[18] L. Hagen and A. B. Kahng, "A New Approach to Effective Circuit Clustering", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1992, pp. 422-427.

[19] G. Kortsarz and D. Peleg, "Generating Sparse 2-spanners", *Third Scandinavian Workshop Proceedings*, 1992, pp. 73-82.

[20] B. Krishnamurthy, "An Improved Min-Cut Algorithm for Partitioning VLSI Networks", *IEEE Trans. Computers*, vol. C-33, May 1984, pp. 438-446.

[21] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids* Holt, Rinehart and Wiston, New York, 1976.

[22] D. W. Matula and F. Shahrokhi, "Sparsest Cuts and Bottlenecks in Graphs", *Discrete Applied Mathematics*, vol. 27, 1990, pp. 113-123.

[23] J.-C. Picard, "Maximal Closure of a Graph and Application to Combinatorial Problems", *Management Science*, Vol. 22, 1976, pp. 1268-1272.

[24] J.-C. Picard and M. Queyranne, "A Network Flow Solution to Some Nonlinear 0-1 Programming Programs, with Applications to Graph Theory", *Networks*, Vol. 12, 1982, pp. 141-159.

[25] J. M. W. Rhys, "A Selection Problem of Shared Fixed Costs and Network Flows", *Management Science*, Vol. 17, 1970, pp. 200-207.

[26] L. A. Sanchis, "Multiple-Way Network Partitioning", *IEEE Trans. Computers*, vol. 38, no. 1, 1989, pp. 62-81.

[27] H. Saran and V. V. Vazirani, "Finding k-cuts within Twice the Optimal", Proc. 32nd Annual Symp. on the Foundations of Computer Science, 1991, pp. 743-751.

[28] Y. C. Wei and C. K. Cheng, "Ratio Cut Partitioning for Hierarchical Designs", *IEEE Trans. Computer-Aided Design*, vol. 10, July 1992, pp. 911-921.

[29] C.-W. Yeh, C. K. Cheng, and T. T. Lin, "A Probabilistic Multicommodity-Flow Solution to Circuit Clustering Problem", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1992, pp. 428-431.

[5] P. K. Chan, M. D. F. Schlag, and J. Y. Zien, "Spectral K-Way Ratio-Cut Partitioning and Clustering", *Proc. 30th ACM/IEEE Design Automation Conference*, 1993, pp. 749-754.

[6] J. Cong, L. Hagen and A. B. Kahng, "Random Walks for Circuit Clustering", *Proc. 4th IEEE Intl. ASIC Conf.*, Rochester, September 1991, pp. 14.2.1 - 14.2.4.

[7] J. Cong, L. Hagen, and A. B. Kahng, "Net Partitioning Yield Better Module Partitions", *Proc. 29th ACM/IEEE Design Automation Conference*, 1992, pp. 47-52.

[8] J. Cong and M. Smith, "A Parallel Bottom-up Clustering Algorithm with Applications to Circuit Partitioning in VLSI Design", 30th *ACM/IEEE Design Automation Conference*, 1993, pp. 755-760.

[9] T. Feo, O. Goldschmidt, and M. Khellaf, "One-Half Approximation Algorithms for the k-Partition Problem", *Operations Research* Vol. 40, Supp. No. 1, 1992.

[10] C. M. Fiduccia and R. M. Mattheyses, "A Linear Time Heuristic for Improving Network Partitioning", *Proc. 19th ACM/IEEE Design Automation Conference*, 1982, pp. 175-181.

[11] G. Gallo, M. D. Grigoriadis, R. E. Tarjan, "A Fast Parametric Maximum Flow Algorithm and Application", *SIAM Journal on Computing*, vol. 18, Feb. 1989, pp. 30-55.

[12] J. Garbers, H. J. Promel, and A. Steger, "Finding Clusters in VLSI Circuits", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1990, pp. 520-523.

[13] M. R. Garey and D. S. Johnson, *Computers and Intractability : A Giude to the Theory of NP-Completeness*, W.H. Freeman and Company, New York, 1979.

[14] A. V. Goldberg, "Finding a Maximum Density Subgraph", *Technical Report No. UCB/CSD 84/171*, May 1984.

[15] R. Gomory and T. C. Hu, "Multi-terminal network flows", *J. SIAM*, vol. 9, 1961, pp. 551-570.

[16] L. Hagen, *personal communication*, April 1994.

[17] L. Hagen and A. B. Kahng, "New Spectral Methods for Ratio Cut Partitioning and Clustering", *IEEE Trans. Computer- Aided Design*, vol. 11, no. 9, Sep. 1992, pp. 1074-1085.

- **Max-Density Subgraph with Prescribed Node** problem : Given a hypergraph $H(V, E)$ and a prescribed node p, find the maximum density subgraph which contains node p.

  This can be solved using the $0-1$ fractional programming technique in [24] by assigning the variable corresponding to p to 1, and transforming the resulting fractional expression to a series of flow computations.

- **Max-Density Subgraph with Excluded Node** problem : Given a hypergraph $H(V, E)$ and a prescribed node p, find the maximum density subgraph which does not contain node p.

  This problem can be solved in the same time complexity as the MDS problem. We can remove node p and all edges incident from p, and solve the MDS problem in the remaining graph.

It is reasonable to suspect that the Prescribed and Excluded Node variants are useful for certain partitioning and clustering purposes.

7 Acknowledgments

We are very grateful to Lars Hagen for his valuable comments, and Laura Sanchis for providing k-way FM code [26]. Also thanks to L. T. Liu for valuable discussion.

References

[1] C. J. Alpert and A. B. Kahng, "Geometric Embeddings for Faster and Better Multi-Way Netlist Partitioning", *Proc. 30th ACM/IEEE Design Automation Conference*, 1993, pp. 743-748.

[2] C. J. Alpert and A. B. Kahng, "Multi-Way Partitioning Via Spacefilling Curves and Dynamic Programming", *Proc. 31th ACM/IEEE Design Automation Conference*, 1994, to appear.

[3] M. L. Balinski, "On a Selection Problem", *Management Science*, Vol. 17, 1970, pp. 230-231.

[4] T. N. Bui and B. R. Moon, "A Fast and Stable Hybrid Genetic Algorithm for the Ratio-Cut Partitioning Problem on Hypergraph", *Proc. 31th ACM/IEEE Design Automation Conference*, 1994, to appear.

#clusters	Benchmarks	k-MSD-FM		k-RCut-FM	
		Density Sum	RCut	Density Sum	RCut
2	p1	2.31	73.80	2.01	36.90
	p2	1.96	11.46	1.83	11.89
	t2	15.49	1264.30	1.83	29.07
	t3	2.54	159.52	1.77	30.82
	t4	15.55	1487.11	1.97	25.85
	t5	15.51	733.31	1.90	17.05
	t6	1.90	28.94	1.70	20.14
3	p1	3.33	63.88	3.19	32.11
	p2	2.87	21.68	2.51	17.74
	t2	29.95	1264.25	2.73	22.42
	t3	3.69	61.39	2.73	22.42
	t4	27.01	1454.00	3.24	19.65
	t5	30.01	761.64	2.76	17.65
	t6	2.90	74.82	2.52	19.88
4	p1	4.51	54.45	3.60	66.01
	p2	3.95	62.17	3.36	16.88
	t2	31.07	856.50	3.32	34.90
	t3	7.11	558.99	3.42	28.79
	t4	28.31	974.08	3.72	29.33
	t5	31.30	519.86	3.60	19.26
	t6	3.82	38.54	3.35	20.36
8	p1	8.68	113.35	7.22	58.56
	p2	7.25	35.11	6.12	23.68
	t2	35.17	394.75	6.54	38.89
	t3	12.07	295.28	6.54	34.95
	t4	32.56	462.10	6.94	38.55
	t5	35.72	246.93	7.01	21.55
	t6	7.56	36.74	6.38	28.25
10	p1	10.16	135.37	8.66	69.36
	p2	9.25	37.23	7.69	24.20
	t2	37.24	319.92	7.65	44.69
	t3	13.57	232.62	8.21	35.01
	t4	34.72	380.93	8.73	37.93
	t5	37.58	199.17	8.59	22.03
	t6	9.19	43.35	7.72	30.95
15	p1	15.36	119.10	12.37	80.44
	p2	13.57	44.59	11.06	25.35
	t2	41.74	228.75	11.32	45.13
	t3	19.06	172.46	12.65	34.84
	t4	40.07	274.91	12.83	39.25
	t5	42.46	146.57	12.81	23.05
	t6	13.52	57.29	11.06	34.15
20	p1	19.95	133.52	16.04	83.29
	p2	18.30	44.64	13.65	31.84
	t2	46.17	180.43	14.79	48.35
	t3	23.78	146.17	15.52	40.51
	t4	45.91	229.43	17.33	38.50
	t5	48.98	130.64	16.74	25.37
	t6	18.20	55.40	15.12	37.36

Table 5: Comparison of max sum of densities metric and ratio cut metric

#clusters	Benchmarks	Upper Bound	MDS-Peeling + FM	% of optimum
8	p1	10.00	8.63	86.30%
	p2	8.06	7.16	88.83%
10	p1	12.50	10.41	83.28%
	p2	10.08	8.73	86.61%
15	p1	18.75	14.18	75.62%
	p2	15.11	12.25	81.07%
20	p1	25.00	18.80	75.20%
	p2	20.15	15.25	75.68%

Table 4: Comparison of MDS-Peeling + FM and upper bound of k-MSD with large k. 20-way MDS-Peeling takes 110 sec for p1 and 768 sec for p2 in SUN SPARC 10

Finally, in order to compare max sum of densities and ratio cut objectives, we also implemented k-way ratio cut FM (k-RCut-FM), which updates the gains based on the ratio cut metric. We ran both k-MSD-FM and k-RCut-FM for 10 runs each on the MCNC benchmarks. Figure 5 shows the best result among the 10 runs: k-MSD-FM always has better sum-of-densities results than k-RCut-FM, and can even obtain a better ratio cut value than k-RCut-FM in two cases (p2/2-way and p1/4-way).

6 Conclusions

We have introduced a new density-based metric for partitioning and clustering. This metric not only gives an internal view of cluster structure but, surprisingly, also captures *cut* objectives. We have given an optimum Provisioning-based flow solution for the MDS problem, as well as MDS-Peeling and k-MSD-FM heuristics for the k-MSD problem. We have also shown that density-based partitioning can be used to solve the 2-way ratio cut partitioning problem, giving results that are competitive with the best previously published results for 2-way ratio cut partitioning. We may address some related density-based formulations as follows:

- **Bounded Size Maximum Density Subgraph (BMDS)** problem : Given a hypergraph $H(V, E)$ and an integer B, find the maximum density subgraph of H with the size $\leq B$.

  While MDS was polynomial time solvable, BMDS is NP-Complete because we can solve the max clique problem by trying all possible values of the cluster size bound.

Benchmarks	#modules/#nets	IG-Match [7]	DP-RP [2]	NCNC	NCNC + FM	Parameters	CPU (sec)
bm1	882/902	5.53*	5.53*	5.53	5.53*	(0.9 0.99 0.01 0.6)	12.9
19ks	2844/3282	5.96	5.44	5.59	5.04*	(0.6 0.95 0.01 0.8)	90.2
Prim1	833/902	13.39*	3.50	14.27	13.39*	(0.9 0.95 0.01 0.8)	12.0
Prim2	3014/3029	4.58*	5.09	4.64	4.58*	(0.7 0.95 0.01 0.7)	89.0
Test02	1663/1720	12.40	8.07	8.30	7.96*	(0.6 0.95 0.025 0.7)	32.8
Test03	1607/1618	8.98*	10.20	12.65	11.35	(0.9 0.95 0.01 0.8)	25.8
Test04	1515/1658	5.70*	5.85	8.93	7.67	(0.6 0.9 0.025 0.6)	33.0
Test05	2595/2750	3.06*	3.15	6.86	5.84	(0.6 0.9 0.025 0.6)	72.8
Test06	1752/1641	7.48*	9.20	9.31	8.14	(0.7 0.95 0.01 0.7)	35.8

Table 2: Comparison of NCNC and two cluster algorithms in [2] and [7] for MCNC benchmarks. Values in Parameters column show the parameters of $(\alpha, \beta, \gamma, \delta)$ giving the best ratio-cut result. The last column shows the running time in SUN SPARC 10 for a given parameter tuple. All ratio-cut results are reported as a multiple of 10^{-5}. Values marked by an asterisk are the best results among three algorithms for that benchmark. IG-Match is not improvable by FM [16]. We are in process of obtaining the code from the authors of [2] and will run DP-RP + FM for fair comparison.

Benchmarks	Upper Bound	MDS-Peeling + FM	% of optimum
p1	2.50	2.31	92.40%
p2	2.02	1.97	97.52%
t2	29.00	15.49	53.41%
t3	7.00	4.47	63.86%
t4	29.00	15.55	53.62%
t5	29.00	15.51	53.48%
t6	1.99	1.89	94.97%

Table 3: Comparison of MDS-Peeling + FM and the upper bound of 2-MSD

solution quality. We tested this algorithm on the same MCNC benchmarks for 2-way partitioning; Table 3 gives comparison against the upper bound for 2-MSD. (The upper bound of 2-MSD is twice of the densest subgraph density.) For three benchmarks p1, p2, and t6, we obtained over 90% of optimum. Since t2, t4, and t5 contain some very small clusters with very high density, the upper bound becomes very loose: we obtained about 53% of optimum in these cases.

We also tested MDS-Peeling + FM on the Primary1 and Primary2 benchmarks for k-way clustering with large k, again comparing against the upper bound for the k-MSD solution in Table 4. (The upper bound of k-MSD is k times of the densest subgraph density.) Although MDS-Peeling has a worst-case performance ratio of k for k-MSD, the results show that it is actually very close to optimum.

α	β	γ	δ
0.9	0.99	0.01	0.9
0.8	0.95	0.025	0.8
0.7	0.9		0.7
0.6	0.8		0.6

Table 1: The values of the parameters used in Hyper-MDS algorithm.

Third, we note that a node with high degree will potentially involve more cuts. Thus we would like to assign a large cost to a high-degree node. For a parameter $\delta \leq 1$, the node cost is defined as

$$1 + \sum_{k=1}^{d-1} \delta^k$$

where d is the degree of the given node, and δ is the control parameter for the node degree. After assigning the net credit and the node cost, we can use flow to optimally solve the problem of find a cluster with maximum ratio of the sum of net credits to the sum of node costs. This again involves a series of Provisioning instances, and yield our net credit node cost (**NCNC**) algorithm.

5 Experimental Results

All of our algorithms are implemented in C on a single-processor Sun SPARC-10 workstation. Our first experiment was to find heuristic minimum ratio cuts using the NCNC algorithm. We tested various parameters $(\alpha, \beta, \gamma, \delta)$ of the NCNC algorithm as shown in Table 1 for MCNC benchmarks, i.e., a total of 128 combinations. Since the Hyper-MDS algorithm is very fast, we still obtain our ratio cut solutions in reasonable time. Furthermore, we obtain results that are competitive with the best two-way ratio-cut results [2, 7] in the literature. Table 2 shows the comparison of NCNC against IG-Match [7] and DP-RP [2] which are the best ratio cut algorithms in the literature. The results in the NCNC column are obtained from the NCNC algorithm. We also used k-MSD-FM to improve our solution quality in NCNC, yielding the results in the NCNC + FM column. For the 9 benchmarks studied, NCNC obtains 5 best results while IG-Match obtains 7 best results and DP-RP obtains 1 best result.

Next, we combined the two algorithms described in the previous section, i.e., we studied MDS-Peeling + FM which obtains a solution from MDS-Peeling heuristic, and then uses k-MSD-FM to improve the

a high-degree node will potentially involve more cuts.

In order to avoid any bias toward the nets and nodes with large degree, we need to adjust the "*net credit*" for each net and the "*node cost*" for each node. There are many ways to assign net credit and node cost. We have explored the following method, with many other (hopefully more simple) strategies also likely to yield reasonable results.

In the experiments reported below, an initial net credit is assigned to each net according to the net size. While we could simply assign this credit proportional to the net size, this will give large nets very high credit and the resulting partitioning will try to completely contain large nets in the clusters. Thus, to assign net credit for an s-pin net, we use a sub-linear function $f(\alpha, s) = 1 + \alpha + \alpha^2 + \ldots + \alpha^{s-1}$ with $0 \leq \alpha \leq 1$, such that the value of $f(\alpha, s)$ will not grow as fast as the size s. When $\alpha = 1$, $f(\alpha, s)$ is proportional to s; when $\alpha = 0$, $f(\alpha, s) = 1 \ \ \forall s$. We call α the "*control parameter*" for s. Normalizing the net credit of a two-pin net to 1 for a given control parameter $\alpha \leq 1$, the initial net credit is defined as

$$\frac{1}{2} \cdot (1 + \sum_{k=1}^{s-1} \alpha^k)$$

where s is the net size. The two key points are that (i) this function assigns progressively less credit for each additional pin in a given net; and (ii) the geometric series also sets a credit threshold over all net sizes, i.e., no net can be more than $\frac{1}{1-\alpha}$ as important as any other net. Normally, we assign α close to 1. When $\alpha = 1$, a two-pin net has credit 1.

A second observation is that a net with high degree has potential to get more cuts. Let $Nets(i) = \{e \in E \mid v_i \in e\}$ be the set of nets incident on v_i and let $Adj(i) = \{v_j \in e \mid e \in Nets(i)\}$ be the set of neighbor nodes of v_i. Thus a *net penalty* is assigned to each net according to the number of nodes which are the neighbors of the given net e, i.e. $Neighbor(e) = \{v_j \in Adj(i), v_j \notin e \mid v_i \in e\}$. Given two parameters β and γ, the net penalty of a net e is defined as

$$(1 + \sum_{k=1}^{m-1} \beta^k) \cdot \gamma$$

where $m = |Neighbor(e)|$; β is the control parameter for the size of $Neighbor(e)$, and γ is the unit penalty. Normally, we assign β close to 1 and assign a small value to γ. The actual net credit is therefore the difference of the initial net credit and the net penalty.

to another cluster so as to maximize the *sum of densities*. A node is *locked* after it has been moved during the current pass. The best partitioning in terms of the sum of densities metric encountered during the pass is then returned. Additional passes are then performed until no further improvements can be obtained. Following the standard terminology, we define the gain g_{ij} to be the increment in the sum of densities when node i is moved to cluster j. When node i is moved, the gains of all nodes sharing nets with nodes in cluster j must be updated. We use k heaps to maintain the gains: since only a constant number of nodes need to be updated during each move, the updating takes $O(k \log n)$ time and the time complexity of each pass is $O(kn \log n)$. Note that this is slightly more expensive than min-cut k-way FM (for min-cut k-way FM, the gain of each node is an integer; "buckets" can thus store the gains and return the node with gain in $O(k)$ time).

4 A Density-Based Partitioning Algorithm for Ratio Cut

While we have seemingly assumed that a good partitioning has high-density clusters, it turns out that we can use the density-based approach to solve a standard partitioning formulation: this section presents a variation of our Hyper-MDS algorithm to solve the 2-way ratio cut problem. There are two reasons that this is interesting. First, the weakness of spectral ratio cut methods [1, 2, 5, 7, 18] is that they have to transform a hypergraph to a graph using clique model. The hypergraph-based Hyper-MDS algorithm avoids any ad hoc hypergraph-to-graph transformation. Second, we demonstrate that "density can capture cut", while the converse seems difficult.

Recall that the Hyper-MDS algorithm will return the densest subgraph of a hypergraph where each net has "credit" $= 1$, and each node has "cost" $= 1$. This has the following weakness as in terms of the ratio cut metric:

- the method concentrates on collecting many smaller nets into the cluster, and ignores resulting growth of the cluster boundary (i.e. nets cut).
- the method does not distinguish between nets with high degree and nets with low degree; however, a high-degree net will potentially involve more cuts.
- the method does not distinguish between nodes with high degree and nodes with low degree; again,

nodes in the graph have been peeled off. A basic algorithm template is shown in Figure 5. This algorithm implicitly gives an upper bound for the k-MSD solution, and has good performance in practice.

MDS-Peeling Algorithm
Input : Hypergraph $H(V, E)$, and k Output: k clusters

i = 0;
while (E is not empty) **and** ($i < k$) **do**
Find the maximum density subhypergraph $U(V', E')$ of H
Let E'' be the hyperedge cut set of $(V', V - V')$
$E = E - E' - E''$;
$Cluster_i = V'$
$V = V - V'$
$i = i + 1$;
if E is not empty **then**
$cluster_k = V$

Figure 5: The template for MDS-Peeling Algorithm

Observation 1 *Given a hypergraph $H(V, E)$, let D be the density of the maximum density subgraph of H found by MDS-Peeling. Then an upper bound on the density sum S of the optimal k-MSD solution is kD, and this bound is tight.*

Proof : The maximum-density cluster of H has density D, so the sum of densities of k clusters is at most kD. If there are k disjoint clusters each with density D, this upper bound is tight. □

Observation 2 *MDS-Peeling has performance ratio 2 for the 2-MSD problem.*

Proof : Let D be the density of the maximum-density cluster of H, and let D' be the density of the remaining cluster. The performance ratio of the MDS-Peeling is $\frac{D+D'}{2D} = \frac{1}{2} + \frac{D'}{2D} \geq \frac{1}{2}$. □

In general, MDS-Peeling has a tight performance ratio of k for the k-MSD problem. While this may not seem very good, the performance is acceptable in practice (and there are very few "good" bounds are available for existing partitioning and clustering formulations).

Finally, another heuristic for the k-MSD problem is to simply apply the k-way FM ideas from [26]; this yields our k-MSD-FM heuristic. Given a partitioning, we move one node at a time from one cluster

it is sufficient to show that the NP-complete max-cut problem [13] reduces to the Min-Sum-2-Densities problem.

Given a graph $G(V, E)$, construct a graph $G^*(V^*, E^*)$ with $2n$ nodes that is composed of G and a separate isomorphic copy $G'(V', E')$ of G. Assign unit capacity to every edge in $E \cup E'$. Add one edge in G^* between each $v_i \in V$ and the corresponding $v_{i'} \in V'$, and assign capacity M to each of the n such edges. For sufficiently large M, a Min-Sum-2-Densities partitioning $(X, \bar{X})$ in G^* must have $|X| = |\bar{X}|$ with either $v_i \in X$ and $v_{i'} \in \bar{X}$, or $v_i \in \bar{X}$ and $v_{i'} \in X$, since only such a partitioning can have Min-Sum-2-Densities value as small as $\frac{2|E|}{n}$. A partitioning $(X, \bar{X})$ that solves Min-Sum-2-Densities in G^* will have sum of densities $\frac{2|E|-2k}{n}$, where k is the value of a maximum cut of G, and $(X \cap V, \bar{X} \cap V)$ will provide a maximum cut in the original graph G. (See Figure 4.)

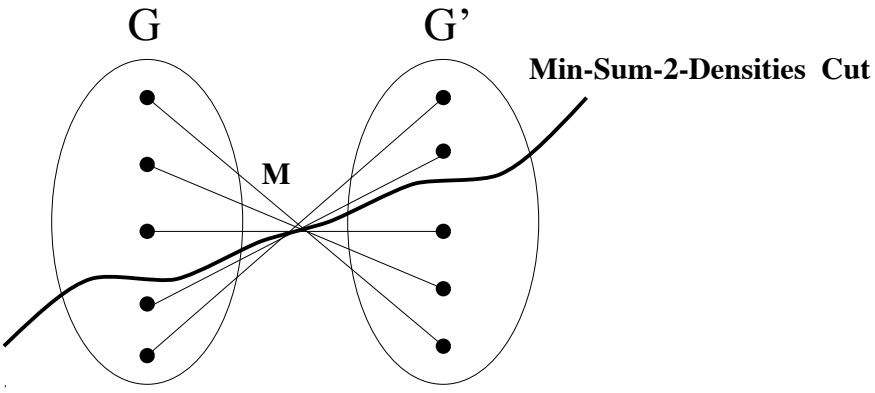

Figure 4: The construction of G^*

To prove that the k-MSD problem is NP-hard for $k \geq 3$, for any given graph G we construct a graph G^* containing G and $k-2$ clusters with very high density D. For sufficiently large D, the Max-Sum-k-Densities partition of G^* must contain the $k-2$ dense clusters and two clusters from G. We thus obtain a reduction from the 2-MSD problem.[2] □

We now develop a greedy heuristic called *MDS-Peeling*, which interactively applies *hyper-MDS* to solve the k-MSD problem. The idea is to peel off a maximum-density cluster $k-1$ times, or until all

[2]Another way to prove that k-MSD is NP-hard for $k \geq 3$ is to reduce from the problem of partitioning a graph into k cliques. This problem is NP-complete for all fixed $k \geq 3$ [13]. If the solution of the k-MSD problem is $\frac{N-k}{2}$, we can partition the graph into k cliques. Note that partitioning a graph into 2 cliques is polynomial-time solvable: simply check whether the complement graph is 2-colorable.

determine the optimal solution.[1] Figure 3 shows the transformation from a hypergraph to a flow network instance.

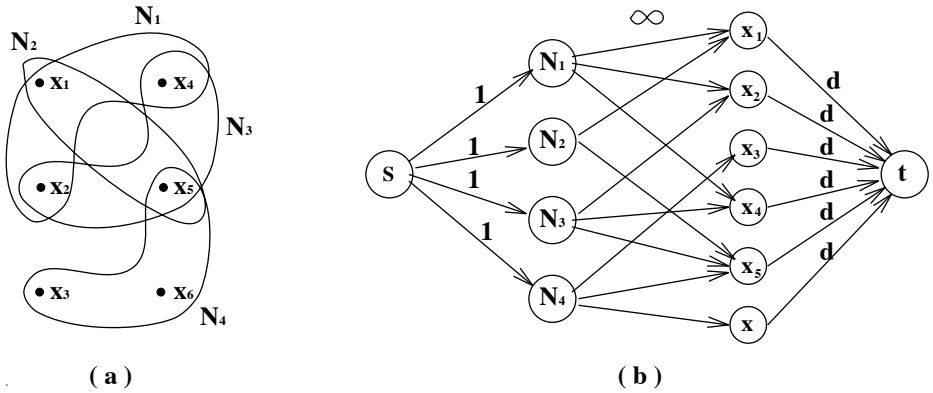

Figure 3: An example of a hypergraph (a) and its corresponding flow network instance (b). The hypergraph contains 6 nodes and 4 signal nets $N_1 = \{x_1, x_2, x_4\}$, $N_2 = \{x_1, x_5\}$, $N_3 = \{x_2, x_4, x_5\}$, and $N_4 = \{x_3, x_5, x_6\}$. We assign a "*guess*" density d to all edges incident from the nodes in the right hand side. When $d = \frac{3}{4}$, the edge set $\{\overrightarrow{sN_4}, \overrightarrow{x_1t}, \overrightarrow{x_2t}, \overrightarrow{x_4t}, \overrightarrow{x_5t}\}$ will be in the minimum edge cut set, thus the maximum density cluster is $\{x_1, x_2, x_4, x_5\}$.

3 The Maximum k-Way Sum of Densities (k-MSD) Problem

In this section, we first show that k-MSD is NP-hard. We then present a greedy heuristic for this problem which has performance ratio of 2 for the 2-MSD problem, and k for the k-MSD problem, but which also performs remarkably closely to optimum in practice.

Theorem 1 *The k-MSD Problem is NP-hard for all fixed $k \geq 2$.*

Proof : We first prove that 2-MSD problem is NP-hard. Note that the problem of maximizing the sum of densities of 2 clusters has equivalent complexity to minimizing the sum of densities of 2 clusters (Min-Sum-2-Densities), since the 2-MSD solution for a given graph G will be the same as the Min-Sum-2-Densities solution in the complement of G. Thus, to confirm that the 2-MSD problem is NP-hard,

[1] As a side note: for any given graph this algorithm yields an upper bound on the size of the maximum clique (or max independent set). If the density of the max-density subgraph is d, then the maximum clique size is $\leq \lfloor 2d \rfloor + 1$.

V is a 0-1 variable x_i where $x_i = 1$ means that node v_i belongs to the maximum density subgraph. This fractional expression can be maximized via a series of max-flow instances, with $O(n)$ flow computations needed in the worst case. Goldberg [14] observed that the smallest possible difference between two distinct density values is $\frac{1}{n(n-1)}$, and developed a different flow model which requires only $O(\log n)$ flow computations to achieve a solution.

In the work which directly led to our approach, Kortsarz and Peleg [19] solved the MDS problem in a graph using the *Provisioning* formulation of Lawler [21].

The Provisioning Problem: Suppose there are n items $x_1, x_2, \ldots, x_n$ to select from, with selection of item x_i incurring cost $c_i > 0$. Suppose there are m sets of items $S_1, S_2, \ldots, S_m$ that are known to confer special benefits. Any given item may be contained in several different sets. If all of the items in set S_j are selected, then a benefit $b_j > 0$ is obtained. The objective is to maximize the net benefit, i.e., the total benefit gained minus the total cost of items selected.

Rhys [25], Balinski [3], and Picard [23] showed that the Provisioning formulation can be transformed into a maximum flow problem in a network. They constructed a bipartite graph where the nodes n_{S_i} on one(say, the left) side represent the sets S_i, and the nodes n_{x_j} on the other(say, the right) side represent the items x_j. A directed edge with infinite capacity is drawn from n_{S_i} to n_{x_j} if S_i contains x_j. The source is linked to each node n_{S_i} by an edge of capacity b_i, and the sink has a link from each node n_{x_j} by an edge of capacity c_j. Edges in the minimum cut set which are incident from nodes on the right side will correspond to the items selected in the optimum Provisioning solution.

We may extend the algorithm in [19] to solve the MDS problem for a hypergraph in polynomial time; this yields our *Hyper-MDS* algorithm for the MDS formulation. The reduction from the MDS problem to a series of Provisioning problem instances is as follows. Let each node be an item, and let each hyperedge in the netlist be a set containing all items corresponding to its nodes. Suppose we want to check whether there exists a subhypergraph with density d or higher, i.e., we seek a node subset U with $\frac{|E(U)|}{|U|} \geq d$ (here, $|E(U)|$ denotes the number of hyperedges completely contained in node set U). This may be restated as $|E(U)| - d|U| \geq 0$. But this is simply the Provisioning formulation with the cost of each item being d and the benefit of each set being 1. Since the difference between two distinct densities is at least $\frac{1}{n(n-1)}$, we can use binary search to guess the maximum density. Only $O(\log n)$ flow computations are needed to

- **Maximum k-Way Sum of Densities (k-MSD)** Problem: Given a hypergraph $H(V, E)$, partition V into k clusters, $P_1, P_2, \ldots, P_k$, such that the sum of the cluster densities is maximized.

Certainly, there are similarities between the sum of densities and other cut-based objectives. If the given graph is complete graph, chain, or cycle, maximizing the sum of densities will generate the same bipartition as minimizing the ratio cut. If we restrict the cluster sizes to be all identical, then a maximum sum of densities solution will also be a minimum cut solution. It should also be pointed out that the density criterion can be pathologically blind to global structure in the netlist: for example, if there is a tiny component with many edges, it will always be a part of a MDS or k-MSD based circuit decomposition. (This issue arises in our studies of the Test02, Test04 and Test05 benchmark circuits below.)

The remainder of this paper concerns what is implicitly our fourth motivation, namely, that certain maximum-density optimizations turn out to be *optimally* solvable using efficient network flow methods. A variety of heuristics can be devised for both the MDS and k-MSD problems, and for "emulation" of cut-based optimizations. Section 2 gives a survey of the MDS problem, and provides a polynomial time solution. Section 3 shows that the k-MSD problem is NP-hard, and gives a heuristic algorithm with good performance. Section 4 shows the versatility of the density-based approach by perturbing the MDS solution into an effective ratio cut bipartitioning heuristic. Experimental results are summarized in Section 5.

2 The Maximum Density Subgraph Problem

Many researchers have solved the MDS problem for a given *graph* [24, 14]; this is a special case of our hypergraph formulation. The usual transformation is to a series of minimum cut computations, which in turn are accomplished using maximum flow techniques. Picard and Queyranne [24] formulated the MDS problem as a special case of 0-1 fractional programming, i.e.,

$$\text{maximize} \quad \frac{\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}x_i x_j}{\sum_{k=1}^{n} x_k}$$

where a_{ij} are the elements of the node-to-node adjacency matrix of G, and associated with each node $v_i \in$

criterion can lead to a more natural bipartition than the ratio cut objective. One can see that the ratio cut objective somehow overemphasizes the reduction of net cut size, and ignores the internal structure inside the clusters. The optimum sum-of-densities cut will shift as the internal structure of the graph changes, while the optimum ratio cut remains the same.

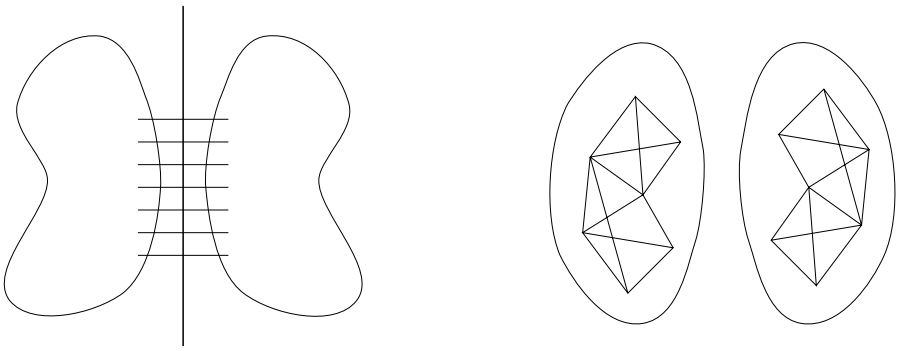

Figure 2: (a) Cut objectives pertain to the cutsize at the cluster boundary, while (b) density objectives pertain to the implied area or wirelength inside the cluster.

A second motivation is suggested by Figure 2: the cut objective has an *external* view of the circuit components, while the density objective has an *internal* view. While the cutsize at the boundary of a component has been used for area and wirelength estimation via the Rent parameter or other devices, it is not clear whether the number of terminals really forces any non-trivial lower bound on the resulting wirelength in the layout. In other words, having many terminals will impact wirelength only in a probabilistic sense, since abutments or other short routes may be possible. Having a high density can imply a less trivial lower bound on required area/wirelength in the layout, and it is possible that a "density spectrum" will provide new means of wirelength estimation in the future, via the optimal network-flow for maximum-density subgraph computation methods that we describe below. A third motivation, which we explore below, is that density can capture *both* cut-based and density-based objectives, which seems beyond the ability of any cut-related formulation.

Two problem formulations are addressed in this paper.

- **Maximum Density Subgraph (MDS)** Problem: Given a hypergraph $H(V, E)$, find a subhypergraph of H with largest density.

example, a cluster with one edge and two nodes (i.e., a 2-clique) is always considered to have high density.

The main theme of our work is that (cut-based) partitioning and (density-based) clustering have no obvious common ground for "middle" values of k, the number of components in the circuit decomposition. Yet, due to the commonality of their applications to hierarchical circuit layout, it is arguable that these approaches *should* "meet in the middle". At present, partitioning is more well-developed than clustering: there are good flow-based, spectral, and iterative methods (some of which even have error bounds), while most clustering objectives are difficult to evaluate (e.g., DS or k-l connectivity), let alone optimize.

In this paper, we propose a new *density*-based metric for k-way circuit decomposition (clustering). Formally, the *density* of a hypergraph $H = (V, E)$ is the ratio of the number of hyperedges to the number of nodes, i.e., $\frac{|E|}{|V|}$. Given a node subset $V' \subset V$, the density of the sub-hypergraph $H' = (V', E')$ induced by V' is the number of hyperedges in E that are *completely* contained in E', divided by the number of nodes in V'.

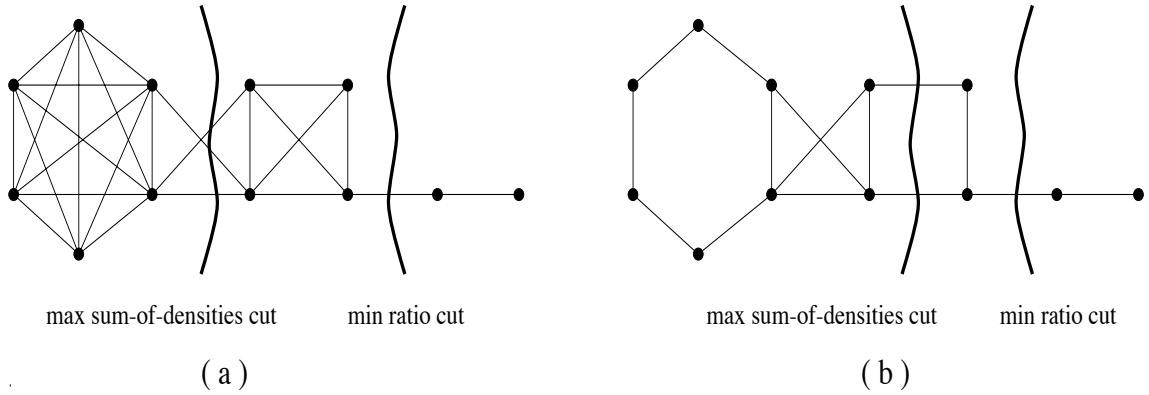

Figure 1: (a) The minimum ratio cut gives an uneven partitioning while the maximum sum-of-densities cut gives a more balanced and natural partitioning. (b) When the circuit structure is changed, the maximum sum of densities cut shifts, while the minimum ratio cut remains the same.

Our first motivation is that the density concept can in fact lead to "natural" circuit decompositions. Thus, our work is similar in spirit to that of Wei and Cheng [28], who in 1989 proposed ratio cut as a "more natural" decomposition objective. There are easy examples for which ratio cut will *not* give a "natural" circuit bipartition, while a simple density-based criterion, such as maximizing the sum of the two partition densities, gives a natural bipartition. The examples of Figure 1 show that the density

First, *minimum cut* objectives minimize the total number of signal nets crossing between different components; variant formulations can optionally impose size constraints on the components. Saran and Vazirani [27] used the Gomory-Hu cut tree [15] to find a k-way minimum cut that is within a factor of 2 of optimal when there are no size constraints on the components. When there are size constraints (e.g., an exact bisection constraint is common for top-down partitioning-placement), Krishnamurthy [20] and Sanchis [26] showed the effectiveness of *level gains*, in conjunction with the Fiduccia-Mattheyses (FM) method [10]. More recent research has focused on the *ratio cut* bipartitioning metric of Wei and Cheng [28], which minimizes $\frac{C(U,V)}{|U| \cdot |V|}$ where $C(U,V)$ is the number of signal nets crossing between the two partitions U and V. This metric is "natural" in that it addresses both the minimum cut objective and the balance requirement for partition sizes. Hagen and Kahng [17] used spectral methods to obtain good ratio cut partitioning solutions. Later, Chan et al. [5] introduced the *scaled cost* metric, which seeks a clustering $P_1, P_2, \ldots, P_k$ that minimizes $\frac{1}{n(k-1)} \sum_{i=1}^{k} \frac{C_i}{|P_i|}$; here, C_i is the number of signal nets crossing the boundary of the partition P_i. This is shown in [5] to be a generalization of the two-way ratio cut objective. Yeh et al. [29] have given an alternate *cluster ratio* generalization of two-way ratio cut to k-way circuit decomposition.

Second, many papers in the literature have approached circuit decomposition from the opposite direction, relying on an intuitive view of a *cluster* as a group of circuit elements with very dense connections among them. The goal of a clustering algorithm is to identify the "good" clusters in a circuit, often via some bottom-up approach. The key issue is how to assess whether a cluster is good. Several clustering metrics have been proposed in the literature. Garbers et al. [12] used *k-l connectivity* to measure the cluster quality. In a graph, two nodes are *k-l connected* if there exist k edge-disjoint paths connecting them such that each path has length $\leq l$. The transitive closure of this relation induces a heuristic clustering. Cong et al. [6] proposed measuring cluster quality by the ratio of *Degree* to *Separation* (i.e., the "DS metric"), where the *Degree* is the average net degree of any node in the cluster, and *Separation* is the average length of the shortest path between two nodes in the cluster. This metric considers the global connectivity information and is somewhat more well-behaved than *k-l* connectivity. However, computing the *Separation* value requires $O(n^3)$ effort when clusters are large. Finally, in what is perhaps the most closely related work to ours, Cong and Smith [8] defined a measure of cluster density as $\frac{|E|}{C(n,2)}$, where $|E|$ is the number of edges and n is the number of nodes in the cluster. This metric is biased to small clusters since $C(n,2)$ increases quadratically in n, while most circuit netlists have constant degree bounds. For

When Clusters Meet Partitions: A New Density Objective for Circuit Decomposition

J.-H. Dennis Huang and Andrew B. Kahng

UCLA Computer Science Department, Los Angeles, CA 90024-1596 USA
jenhsin@cs.ucla.edu, abk@cs.ucla.edu

Abstract

Recent research on multi-way partitioning has focused on the *minimum cut* [20, 26, 27] or generalized *ratio cut* [28, 29, 5] cost metrics. At the same time, clustering research has focused on such objectives as k-l connectivity [12], DS metric [6], or clique-finding [8]. In this paper, we make the basic observation that *cut* objectives in partitioning, and *density* objectives in clustering, are fundamentally incompatible. Moreover, for multi-way decomposition applications (e.g., decomposing a system onto multiple FPGA chips), the two approaches fail to smoothly "meet in the middle".

We present a new measure of multi-way circuit decomposition, based on a *sum of densities* objective. Here, the *density* of a subgraph is the ratio of the number of edges to the number of nodes in the subgraph. In that we feel that this is a natural measure of circuit decomposition (indeed, arguably more natural than ratio cut for a variety of applications), our new objective can perhaps be viewed in the same spirit as the proposal of *ratio cut* by Wei and Cheng in 1989. Several results are given. First, we prove that k-way graph decomposition to maximize the sum of the k component densities is NP-hard; we give a heuristic algorithm with good performance for this problem. We also develop a density-based bipartitioning algorithm, and achieve competitive results with recent works [2, 7] in terms of the ratio cut objective. A key element in this algorithm is an efficient, flow-based method with finds the *optimal* maximum-density subhypergraph in a given netlist hypergraph. Our results show that in some sense, the density measure can be made to "capture" cut-based objectives such as ratio cut. On the other hand, the converse would seem impossible. The paper concludes with a brief list of future directions.

1 Introduction

In VLSI applications, both partitioning and clustering have been used to reveal "natural" circuit decompositions, i.e., divisions of the n nodes of a (hyper)graph into k disjoint subsets. Two objectives have been used.