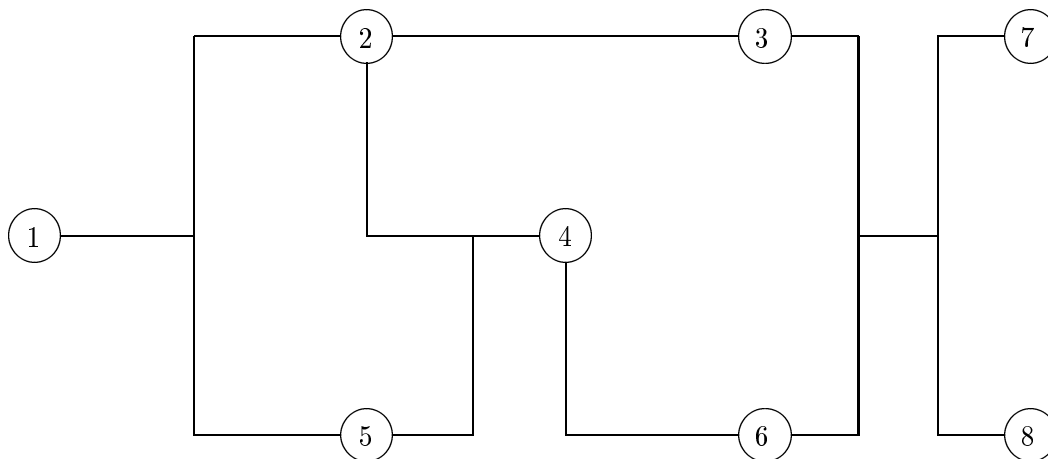


Figure 1 : Example of a simple circuit

As shown in the Figure 1, multiple modules are connected with single arcs. Vertices(which are often called modules) 1,2 and 5 are connected in a single net. In this example sets of vertices $(1,2,5)$, $(2,4,5)$, $(2,3)$, $(4,6)$ and $(3,6,7,8)$ constitute a set of signal nets E .

In circuit partitioning problem, we want to divide a circuit into two parts with a specified sizes on both sides while minimizing the number of connections(nets).

First problem we had to solve was devising a way to convert this hypergraph into an undirected graph that fits the SDP(Semidefinite programming) which requires weights between a pair of vertices. This had to be done because in the original problem we have to minimize the number of nets on the cut. Yet in SDP we need to design this so that sum of weights between vertices are used instead of nets.

2 Translating hypergraph

There are many ways to approximate hypergraph format into weighted undirected graph. The method we used is assigning the same weights $4/(\text{number of modules in a net})^2$ to all the arcs assuming that all the modules are connected to all of the other modules. This design has advantage over the other models in the fact that it tends give more freedom to bigger nets. Since weights in the bigger net is relatively smaller than those of smaller nets, algorithm will prefer cutting arcs in the bigger nets than smaller nets to make areas on both sides as equal as possible. Then why we want to cut bigger nets? The reason is however we may cut a big net, it is counted as one net on the cut. So if we can use bigger nets to balance both sides, we do not need to use smaller nets for balancing areas. Figure 2 is undirected graph form generated for the circuit shown in Figure 1.

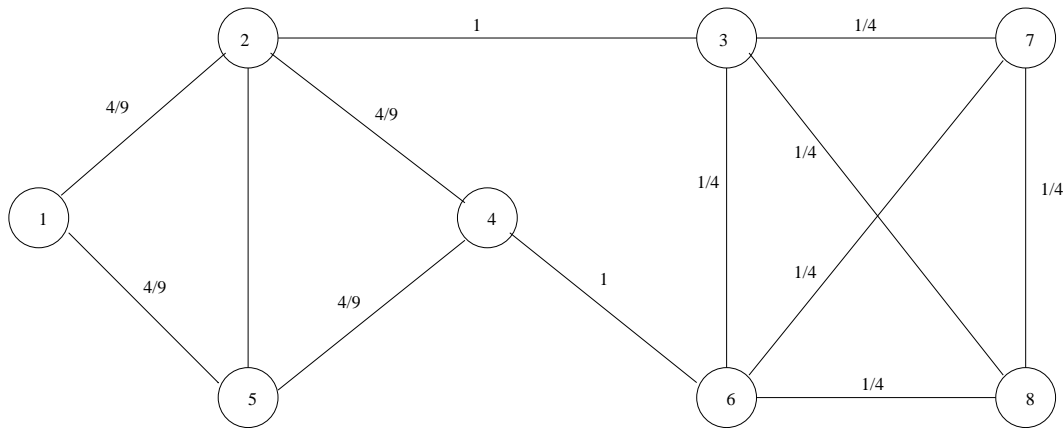


Figure 2 : Changing hypergraph into undirected graph

As in the figure 2, we use graphical representation to apply bipartitioning problem to SDP assigning $4/(\text{number of modules in a net})^2$. (Weight on the arc between module 2 and 5, $8/9$, is the sum of the weights $4/9$ from the net (1, 2, 5) and $4/9$ from the net (2, 4, 5).)

In the real circuits' cases modules are composed of pins and cells which are called as modules. They are like vertices in the graph. Cells have small areas that fill the circuit which should be balanced on both sides. However we assumed that all the cells have weight 1 for the problem above.

3 SDP implementation

Next step of this application is implementing a model that can be solved using SDP. We first build a model that makes both sides as equal as it can be while minimizing the sum of weights on the cut. Here is the implementation of the problem.

$$\begin{aligned}
& \text{(Weighted Ecut)} && \text{Min} && L \bullet X \\
& && \text{Subject to} && \text{diag}(X) = e, \\
& && && (aa^T) \bullet X = 0, \\
& && && X \succeq 0.
\end{aligned} \tag{1}$$

In (1) a is a vector of areas for each module. And $(aa^T) \bullet X = 0$ is a relaxation of the condition $a^T x = 0$ which forces the areas on both sides to be the same. And $\text{Min } L \bullet X$ is used to minimize the sum of weights on the cut. L is laplacian matrix that generated by doing $L = [\text{diag}(We) - W]$ where W is an adjacency matrix generated in section 2.

w_{ij} , i th row j th column element of matrix W , represents the weight on the connection between node i and node j . And it is 0 when they are not connected.

For the translated graph in Figure 2, matrix W will look like;

$$\begin{pmatrix}
0 & 4/9 & 0 & 0 & 4/9 & 0 & 0 & 0 \\
4/9 & 0 & 1 & 4/9 & 8/9 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1/4 & 1/4 & 1/4 \\
0 & 4/9 & 0 & 0 & 4/9 & 1 & 0 & 0 \\
4/9 & 8/9 & 0 & 4/9 & 0 & 0 & 0 & 0 \\
0 & 0 & 1/4 & 1 & 0 & 0 & 1/4 & 1/4 \\
0 & 0 & 1/4 & 0 & 0 & 1/4 & 0 & 1/4 \\
0 & 0 & 1/4 & 0 & 0 & 1/4 & 1/4 & 0
\end{pmatrix} \tag{2}$$

And vector a will have all 1 as its elements for this test problem. When we know areas of modules, a will be the vector that has areas as its elements.

Once circuit is translated into SDP format file, we solve this problem using DSDP(dual-scaling semidefinite programming solver written in COPL) which will find the relaxed vector optimal solution \hat{x} for this problem, which satisfies $\hat{x}\hat{x}^T = \hat{X}$

Dual-form of this problem is;

$$\begin{aligned}
 \text{(DSP)} \quad & \text{Maximize} \quad \sum_{i=1}^n y_i \\
 & \text{Subject to} \quad Iy + \lambda(aa^T) + S = L, \quad S \succeq 0,
 \end{aligned} \tag{3}$$

where y is a vector with the dimension n (number of nodes in the problem) which, combined with λ , makes S positive semidefinite.

Then Dual-potential function of this problem becomes;

$$\psi(y, \lambda, \bar{z}) = \rho \ln(\bar{z} - e^T y) - \ln \det S,$$

where

$$\bar{z} = L \bullet X \text{ for some feasible } X, I \text{ is an identity matrix and } \rho \geq n + \sqrt{n}.$$

Beginning with a strictly feasible dual point (y^k, λ^k, S^k) and upper bound \bar{z}^k , each iteration solves the problem:

$$\begin{aligned}
 \text{Minimize} \quad & \nabla \psi^T(y^k, \lambda^k, \bar{z}^k)(v - v^k) \\
 \text{Subject to} \quad & \| (S^k)^{-0.5} \mathcal{A}^T(y - y^k, \lambda - \lambda^k) (S^k)^{-0.5} \| \leq \alpha,
 \end{aligned} \tag{4}$$

where

$$\nabla \psi(y, \lambda, \bar{z}) = -\frac{\rho}{\bar{z} - e^T y} e_{n+1}[1, n] + \mathcal{A}(S^{-1}), \tag{5}$$

$$\mathcal{A}(X) = \begin{pmatrix} x_{11} \\ \vdots \\ x_{nn} \\ (aa^T) \bullet X \end{pmatrix} \quad \text{and} \quad \mathcal{A}^T(y, \lambda) = Iy + \lambda(aa^T).$$

v^k is a vector with dimension $n+1$, which is $[y_1^k, y_2^k, \dots, y_n^k, \lambda^k]$, $e_{n+1}[1, n]$ is a vector composed of first n ones and a zero and α is a constant in $(0, 1)$. We improve dual objective values iteratively by doing this. Then recover primal solution X^k using y^k getting new $\bar{z}^k = L \bullet X^T$. We stop iteration when primal-dual gap $\bar{z}^k - e^T y^k$ is small enough. For details of constructing X^k and algorithms refer to [7].

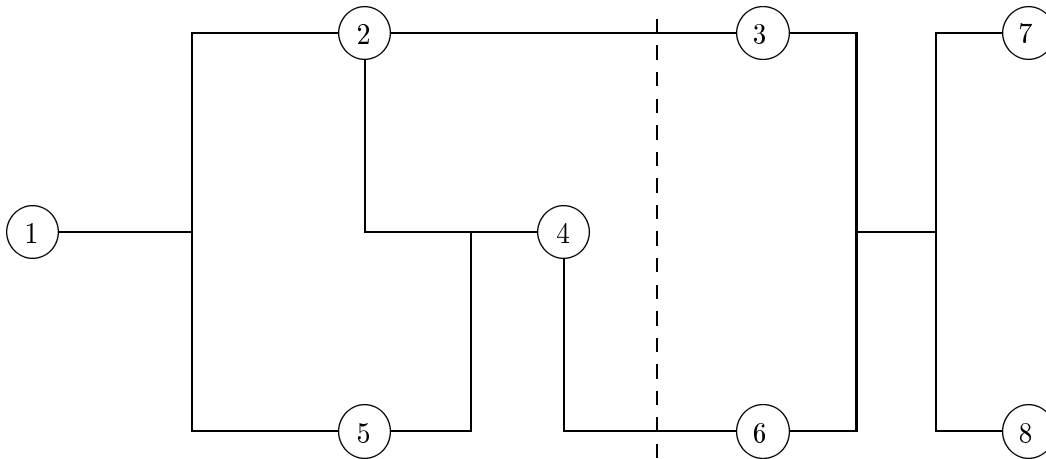
4 Rank reduction

Vector solutions are generated by some rank reduction techniques like; rows of primal matrix, random unit vector method and dominant eigen vector method.(Refer to [7], [6] and [5] for details

about rank reduction.) Each of these vector solutions are composed of -1 and 1 which denotes the 2-way partition of the modules in entire circuit file. When we search for the best solution, original net file is used to count real number of nets exist on the cuts generated.

For example, assuming there are 5 modules a, b, c, d, e and f, if a and b are on one side and rest on the other side, we look for the number of nets that contain one or more from one side and one or more from the other side. If we can find only two nets containing some elements from both sides which are (a,b,c) and (b,d,e), number of nets for this problem will be 2. We look for these cases to count all the nets on the cut. Among many vector solutions, vector solution bears the least nets on the cut is saved as the best solution. This is necessary because vector solution minimize the SDP objective value $L \bullet X$ which is just an approximation of the original problem.

For the example we used, best vector solution is; $\{-1,-1, 1,-1,-1, 1, 1, 1\}$. This will put nodes $\{ 1, 2, 4, 5 \}$ on one side and nodes $\{ 3, 6, 7, 8 \}$ on the other sides cutting our circuit like below;



For this cut, cut size will be 2, which is optimal, and sums areas on both sides will be the same with 4.

5 Test result

Table on the next page compares the dsdp with other heuristic methods. This test result is obtained from [4] written by Shantanu Dutt and Wenyong Deng in 1996. In this table circuits are partitioned into two equal areas on each part. One clear trend is dsdp works better for the bigger problems. We add some explanations about the methods used on the test.

Problem	FM	LA-3	Window	Prop	dsdp	modules	dsdp_time
bm1	55	55	70	54	55	882	1.57
p1	57	55	60	59	57	833	0.04
p2	236	183	258	154	176	3014	2.98
s13207	92	89	n/a	83	* 78	8772	59.1
s15850	112	75	n/a	73	* 67	10470	88.08
s9234	53	58	n/a	55	* 50	5866	8.95
struct	45	45	n/a	38	* 37	1952	0.5
19ks	142	153	136	120	131	2844	1.57
biomed	83	91	163	88	86	6514	32.11
industry2	428	378	392	254	* 242	12637	423.68
t2	115	105	105	91	107	1663	0.28
t3	72	90	67	58	* 58	1607	0.24
t4	86	88	61	58	61	1515	0.2
t5	97	96	101	82	92	2595	1.41
t6	71	63	70	81	70	1752	0.46

Table 1: Comparison of dsdp result with other methods. 50-50% Balance cut results. (dsdp_time is in hours)

1. FM : 100 runs of ‘Fidducia-Matthetses’ codes.
2. LA-3 : Look ahead method.
3. Window : One of the clustering based partioner.
4. Prop : Probability based approach proposed by [4].
5. dsdp : dual semi-definite programming.

Firstly FM code looks for the immediate improvement using local net list information. Look ahead method had been proposed to improve FM method. This method literally looks ahead of several steps to figure out more than immediate gain when side of a module has changed. Window is a clustering-based technique used by the author of [3] Charles J. Alpert. And Prop is a probability-based method proposed by Shantanu Dutt and Wenyong Deng on [4]. And last method dsdp is our approximated global optimization method using dsdp code.

6 Epilogue

We present this application to propose a new method in circuit partitioning problem. Unlike other heuristic methods our method will search for the global optimal solution of the approximated circuit representation. It appears that global search works a little better than other methods compared in this report.

Even with some limitations, mainly memory and time problem, we discovered a strong possibility that we can apply dual scaling interior point algorithm to this problem. For a circuit with

n modules we need a $n(n+1)/2$ double space triangular half matrix, which is our bottle neck now, with double precision which takes a lot of memory space and factorizing and solving linear equation takes too much time.

Now we are working on the algorithm that does not generate and solve this matrix explicitly and it looks quite successful so far. We hope to step into the next stage of our project soon.

And now we have not implemented allowing some difference between each two sides like about 5% difference. But it is a matter of time that we implement this part into our code. This will allow us better comparison with other existing methods.

References

- [1] C. Berge, *Graphs and Hypergraphs* (American Elsevier, New York, 1976).
- [2] Charles J. Alpert and Andrew Kahng, "Recent directions in netlist partitioning: a survey," Ph.D. Thesis, UCLA computer science department, Los Angeles, CA, 1995.
- [3] Charles J. Alpert and Andrew Kahng, "A general framework for vertex orderings, with applications to circuit clustering," Proc. IEEE/ACM international Conference on CAD, November 1994, pp 63-67.
- [4] Shantanu Dutt and Wenyong Deng, "A probability-based approach to VLSI circuit Partitioning," Design automation conference report, Department of electrical engineering, The University of Minnesota, Minneapolis, MN, 1996.
- [5] Steve John Benson, "Solving large scale combinatorial optimization problems," Ph.D Thesis, Department of Mathematics, The University of Iowa, Iowa City, 1998.
- [6] S. Benson, Y. Ye, and X. Zhang, "Solving sparse, large scale positive semidefinite programs," Department of Management Science, The University of Iowa, Iowa City, 1997, To appear in *SIAM J. of Optimization*.
- [7] S. Benson, Y. Ye, and X. Zhang, "Mixed Linear and Semidefinite Programming for Combinatorial and Quadratic Optimization," Department of Management Science, The University of Iowa, Iowa City, February 13, 1998; revised in April 1999.
- [8] Y. Ye, *Interior Point Algorithms : Theory and Analysis* (Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, New York, 1997).