Interactive Partitioning
System Demonstration, Short

Neal Lesh[1], Joe Marks[1], and Maurizio Patrignani[2]

[1] MERL — A Mitsubishi Electric Research Laboratory, Cambridge, MA 02139
{lesh,marks}@merl.com
[2] Dip. di Informatica e Automazione, Università di Roma Tre, Rome, Italy
patrigna@dia.uniroma3.it

Abstract. Partitioning is often used to support better graph drawing; in this paper, we describe an interactive system in which graph drawing is used to support better partitioning. In our system the user is presented with a drawing of a current network partitioning, and is responsible for choosing appropriate optimization procedures and for focusing their application on portions of the network. Our pilot experiments show that our network drawings succeed in conveying some of the information needed by the human operator to steer the computation effectively, and suggest that interactive, human-guided search may be a useful alternative to fully automatic methods for network and graph partitioning.

1 Introduction

Interactive, semi-automatic systems have been developed for several conventional optimization tasks, such as routing, scheduling, and layout; interactive systems have also been developed for unconventional optimization tasks such as geometric design and data analysis. One motivation to involve people in an optimization process is to combine a computer's fast processing speed with a human's superior ability in such areas as visual perception, learning from experience, and strategic assessment.

Bringing a human into the loop requires the development of operators with which a human user can steer or focus the optimization process, and visualizations that allow the user to apply these operators effectively. In recent work we have investigated a very tight integration of human and computer to solve optimization problems; we call our approach Human-Guided Simple Search (HuGSS), and we have applied it successfully to the problem of capacitated vehicle routing with time windows [3]. We are currently trying to apply the same general approach to the problem of k-way network partitioning, an important NP-hard problem that arises in VLSI design and elsewhere [2].

In this paper we demonstrate the visualization aspects of our interactive network-partitioning system. The networks that we consider are derived from chip circuits modeled as hypergraphs; these hypergraphs have up to 200,000 nodes and as many hyperedges. We need to visualize those aspects of a network that are relevant to partitioning it optimally: the number and strength of

J. Marks (Ed.): GD 2000, LNCS 1984, pp. 31–36, 2001.

hyperedges in the cut set; the size and structure of the partition blocks; and ultimately the partition blocks and network nodes on which to focus the search for best results. Furthermore, the computational time of the whole visualization process should be appropriate for an interactive environment.

2 Problem Description

A *network* or *hypergraph* G is a pair (V, H), where V is a set of *nodes* and H is a set of nonempty subsets of V, called *hyperedges*. *Constrained k-way partitioning* is the NP-hard problem of partitioning the nodes of a network into k disjoint subsets, called *blocks*, so as to minimize the number of hyperedges spanning two or more blocks. The sizes of the blocks are allowed to vary around the value $\frac{n}{k}$, where $n = |V|$; a typical value for the allowed variance is 10%. In current state-of-the-art benchmark problems, $|V|$ and $|H|$ range from 10,000 to 200,000, and k is less than 10 [1]. A recent survey describes the many heuristics that have been devised for this problem [2].

Typical heuristics for difficult combinatorial optimization problems combine some form of gradient descent to find local minima with some strategy for escaping nonoptimal local minima and exploring the solution space. The HuGSS framework for interactive search [3] divides these two subtasks cleanly between human and computer: the computer is responsible only for finding local minima using a simple search method; using visualization and interaction techniques, the human user identifies promising regions of the search space for the computer to explore, and intervenes to help it escape nonoptimal local minima.

The HuGSS approach translates to the following search and focus operators available to the user in our network-partitioning system:

1. Manually edit the current partition by moving nodes between blocks.
2. Launch a refinement heuristic on the whole network, or on a focused subset of the blocks or nodes. Several heuristics are available in our current prototype, ranging from simple hill-climbing methods to more sophisticated techniques.
3. Navigate a history list of previous solutions and revert to an earlier one.

The key to our system is the effective selection and focusing of search heuristics in Step 2 by the human user. For this to be possible, the user must be able to identify fruitful areas of the network on which to concentrate the computer's search, and also be able to choose appropriate heuristics for the subproblems encountered in the focus areas. For example, he might try to identify groups of nodes that are loosely connected to nodes in their current block but strongly connected to nodes in other blocks. Including such nodes in a focused search will allow the computer to spend more effort looking at ways to rearrange just these nodes to achieve a better partition. And depending on the search focus and the nature of the current partition, some search heuristics might be better than others: for example, a tightly focused search might benefit from the application of a more thorough search heuristic than one could afford to use on the whole network.

3 Our Network Visualization

We designed our visualization to help the user make decisions about how to focus the search and to select search heuristics.

Because it is difficult to unambiguously draw a hypergraph, we convert the hypergraph into a graph with weighted edges. We replace each hyperedge of degree n with a clique of $\frac{n(n-1)}{2}$ edges whose weights are $\frac{2}{n(n-1)}$.

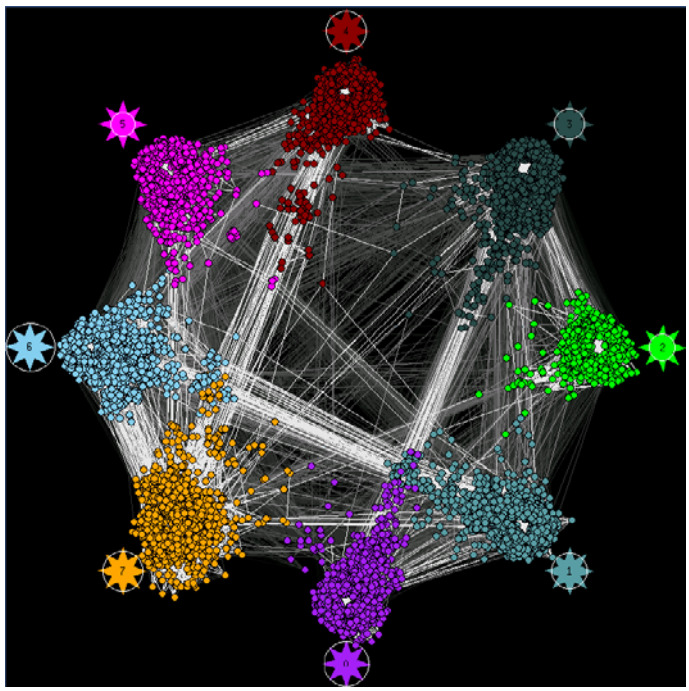

Fig. 1. A visualization of a partition of network `ibm01`.

We use a force-directed approach to determine the position of each node. Each edge in the induced graph is replaced by a spring that opposes stretching or compressing with a force proportional to its weight. Furthermore, a spring is attached between each node and the *hub* of its block. The hubs themselves are fixed and positioned uniformly around a circle. For reasons of efficiency, no other forces are considered, including repulsive forces between nodes.

A drawing produced with this algorithm is shown in Figure 1. Since each node is attracted towards the hub of its block, nodes in the same block tend to overlap near their hub, and only the ones that are strongly linked to other blocks stretch out and are easily distinguishable. These are typically the nodes that interest us the most, since they are those most likely to move usefully between blocks.

In addition to node position, other visual methods are used to convey information about the partition. Membership in a block is indicated by node color.

Edge weights are mapped onto intensity, so that edges of greater weight appear brighter. And at each hub an icon comprising a star-shaped polygon and circle indicates whether the block is near its minimum or maximum allowed size.

The visualization in Figure 1 is useful for comprehending the general structure of a partition and the relative sizes of the blocks. Figure 2 shows the result of a second force-directed system we designed specifically for visualizing pairs of blocks in isolation. We install a spring for each edge between nodes of the block pair and add a rightwards-pulling force to each node in proportion to the weight of its edges to the other blocks. Thus, each node is pulled towards the top or bottom of the screen, depending on its affinity for the top or bottom block and pulled towards the right depending on its affinity for the other blocks. The nodes in the left center of the screen are those that are strongly connected to both blocks and weakly connected to the other blocks (see Figure 2a); a focused search involving these nodes[1] might have several chances for reducing the cut set between the blocks by exchanging some of these nodes. A better scenario is shown in Figure 2d: it is clear that many nodes in the bottom block would prefer to be in the top block, but it is full—the circle surrounds the block icon in the top-left corner—and cannot accommodate them at the moment. However, if nodes can be moved out of the top block without incurring additional cut-set costs (we envision providing an operator that allows the user to request such an adjustment), the bottom-block nodes might then be included with some likely savings in cut-set cost. Finally, Figures 2b and 2c show less promising cases: there are few nodes loosely connected to the other blocks that are also strongly connected to both blocks in the pair; moving nodes between these blocks is less likely to reduce the cut-set cost.

4 Preliminary Experiments

We ran some initial experiments to measure the effectiveness of our visualization in enabling users to identify promising pairs of blocks upon which to focus a simple refinement algorithm. Although these "selection-only" experiments are only a crude test –we anticipate focusing searches at a finer granularity in the finished system–it served to validate that our visualizations contain useful cues.

In each experiment, a group of two or three users (the authors of the system) were presented with a partition of a large network. Using only the visualization capabilities of our system (i.e., they could not launch any optimization algorithms or move any nodes between blocks manually), their task was to select and rank the most promising pairs of blocks for subsequent focused searches.

For our experiments we obtained 8-way partitions of the `ibm04`,..., `ibm08` networks from the benchmark set in [1] by running the Sanchis refinement algorithm ([5]) on the best solution produced by 200 runs of the `hMeTis` system [4], one of the most widely used network-partitioning systems. We computed a score

[1] A more useful search might focus on these nodes and their close neighbors in the hypergraph structure. Our interface allows the user to expand the current set of focused nodes to include all their neighbors.

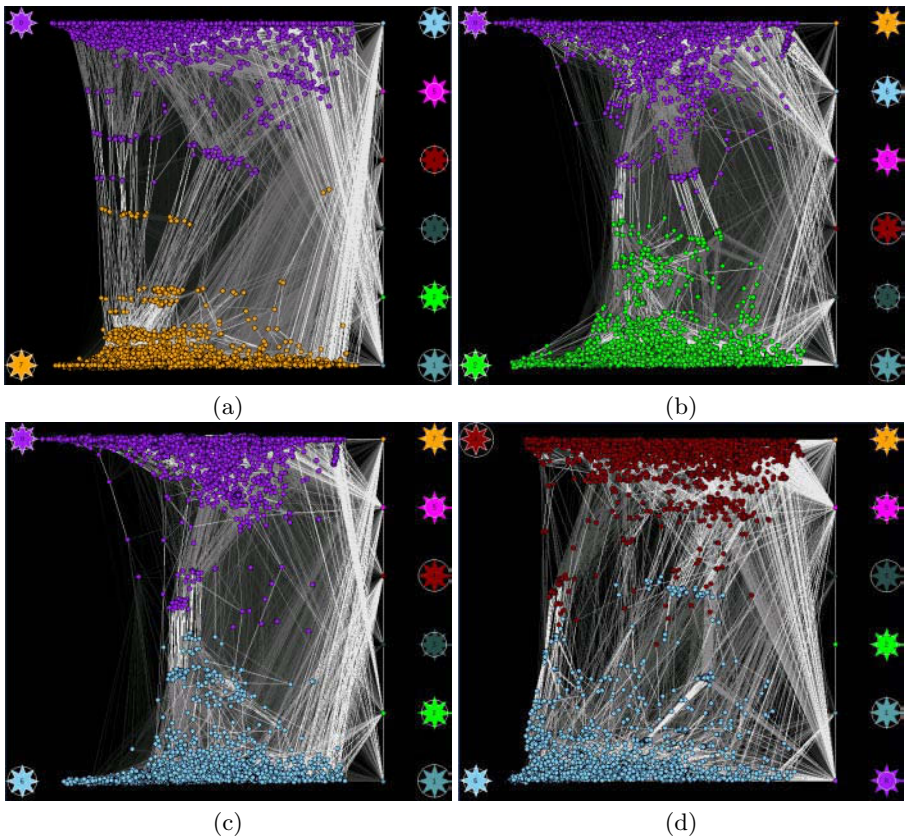

Fig. 2. Visualizing pairs of blocks

for each pair of blocks by running the `hMeTis` algorithm on just those blocks 20 times and then launching the Sanchis algorithm on the best partition found. To compute the score for each pair we repeated this process 10 times and averaged the results.

As shown by the solid line in Figure 3, our visualizations easily allow people to select the most promising pairs of blocks. For $n = 1$ to 14, the graph shows the average sum of the scores of the first n selected pairs divided by the total sum of the scores of all the pairs. For example, the scores of the first three pairs selected were, on average, about 50% of the sum of the scores of all 28 possible pairs. This shows that the user selections were far better than the expected value of random selections, as indicated by the dotted line in the chart.

However, the relatively simple heuristic of ranking the pairs of blocks by the total weight of the edges between them also produced excellent results, just slightly worse than that achieved by human selection. Although the difference is small (after three selections human selection is 8% better than the size heuristic on average), it is not insignificant: given the current state of the art and the com-

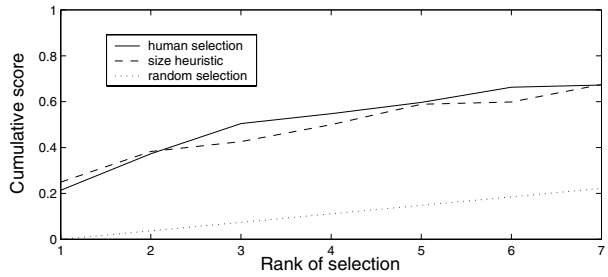

Fig. 3. Experimental results

mercial significance of VLSI design, very small improvements in the performance of partitioning systems are both hard to achieve and very useful. Furthermore, when we correctly selected block pairs out of order relative to the size heuristic, we usually did so based on our observation of complex patterns in the visualizations, as indicated by the discussion of Figure 2. We conjecture that future work will allow us to take greater advantage of these observations by employing search and focus operators that are more sophisticated than simply applying the same refinement algorithm to all the nodes in a block pair.

5 Conclusions and Future Work

Traditionally graph drawing has been used to support analysis-oriented tasks involving databases, software engineering, and computer and communication networks. In this paper we demonstrate the potential use of graph drawing to support an optimization-oriented task, network partitioning. Although at an early stage in our research, we have implemented a fully working system and established that graph drawing can supply useful information about large network partitions to a human user. In future work we plan to refine our set of search and focus operators and devise experiments to see if human users can use our system to improve state-of-the-art solutions to network-partition problems.

References

1. C. J. Alpert. The ISPD98 circuit benchmark suite. In *Proc. of the Intl. Symposium of Physical Design (ISPD'98)*, pages 80–85, 1998.
2. C. J. Alpert and A. B. Kahng. Recent directions in netlist partitioning: A survey. *Integration: The VLSI Journal*, 19:1–81, 1995.
3. D. Anderson, E. Anderson, N. Lesh, J. Marks, B. Mirtich, D. Ratajczak, and K. Ryall. Human-guided simple search. To Appear in Proc. of AAAI 2000. Also `http://www.merl.com/reports/TR2000-16/index.html`.
4. G. Karypis and V. Kumar. Multilevel k-way hypergraph partitioning. In *Proc. of the 36th Design Automation Conference*, pages 343–348, 1999.
5. L. A. Sanchis. Multiple-way network partitioning. *IEEE Trans. on Comp.*, 38:62–81, 1989.