COMBINATORICA 7 (2) (1987) 171—191

GRAPH BISECTION ALGORITHMS WITH GOOD AVERAGE CASE BEHAVIOR

T. N. BUI, S. CHAUDHURI, F. T. LEIGHTON and M. SIPSER

Received 31 January 1985

Revised 12 December 1985

In the paper, we describe a polynomial time algorithm that, for every input graph, either outputs the minimum bisection of the graph or halts without output. More importantly, we show that the algorithm chooses the former course with high probability for many natural classes of graphs. In particular, for every fixed $d \geq 3$, all sufficiently large n and all $b = o(n^{1-1/\lfloor (d+1)/2 \rfloor})$, the algorithm finds the minimum bisection for almost all d-regular labelled simple graphs with $2n$ nodes and bisection width b. For example, the algorithm succeeds for almost all 5-regular graphs with $2n$ nodes and bisection width $o(n^{2/3})$. The algorithm differs from other graph bisection heuristics (as well as from many heuristics for other NP-complete problems) in several respects. Most notably:

(i) the algorithm provides exactly the minimum bisection for almost all input graphs with the specified form, instead of only an approximation of the minimum bisection,

(ii) whenever the algorithm produces a bisection, it is guaranteed to be optimal (i.e., the algorithm also produces a proof that the bisection it outputs is an optimal bisection),

(iii) the algorithm works well both theoretically and experimentally,

(iv) the algorithm employs global methods such as network flow instead of local operations such as 2-changes, and

(v) the algorithm works well for graphs with small bisections (as opposed to graphs with large bisections, for which arbitrary bisections are nearly optimal).

1. Introduction

Given a $2n$-node graph G, a *bisection* of G is a set of edges whose removal allows G to be disconnected into two n-node subgraphs. A *minimum bisection* of G is a bisection with minimum cardinality. The cardinality of the minimum bisection is also known as the *bisection width* of the graph.

The problem of finding the minimum bisection of a graph lies at the heart of many network problems for which solutions depend on the divide-and-conquer paradigm. A typical example is the VLSI placement and routing problem [7, 9, 20]. Placement and routing programs typically proceed by splitting a network in halves, recursively laying out each half, and then reinserting the wires connecting the two halves. Usually, the quality of the resulting layout depends greatly on the number of

This research was supported by Air Force contract AFOSR—82—0326, DARPA contract N00014—80—C—0622, and an NSF Presidential Young Investigator Award with matching funds from Xerox.

A preliminary version of this paper appeared in the Proceedings of the 25th Symposium on the Foundations of Computer Science, (1984), 181—192.

AMS subject classification (1980):68 C 25

wires that have to be reinserted in the final step, i.e., on the size of the initial bisection. This phenomenon has been observed many times over in practice and has recently been proved formally in a mathematical model of VLSI design [4]. In addition to placement and routing, many other divide-and-conquer based algorithms run much faster or perform much better on graphs that have small bisections (e.g., see [1, 17]).

Considering the widespread applications of the graph bisection problem, it is unfortunate that it is NP-complete [11]. Even worse, there are no known approximation algorithms for graph bisection, and exact algorithms are known only for the special case of trees and bounded-width planar graphs. Even for the case of planar graphs (which always have bisection width $O(\sqrt{n})$ [18]), no approximation algorithms are known.

As a result, most practitioners are forced to resort to heuristics, which they hope will find nearly optimal bisections most of the time. For "most" graphs, this actually happens, but the success of a heuristic may have nothing to do with its cleverness: it is a simple fact that almost all graphs with n nodes and $e \geq (1+\varepsilon)n$ edges for any fixed $\varepsilon > 0$ (or similarly, most n-node $(2e/n)$-regular graphs) have only bisections of size $\Theta(e)$. Hence the worst possible bisection and the minimum bisection differ by at most a constant factor for these graphs. Moreover, as $e/n \to \infty$, this ratio goes to 1, and the best and worst bisections differ by only a low order term.

Because "random" graphs may not serve well to distinguish really good heuristics from so-so or even horrible heuristics (e.g., heuristics that try to maximize the bisection), it is useful to examine graphs for which the minimum bisection is much smaller than the average and/or largest bisections. Numerous papers (for the most part empirical studies) have attempted to do precisely this, but most end up constructing graphs according to a specified procedure that (at best) imposes an upper bound on the bisection width of the constructed graphs. Unfortunately, it is usually not clear what relationship exists between the behavior of an algorithm on an average graph in such a class, and on an average graph with specified properties (such as fixed bisection width). Of course, it is the behavior of algorithms on graphs randomly selected from a class of the latter type that is of greatest interest.

In this paper we investigate the class $\mathcal{G}(n, d, b)$ of labelled simple graphs that have $2n$ nodes, node degree d, and bisection width b for fixed $d \geq 3$ and $b = o(n^{1-1/\lfloor (d+1)/2 \rfloor})$. In other words, we consider precisely the distribution of random $2n$-node, d-regular graphs conditioned on having minimum bisection b. Since every graph with dn edges has average bisection $dn/2$, the minimum bisection for these graphs is much smaller than the average bisection. Moreover, we will show that the graph bisection problem is NP-complete for $\mathcal{G}(n, d, b)$ whenever $b \geq n^{\varepsilon}$ for any constant $\varepsilon > 0$. Hence $\mathcal{G}(n, d, b)$ is a natural and suitable class of graphs for analysis.

The main contribution of this paper is the development of a new polynomial-time algorithm for graph bisection and its analysis. The algorithm is based on the maxflow-mincut theorem and is especially well suited to graphs with small bisections. In fact, for almost all graphs G in $\mathcal{G}(n, d, b)$ with $b = o(n^{1-1/\lfloor (d+1)/2 \rfloor})$, the algorithm finds exactly the minimum bisection of G along with a proof certifying that the bisection is in fact optimal. The certification of optimality is an especially nice feature of this algorithm that is not present in other algorithms.

The new algorithm also works well experimentally. As predicted by the theory, it almost always discovers the minimum bisection for graphs in $\mathcal{G}(n, d, b)$ with small b. It is worth noting that standard procedures like the greedy, simulated annealing [16] and Kernighan-Lin [15] algorithms also appear to do very well for graphs in $\mathcal{G}(n, d, b)$ when $d \geq 4$, but perform very poorly when $d=3$. In fact, these algorithms often produce $\Theta(n)$ bisections for $2n$-node trivalent graphs, even when $b=0$.

The remainder of the paper is divided as follows. In Section 2, we prove several lemmas about graphs in $\mathcal{G}(n, d, b)$, and show that the graph bisection problem is NP-complete for $\mathcal{G}(n, d, b)$ whenever $b \geq n^{\varepsilon}$ for any fixed $\varepsilon > 0$. Our new algorithm is described and analyzed in Section 3. We first consider graphs with $o(\sqrt{n})$ bisection width in Section 3.1, and then graphs with $o(n^{1-1/\lfloor (d+1)/2 \rfloor})$ bisections in Section 3.2. We analyze the running time of the algorithm in Section 3.3. Section 4 contains the experimental data. Remarks and directions for research are included in Section 5. We conclude with acknowledgements and references.

2. Preliminaries

2.1. Analyzing random graphs with small bisection width

Methods of constructing d-regular graphs with uniform probability are well known [5]. In what follows, we extend one such standard method to construct d-regular graphs with bisection width b with near uniform probability for $b = o(n^{1-1/\lfloor (d+1)/2 \rfloor})$.

Step 1. Consider a set of $2n$ distinctly labelled nodes, and randomly designate half of them as left nodes, and half as right nodes. Then replace each node with d distinctly labelled points. (E.g., node 1 is replaced by points 1.1, 1.2, ..., 1.d.)

Step 2. Randomly match b left points to b right points.

Step 3. Randomly match the remaining $dn-b$ left points among themselves and the remaining $dn-b$ right points among themselves.

Step 4. Coalesce each set of d points back into a node.

Step 5. Output the graph, maintaining the node and point labels.

Let $\mathcal{G}^*(n, d, b)$ be the collection of graphs (included according to multiplicity) that are constructed by the previous routine. At first glance it is not clear that $\mathcal{G}^*(n, d, b)$ has any relation at all to $\mathcal{G}(n, d, b)$. For example, $\mathcal{G}^*(n, d, b)$ contains graphs with multiple edges and loops as well as graphs with bisection width less than b. No such graphs are contained in $\mathcal{G}(n, d, b)$. Moreover, graphs in $\mathcal{G}(n, d, b)$ occur with varying frequencies in $\mathcal{G}^*(n, d, b)$, depending on the number of b-bisections in the graph and on the number of ways of labelling points.

Despite all of these obstacles, however, we show in Theorem 1 that any condition that holds with probability $1-o(1)$ for $\mathcal{G}^*(n, d, b)$ as $n \to \infty$ also holds with probability $1-o(1)$ for $\mathcal{G}(n, d, b)$. This result is crucial to the paper since it allows us to analyze the much simpler class $\mathcal{G}^*(n, d, b)$ in order to prove theorems about the more natural class $\mathcal{G}(n, d, b)$. Without such an indirect analysis, it is unlikely that we would be able to prove anything at all about graphs randomly selected from

$\mathcal{G}(n, d, b)$. For example, no closed expression is known for the number of d-regular $2n$-node graphs (simple or otherwise), yet the number of graphs (counted according to multiplicity) contained in $\mathcal{G}^*(n, d, b)$ is easily calculated.

Before proving Theorem 1, however, we need several lemmas. These lemmas highlight some of the more interesting properties of graphs in $\mathcal{G}(n, d, b)$ and $\mathcal{G}^*(n, d, b)$, and will be used throughout the paper. We start with a lemma concerning random pointwise-labelled d-regular graphs (possibly with multiple edges and loops). Such graphs are generated in much the same fashion as graphs in $\mathcal{G}^*(n, d, b)$. The term *pointwise-labelled* refers to the existence of labelled points at each node, as in Step 1 of the procedure for $\mathcal{G}^*(n, d, b)$.

Lemma 1. *There is a constant $c>0$ such that for all $d\geqq 3$, $n\to\infty$ and almost every pointwise-labelled n-node d-regular graph G, every k-node subset S of G (for all $k\leqq \leqq n/2$) is incident to at least cdk edges that connect nodes in S to nodes in $G-S$.*

Proof. Let $M(dn)$ denote the number of pointwise-labelled n-node d-regular graphs. It is easily seen that

$$M(dn) = \binom{dn}{\frac{dn}{2}} (dn/2)!\, 2^{-dn/2}.$$

The number of pointwise-labelled n-node d-regular graphs that have a k-node subset with exactly t connections to the rest of the graph is at most

$$\binom{n}{k}\binom{dk}{t}\binom{dn-dk}{t} t!\, M(dk-t)M(dn-dk-t).$$

Taking the ratio of the two formulas and simplifying, we find that the probability that such a graph has a k-node subset with only $t=cdk$ connections is

$$O\left(\left[c^c(1-c)^{(1-c)/2}\alpha^{(1-c)/2-1/d}\left(1-\frac{c}{\alpha}\right)^{(\alpha-c)/2}\left(1+\frac{1}{\alpha}\right)^{(1+\alpha)(1/2-1/d)}\right]^{-dk}\right)$$

where $\alpha=(n-k)/k\geqq 1$. For $c\leqq 1/4$, $d\geqq 3$ and $\alpha\geqq 1$, this is

$$O\left(\left[c^c(1-c)^{(1-c)/2}e^{-c/2}e^{1/6}\left(\frac{n-k}{k}\right)^{1/24}\right]^{-3k}\right).$$

Since $c^c(1-c)^{(1-c)/2}e^{-c/2}\to 1$ as $c\to 0$, we can conclude that

$$c^c(1-c)^{(1-c)/2}e^{-c/2}e^{1/6} \geqq 1+\delta$$

for all sufficiently small c and some constant $\delta>0$. Hence for small enough $c>0$, the probability that a pointwise-labelled n-node d-regular graph has a k-node subset with less than cdk connections is

$$O\left(cdk\left[(1+\delta)\left(\frac{n-k}{k}\right)^{1/24}\right]^{-3k}\right).$$

It is easily checked that the preceding expression converges to 0 as $n\to\infty$ for all $1\leqq k\leqq n/2$. In fact, the sum of these terms for $1\leqq k\leqq n/2$ is $O(n^{-1/8})$ which also

converges to 0. Thus the claim holds simultaneously for all k-node subsets in almost every graph. ∎

Results such as Lemma 1 are common in probabilistic graph theory and have numerous useful applications. We include one such application in the following corollary. Although the result is only stated for pointwise-labelled graphs, possibly containing loops and multiple edges, it is easily extended to simple labelled d-regular graphs.

Corollary 1. *For all* $d \geq 3$, $n \to \infty$ *and almost every pointwise-labelled n-node d-regular graph G, every bisection of G has size between* $(1-\varepsilon)dn/4$ *and* $(1+\varepsilon)dn/4$ *where* $\varepsilon = O(1/\sqrt{d})$.

Proof. Setting $k = n/2$ and $\alpha = 1$ in the proof of Lemma 1, we find that the probability that G has a bisection of size $cdn/2$ is

$$O([c^c(1-c)^{1-c}2^{1-2/d}]^{-dn/2}).$$

This expression is maximized at $c = 1/2$ and thus the probability that G has a bisection of size less than $(1-\varepsilon)dn/4$ or greater than $(1+\varepsilon)dn/4$ is

$$O\left(n\left[\left(\frac{1+\varepsilon}{2}\right)^{(1+\varepsilon)/2}\left(\frac{1-\varepsilon}{2}\right)^{(1-\varepsilon)/2}2^{1-2/d}\right]^{-dn/2}\right).$$

Simplifying, we find that the preceding expression is

$$O(n[e^{\varepsilon^2/2}2^{-2/d}]^{-dn/2})$$

which tends to 0 for $\varepsilon > 2/\sqrt{d \log e}$. ∎

Of more immediate concern to us in this paper is the following corollary to Lemma 1. In the corollary and throughout the rest of the paper, the phrase *left* or *right half of a graph in* $\mathcal{G}^*(n, d, b)$ refers to the left or right, respectively, nodes created in Step 1 of the procedure for $\mathcal{G}^*(n, d, b)$ and to the edges inserted in Step 3, but does not include the bisection edges inserted in Step 2.

Corollary 2. *There is a constant* $c > 0$ *such that for all* $d \geq 3$, $n \to \infty$ *and almost every graph G that forms the left or right half of a graph in* $\mathcal{G}^*(n, d, b)$, *every m-node subset S of G that is incident to t bisection edges, is also incident to at least* $cdm - t$ *edges connecting S to* $G - S$ *for all* $m \leq n/2$ *and* $t \leq b$.

Proof. For simplicity, assume b is even and randomly connect the b bisection points in G with $b/2$ edges to form a new graph G'. It is easily observed that G' is a random d-regular pointwise-labelled graph, possibly containing loops or multiple edges. Hence, if G' is one of the $1 - o(1)$ portion of d-regular graphs satisfying Lemma 1, there will be at least cdm edges connecting S to $G' - S$. In that case, there clearly must be at least $cdm - t$ edges connecting S to $G - S$ in G. ∎

The following lemmas will serve to further strengthen Corollary 2.

Lemma 2. *Given any* $r \geq 2$, $d \geq 3$, $m = o(n^{1-1/r})$ *and* $n \to \infty$, *if m items are chosen at random from n groups of d items each, then with probability* $1 - o(1)$ *fewer than r items will be selected from each group. Moreover, the same conclusion holds provided that*

each item is selected at random from some (possibly varying) subset of at least $n-m$ groups.

Proof. Assume that each item is selected at random from some subset of at least $n-m$ groups. The probability that the ith item selected comes from the jth group is at most $d/[(n-m)d-m]$ since there are at least $n-m$ groups of d items to choose from and at most $i-1<m$ of the items have already been chosen. Hence, the probability that k items are chosen from the same group is at most

$$n\binom{m}{k}\frac{1}{\left(n-m-\frac{m}{d}\right)^k}.$$

Simplifying, we find that this probability is $O(m^k/n^{k-1})$ which for $m=o(n^{1-1/r})$ is $o(n^{1-k/r})$. Summing over $k\geq r$, we find the probability that fewer than r items are selected from each group is $1-o(1)$. ∎

Lemma 3. *For all fixed $d\geq 3$, all $b=o(n^{1-1/\lfloor(d+1)/2\rfloor})$, $n\to\infty$ and almost every graph G that forms the left or right half of a graph in $\mathcal{G}^*(n, d, b)$, every subset of G with at most $n/2$ nodes that is incident to k bisection edges for any $k\leq b$, is also incident to at least $k+1$ edges connecting nodes in S to nodes in $G-S$.*

Proof. Let G be the left half (without loss of generality) of a graph constructed according to the procedure for $\mathcal{G}^*(n, d, b)$. In what follows, we will show that the nodes of G which are incident to bisection edges have sufficiently bushy neighborhoods so that any set S incident to k bisection edges and at most k edges that connect nodes in S to nodes in $G-S$ must contain at least $4k/cd$ nodes, where c is the constant defined in Corollary 2. We will then use Corollary 2 to obtain a contradiction of the hypothesis that S is incident to at most k edges which link S to $G-S$.

Without loss of generality, we can assume that the edges created in Step 3 to form G were generated in order of increasing distance from the bisection edges. In particular, we are interested in the generation of edges within distance $d\log(1/c)$ of the bisection where c is the constant defined in Corollary 2. For fixed d, there are at most $m=O(b)=o(n^{1-1/\lfloor(d+1)/2\rfloor})$ such edges. Applying Lemma 2 to the node by node generation of edges to form G, it is easily shown that, with high probability each node of G within distance $d\log(1/c)$ is incident to fewer than $\lfloor(d+1)/2\rfloor$ previously generated edges (i.e., edges that are also incident to previously processed nodes).

Let S be a set of at most $n/2$ nodes of G that is incident to k bisection edges and at most k edges that connect S to $G-S$. Let e_i denote the number of edges in S that link two nodes which are of distance i from the bisection and $e_{i,i+1}$ denote the number of edges in S that link nodes which are of distance i and $i+1$ from the bisection. (Throughout this proof, *distance* means the length in edges of the shortest path totally contained in S to the bisection. Nodes incident to bisection edges are considered to be of distance 1 from the bisection.) By definition, $e_{0,1}=k$. Also define n_i to be the number of nodes in S at distance i from the bisection, and f_i to be the number of edges that link a distance i node of S to $G-S$. By assumption $f_0=0$ and $\sum_{i=0}^{\infty} f_i\leq k$.

Because the edges of S were generated in order of increasing distance from the bisection, and since each node is incident to at most $(d-1)/2$ previously generated edges, we can deduce that

$$(*) \qquad n_i \geq \frac{e_{i-1,i}+e_i}{\frac{d-1}{2}}$$

and that

$$e_{i,i+1} \geq n_i\left(\frac{d+1}{2}\right)-e_i-f_i$$

for every $i \leq d \log (1/c)$. Combining the two inequalities, we find that

$$\begin{aligned} e_{i,i+1} &\geq \left(1+\frac{2}{d-1}\right)e_{i-1,i}+\frac{2}{d-1}e_i-f_i \\ &\geq \left(1+\frac{2}{d-1}\right)e_{i-1,i}-f_i. \end{aligned}$$

It is not difficult to show that $e_{i,i+1}$ is minimized for every $i \leq d \log (1/c)$ by setting $f_1=k$ and $f_i=0$ for $i \geq 2$. Then it is clear that

$$(**) \qquad \begin{aligned} e_{i,i+1} &\geq \left(1+\frac{2}{d-1}\right)^{i-1}\left(\left(1+\frac{2}{d-1}\right)e_{0,1}-k\right) \\ &= \frac{2k}{d-1}\left(1+\frac{2}{d-1}\right)^{i-1}. \end{aligned}$$

Hence for $i \leq d \log (1/c)$ using $(*)$ and $(**)$ we get $\sum_i n_{i+1} > 4k/cd$ and hence S contains at least $4k/cd$ nodes. By Corollary 2, however, this means that there are at least $cd(4k)/cd-k=3k$ edges linking S to $G-S$. This provides the necessary contradiction and concludes the proof. ∎

We are now able to prove the main result of this section.

Theorem 1. *For all fixed $d \geq 3$ and all $b=o(n^{1-1/\lfloor (d+1)/2 \rfloor})$, any condition that holds with probability $1-o(1)$ for $\mathcal{G}^*(n, d, b)$ as $n \to \infty$ also holds with probability $1-o(1)$ for $\mathcal{G}(n, d, b)$ as $n \to \infty$.*

Proof. We first observe that every graph G in $\mathcal{G}(n, d, b)$ that has a unique minimum bisection is generated with the same frequency in $\mathcal{G}^*(n, d, b)$. This is because the partition into left and right halves at Step 1 of the procedure for $\mathcal{G}^*(n, d, b)$ is precisely determined by the unique b-bisection of G. Since G is labelled, the edges are also distinguished. Finally, since G contains no multiple edges or loops, there are exactly $d!$ ways to pointwise label the d edges incident to each node. Hence there are $(d!)^{2n}$ pontwise labellings for each labelled graph.

Graphs in $\mathcal{G}(n, d, b)$ with nonunique b-bisections appear proportionally more often in $\mathcal{G}^*(n, d, b)$ than do graphs with unique b-bisections. Moreover, $\mathcal{G}^*(n, d, b)$

also contains graphs with multiple edges and loops, and with bisection width less than b. However, we will show in what follows that these bad graphs constitute at most a constant fraction of the graphs in $\mathcal{G}^*(n, d, b)$. We commence by showing that $\Omega(e^{-(d^2-1)/2})$ of the graphs generated for $\mathcal{G}^*(n, d, b)$ have no loops or multiple edges. For fixed d, this is a constant fraction.

The probability of generating a multiple edge during an insertion of a bisection edge in Step 2 is at most

$$b\frac{(d-1)^2}{(dn-b)^2} \leqq \frac{b}{n^2} \leqq \frac{1}{n}.$$

Hence the probability of avoiding multiple edges altogether during Step 2 is at least

$$\left(1-\frac{1}{n}\right)^b \geqq 1-\frac{b}{n} = 1-o(1).$$

The probability of avoiding loops and multiple edges during Step 3 is at least $\Omega(e^{-(d^2-1)/2})$. To prove this, we apply a result of Bollobás [5] that a random pointwise-labelled d-regular graph contains no loops or multiple edges with probability $\Omega(e^{-(d^2-1)/4})$. If the left and right sides of a graph in $\mathcal{G}^*(n, d, b)$ are randomly extended to become d-regular, then with probability $\Omega(e^{-(d^2-1)/2})$, neither contains loops or multiple edges. Hence, at least $\Omega(e^{-(d^2-1)/2})$ of the graphs in $\mathcal{G}^*(n, d, b)$ contain neither loops nor multiple edges.

We conclude the proof by showing that only a small portion of the graphs occurring in $\mathcal{G}^*(n, d, b)$ have bisections less than b or multiple bisections of size b. This fact is an immediate consequence of Lemma 3, since the existence of such a bisection for a graph G in $\mathcal{G}^*(n, d, b)$ would imply the existence of a subset S with at most $n/2$ nodes in the left or right half of G that is incident to k bisection edges for some $k \leqq b$ but to k or fewer other edges that link S to $G-S$.

In conclusion, sampling graphs in $\mathcal{G}(n, d, b)$ is equivalent to sampling a constant portion $\Omega(e^{-(d^2-1)/2})$ of the graphs in $\mathcal{G}^*(n, d, b)$ and ignoring point labels. Hence, any condition that holds for $1-o(1)$ of the graphs in $\mathcal{G}^*(n, d, b)$ must also hold for $1-o(e^{-(d^2-1)/2})=1-o(1)$ of the graphs in $\mathcal{G}(n, d, b)$. ∎

2.2. Proof of NP-completeness

In this section we will show that the problem of deciding whether or not a d-regular graph has a bisection of size b or less is NP-complete whenever $d \geqq 3$ and $b=n^\varepsilon$ for any fixed ε in the range (0, 1). We will reduce the general graph bisection problem to this problem. The proof will be done in two steps. Given a graph G and an integer b, we transform G to a 3-regular graph G^* such that G has a bisection of size b or less if and only if G^* has a bisection of size b or less. We next transform G^* into a d-regular graph G' such that G^* has a bisection of size b or less if and only if G' has a bisection of size b' or less, where $b'=n^\varepsilon$, and n is the size of G', for any fixed $\varepsilon \in (0, 1)$. We start with the following lemma.

Lemma 4. *Let H be an n-node honeycomb-like 3-regular graph as in Figure 1. Every s-node subset S of H, where $s \leqq n/2$, is adjacent to at least $\sqrt{s/2}$ nodes not in S.* ∎

The proof of this lemma is not difficult and we will omit it.

Theorem 2. *The problem of deciding whether or not a d-regular graph has a bisection of size b or less is* NP*-complete, whenever $d \geqq 3$ and $b = n^{\varepsilon}$ for any fixed $\varepsilon \in (0, 1)$.*

Proof. Let G be an n-node graph and b an integer. We will construct a 3-regular graph G^* on $m = \Theta(n^6)$ nodes as follows. Replace each node of G with an n^5-node honeycomb-like graph H of Lemma 4. An edge between two nodes in G is replaced by an edge connecting two edges of the two corresponding graphs H in G^*, thus creating two new nodes of degree 3 (see Figure 2). Furthermore, edges coming into a graph H are dispersed widely so that any r-subset R of H which is incident to s incoming edges will also be incident to at least $s + \sqrt{r/4}$ nodes in $H - R$. This can be done using Lemma 4.

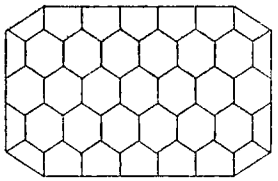

Fig. 1. An example of a honeycomb-like graph

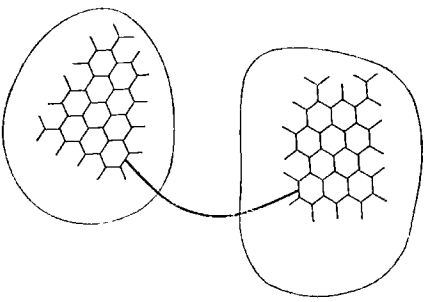

Fig. 2. The connection of two H-graphs

Let B be a minimum bisection of G^*, i.e., B is a minimum set of edges whose removal divides G^* into two subgraphs of equal size.

Claim 1. *B induces a corresponding bisection of G, i.e., B contains only edges that correspond to the original edges of G.*

Proof. Suppose not, then we can rearrange the bisection to obtain a new cut by moving each copy of H that is cut by B entirely to the side of the bisection containing the majority of its nodes. Suppose we have to move t nodes, then the new cut has at least $\sqrt{t/4}$ fewer edges than the original bisection. This new cut, however, might no longer be a bisection. To make it a bisection we have to move at most t/n^5 H-graphs. This will increase the size of the cut by at most t/n^4 edges. Since the size of any bisection of G is at most n^2, it can be easily seen that t is at most n^4. Thus the new bisection is smaller than the original one, a contradiction. ∎

The next step is to transform G^* into a d-regular graph G' as follows. Replace each edge of G^* with k edges where k is such that $bk = (100mk)^c$. We then replace each node of G^* with a graph H' satisfying the following conditions:

(i) H' has $3k$ degree $d-1$ nodes and $97k$ degree d nodes, and
(ii) every subset of H' with $r \leqq 50k$ nodes, s of which are degree $d-1$ nodes, is incident to at least $s+1.3r/40$ nodes not in the subset.

Graphs like H' are easily constructed from expander graphs, even for $d=3$.

Edges in G^* now can be replaced by edges connecting the $(d-1)$-degree nodes in the corresponding H' graphs. The resulting graph G' will then be d-regular.

Claim 2. *G' has a bisection of size bk or less if and only if G^* has a bisection of size b or less.*

Proof. Similar to the one before. Given a bisection of G', form a new cut by moving each copy of H' entirely to the side of the bisection containing the majority of its nodes. If t nodes are moved in this step then the new cut contains at least $1.3t/40$ fewer edges than the original bisection. Although the new cut corresponds nicely to a cut of G^*, it is not necessarily a bisection. To make a bisection, move up to $t/(100k)$ copies of H' from one side to the other. This increases the cut by at most $3t/100$ edges which is less than the $1.3t/40$ edge decrease performed earlier.

Thus a bk-bisection of G' can be converted into a b-bisection of G. This proves Claim 2 and the theorem. ∎

3. A bisection algorithm that almost always works

In this section, we describe a graph bisection algorithm that finds the minimum bisection for almost every graph G in $\mathcal{G}(n, d, b)$ for fixed $d \geqq 3$ and $b = o(n^{1-1/\lfloor (d+1)/2 \rfloor})$. In addition, the algorithm is constructed so that every time a bisection is output, it is guaranteed to be optimal.

The idea of the algorithm is quite simple: we wish to convert G into an instance of the maxflow problem for which the mincut is the minimum bisection. Of course, it's hard to do this without knowing which edges comprise the minimum bisection, but we can come close. In fact, we will find that by replacing the neighborhoods around two nodes u and v with an infinite capacity source and sink, the resulting flow problem will often have a mincut close to a bisection. By exploiting this phenomenon, we are able to prove the desired result.

The description of the algorithm and its analysis is divided into three subsections. In Section 3.1, we present and analyze a simple algorithm for graphs with $o(\sqrt{n})$ bisections. The general algorithm is described and analyzed in Section 3.2. In Section 3.3, we bound the running time of the algorithms.

Throughout, we will state and prove "almost all"-type theorems for graphs in $\mathcal{G}^*(n, d, b)$. By Theorem 1, such results also hold for graphs in $\mathcal{G}(n, d, b)$. We start by proving one such result for the size of neighborhoods around nodes in the left and right halves of graphs in $\mathcal{G}^*(n, d, b)$.

Lemma 5. *For all fixed $d \geqq 3$, all $b = o(n^{1-1/\lfloor (d+1)/2 \rfloor})$, $n \to \infty$ and almost every graph G that forms the left or right half of a graph in $\mathcal{G}^*(n, d, b)$, every node of G is within distance $\log_{(d-1)} m + O(1)$ of at least m other nodes for every $m = o(n^{1-1/\lfloor (d+1)/2 \rfloor})$.*

Proof. Let G be a graph that forms the left or right half of a graph in $\mathcal{G}^*(n, d, b)$ and let v be a fixed node of G. We will show that with probability $1-o(1/n)$, there are m nodes within distance $\log_{(d-1)} m+O(1)$ of v for every $m=o(n^{1-1/\lfloor(d+1)/2\rfloor})$. Hence, with probability $1-o(1)$, this condition will be true for every node of G. (Note that distance in G is measured by paths that are contained entirely within G. Artificial bisection edges inserted in Step 2 of the graph generating procedure are not allowed in such paths.)

Without loss of generality, we can select the edges of G in Step 3 of the procedure for $\mathcal{G}^*(n, d, b)$ in order of increasing distance from v. Initially, we try to select neighbors for v. Of course, some of the points comprising v may be incident to bisection edges (thus not having a neighbor in G), some might be incident to other points in v, and some might be incident to points in some other common node of G. Let n_1 denote the number of nodes in G found to be adjacent to v via a single edge. Similarly, define n_i to be the number of nodes selected only once to be adjacent to a node of distance $i-1$ from v. For each i, it is easily shown that $n_{i+1} \geqq n_i(d-1)-2r_i$, where r_i is the number of points at distance i from v that become incident to a bisection edge, to another point at distance i, or to a point in a node that already is known to be of distance $i+1$ from v. The probability that a point falls into one of these classes is at most

$$\frac{b+n_i d+n_i d^2}{nd-md} = o(n^{-1/\lfloor(d+1)/2\rfloor})$$

since $n_i \leqq m = o(n^{1-1/\lfloor(d+1)/2\rfloor})$ and d is fixed.

Hence the probability that r_i of the $n_i(d-1)$ points fall into this bad class is at most

$$\binom{n_i(d-1)}{r_i} o(n^{-r_i/\lfloor(d+1)/2\rfloor}) = o\left(\left[\frac{n_i(d-1)e}{r_i n^{1/\lfloor(d+1)/2\rfloor}}\right]^{r_i}\right)$$

$$= o\left(\left[\frac{n_i de}{r_i n^{2/(d+1)}}\right]^{r_i}\right).$$

For $n_i \leqq n^{1/(d+1)}$, choosing $r_i = 4d$ is more than sufficient to make this probability $o(1/n^2)$. Otherwise, it is sufficient to make $n_i de/r_i n^{2/(d+1)} \leqq 1/2$ and $r_i \geqq 2 \log n$. For $n_i \geqq n^{1/(d+1)}$, this can be accomplished by setting $r_i = 2n_i \log n/n^{1/(d+1)}$.

Thus for any $m=o(n^{1-1/\lfloor(d+1)/2\rfloor})$, we can conclude that with probability $1-o(1/n)$,

$$n_{i+1} \geqq n_i(d-1)-8d$$

for all i such that $n_i \leqq n^{1/(d+1)}$, and

$$n_{i+1} \geqq n_i\left(d-1-\frac{4\log n}{n^{1/(d+1)}}\right)$$

for all i such that $n_i \leqq m$. Provided that n_f is greater than $8d/(d-2)$ for some constant f, the first recurrence can be solved to find that $n_i = \Theta((d-1)^{i-f})$. The second recurrence extends this result to large i. Hence with probability $1-o(1/n)$, v has m neighbors within distance $\log_{(d-1)} m+O(1)$. Although we have omitted the proof that n_f is greater than $8d/(d-2)$ for some constant f (with probability $1-o(1/n)$), the details are not difficult to work out. ∎

3.1. Bisecting graphs with $o(\sqrt{n})$ bisection width

Let G be a d-regular graph. For each node v in G, define the *neighborhood* $N(v)$ of v to be the set of all nodes within distance $\log_{(d-1)} \sqrt{n}-2$ of v. For each pair of nodes u and v, the algorithm finds the mincut $c(u, v)$ in G using the maxflow-mincut algorithm when $N(u)$ is replaced by an infinite capacity source and $N(v)$ is replaced by an infinite capacity sink. More precisely, the edges linking $N(u)$ and $N(v)$ to $G-N(u)-N(v)$ are replaced by edges linking $G-N(u)-N(v)$ directly to the source and sink, respectively. Edges of G linking nodes contained in $N(u)$ to nodes contained in $N(v)$ are replaced by edges linking the source and sink directly. If a cut with the minimum cardinality is a bisection, then the algorithm outputs that cut. Otherwise, the algorithm halts without output. We call this procedure Algorithm 1.

We first show that Algorithm 1 never outputs a suboptimal bisection.

Theorem 3. *Whenever Algorithm 1 outputs a bisection for a d-regular graph, it is guaranteed to be the minimum bisection.*

Proof. Suppose a graph G has a bisection of size b' that is less than the bisection of size b output by the algorithm. Since the sources and sinks are grown to a distance of $\log_{(d-1)} \sqrt{n}-2$, it is easily shown that every mincut has size at most $\sqrt{n}$, and thus $b' < \sqrt{n}$.

Given that G is d-regular for some d, the number of nodes within distance r of the b'-bisection in each half of the $2n$-node graph is at most $2b'(d-1)^r$. Hence, at least half of the nodes in each half of G (with respect to the b'-bisection) have distance greater than $\log_{(d-1)}(n/4b') \geq \log_{(d-1)}(\sqrt{n}/4)$ from the b'-bisection. Hence, the algorithm finds at least one such pair of nodes on opposite sides of the bisection. Because the sources and sinks grown out from these nodes extend for distance at most $\log_{(d-1)} \sqrt{n}-2$, neither will cross the b'-bisection. Hence, the maxflow between the two can be at most b'. This is a contradiction since the algorithm would not have output a b-bisection had there been a mincut of size $b' < b$. ∎

From the preceding analysis, it is clear that Algorithm 1 never outputs bisections for graphs with bisection width greater than $\sqrt{n}$. For almost all graphs with $o(\sqrt{n})$ bisection width, however, the smallest mincut found in Algorithm 1 is precisely the minimum bisection.

Theorem 4. *For all* $d \geq 3$, *all* $b = o(\sqrt{n})$, $n \to \infty$ *and almost every graph* G *in* $\mathcal{G}^*(n, d, b)$; *Algorithm 1 outputs the minimum bisection of* G.

Proof. We will show that for almost all G, the smallest of the mincuts (over all pairs of sources and sinks) is precisely the bisection artificially inserted into G during Step 2 of the procedure for $\mathcal{G}^*(n, d, b)$. The fact that this bisection is optimal then follows from Theorem 3.

There are two cases to consider depending on whether the source and sink originate on the same or different sides of the bisection. In either case, they encompass at least $3b/cd$ nodes on their respective sides (by Lemma 5), where c is the constant defined in Corollary 2. If they are on the same side, then by Corollary 2, the mincut separating them must contain at least $cd(3b/cd)-b \geq 2b$ edges of G (includ-

ing edges incident to the source and/or sink). Such large cuts have no impact on the output.

If the source and sink originate on opposite sides of the bisection, there are again two cases to consider depending on whether or not either includes one or more edges of the bisection. By the arguments in the proof of Theorem 3, at least 1/4 of such source-sink pairs will not reach the bisection. In this case, Corollary 2 and Lemma 3 are easily combined to show that the mincut between the source and sink is precisely the bisection. Were another cut of smaller or equal size to exist, then there would be a cut with k or fewer edges separating the source from k of the bisection edges in (without loss of generality) the left half of G for some k. If the smaller piece of this cut contains the source, Corollary 2 provides a contradiction as before. Otherwise, Lemma 3 applies to provide the contradiction.

If the source and sink originate on opposite sides of the bisection, but one or both includes one or more edges of the bisection, then the mincut must be greater than b. This is because both the source and sink still encompass at least $3b/cd$ nodes on their respective sides. By the argument in the preceding paragraph, however, the implanted bisection is the only cut of size b or smaller separating two such large sets. Since at least one edge of the bisection is included inside the source or sink, that bisection no longer separates them. Hence, the mincut that does separate them must be larger.

In conclusion, at least 1/8 of the source-sink pairs will produce a unique mincut that is the bisection. The remainder will produce larger cuts. ∎

3.2. Bisecting graphs with larger bisection width

Algorithm 1 does not work for graphs with bisection width $b=\Omega(\sqrt{n})$ since the neighborhoods required for such graphs must be grown to a depth of $\log_{(d-1)} b + +\Theta(1)$ and, as a consequence, will almost always contain part of the minimum bisection. Hence the minimum bisection is not likely to appear as a mincut for any source-sink pair.

However, it is possible to prove that for almost all graphs in $\mathcal{G}^*(n, d, b)$ with $b=o(n^{1-1/\lfloor(d+1)/2\rfloor})$, many of the mincuts will contain all the bisection edges not absorbed by the source and sink and, otherwise, only edges that are incident to the source and/or sink. Hence, by summing the number of times each edge appears in a mincut $c(u, v)$ over all pairs u and v, it is possible to readily distinguish the edges in the minimum bisection of such graphs (since they are guaranteed to appear in many more mincuts than edges not in the bisection). This process is the first phase of Algorithm 2. Phase II is designed to verify that bisections found in Phase I are, in fact, optimal. A more detailed description of Algorithm 2 follows.

Algorithm 2. (Do both phases for $q=2, 4, 8, \ldots, o(n^{1-1/\lfloor(d+1)/2\rfloor})$ and then halt.)

Phase I. Initial computation of bisection.

Step 1. For each node v in G define the neighborhood $N(v)$ of v to be the set of all nodes within distance $\log_{(d-1)} q$ of v.

Step 2. For each pair of nodes u and v in G, compute the mincut $c(u, v)$ in G using the maxflow-mincut algorithm when $N(u)$ is replaced by an infinite capacity source and

$N(v)$ is replaced by an infinite capacity sink. (As in Algorithm 1, the edges linking $N(u)$ or $N(v)$ to $G-N(u)-N(v)$ are replaced by edges linking the source or sink to $G-N(u)-N(v)$, respectively. Edges linking nodes contained in $N(u)$ to nodes contained in $N(v)$ are replaced by edges linking the source and sink.)

Step 3. Let B be the set of b edges of G contained in at least $n^2/2$ of the $\binom{2n}{2}$ min-cuts. If B is a bisection then proceed to Phase II. Otherwise, proceed with the next value of q.

Phase II. Verification that B is the minimum bisection.

Step 4. Repeat Steps 1 and 2 above for all u and v on opposite sides of B, except replace each edge of B with an edge of capacity $1+1/d$ and restrict the construction of the sources and sinks so that they do not cross from one side of B to the other.

Step 5. Check that the maxflows computed in Step 4 all have size $b(1+1/d)$. If this is the case, then output B and halt. Otherwise, proceed with the next value of q.

In Theorems 5 and 6 we will show that Algorithm 2 never outputs a suboptimal bisection, and almost always finds the optimal bisection. Both theorems make use of the following simple lemma.

Lemma 6. *For every $2n$-node graph G with node degree at most d, and every s-edge subset S of G,*

$$\sum_{v \in G} \varrho(v, r) \leqq 4sr(d-1)^{r-1}$$

for all r, where $\varrho(v, r)$ is the number of nodes reachable by a path of length r or less originating from v and traveling through S.

Proof. The claim is proved by bounding the number of paths of length r or less that pass through one of the s edges of S. The number of such paths is clearly at most

$$4s \sum_{i=0}^{r-1} (d-1)^i (d-1)^{r-i-1}$$

since at most $2(d-1)^i$ nodes are within distance i of the one side of an edge of S and at most $2(d-1)^{r-1-i}$ are within distance $r-1-i$ of the other side for any i. Simplifying the preceding expression then gives the desired bound. ∎

Theorem 5. *For sufficiently large n, whenever Algorithm 2 outputs a bisection, it is guaranteed to be the minimum bisection.*

Proof. Suppose a $2n$-node d-regular graph G has a bisection B' of size b' which is less than the size b of the bisection B output by Algorithm 2. In what follows, we will show that this implies that a substantial portion of the source-sink pairs computed in Step 4 have flow less than $b(1+1/d)$, thus establishing a contradiction.

A simple counting argument reveals that at least half of the source-sink pairs on opposite sides of B originated with a pair of nodes u and v that are also on opposite sides of B'. The maximum flow for such a pair is at most

$$b' + \frac{1}{d}(b'-s) + dm \leqq b\left(1+\frac{1}{d}\right) + dm - \frac{s}{d}$$

where s is the number of edges in $S=B'-B$ and m is the number of nodes included in $N(u)$ or $N(v)$ but that are across B' from u or v, respectively.

Since the source and sink cannot cross or include edges in B, only nodes that have short paths through S to u or v can be opposite B' from u or v and still be included in the source or sink, respectively. Hence, by Lemma 6 we can deduce that

$$m \leqq \frac{16sr(d-1)^{r-1}}{n}$$

for at least 1/4 of the source-sink pairs that are on opposite sides of B and of B', where $r=\log_{(d-1)} q$ is the radius of the sources and sinks defined in Algorithm 2. Since $q \leqq n^{1-1/\lfloor (d+1)/2 \rfloor}$, dm is much less than s/d for large values of n, thus giving the contradiction. ∎

By Theorem 5, we know that bisections output by Algorithm 2 are optimal whenever n satisfies

$$16d \log_{(d-1)} n < n^{1/\lfloor (d+1)/2 \rfloor}.$$

For small values of n, the inequality is not satisfied although the result is probably still true.

Theorem 6. *For all* $d \geqq 3$, *all* $b=o(n^{1-1/\lfloor (d+1)/2 \rfloor})$, $n \to \infty$ *and almost every graph* G *in* $\mathcal{G}^*(n, d, b)$, *Algorithm 2 outputs the minimum bisection of* G.

Proof. We consider a pass of Algorithm 2 when q is much larger than b, but is still much smaller than $n^{1-1/\lfloor (d+1)/2 \rfloor}$. For such q, we show that with probability $1-o(1)$, all of the artificially inserted bisection edges of G are included in at least $(1-o(1))n^2$ of the $\binom{2n}{2}$ mincuts, and that all other edges are included in only $o(n^2)$ mincuts. Hence, precisely the artificial bisection B is identified at the end of Phase I for almost all G in $\mathcal{G}^*(n, d, b)$. We conclude by showing that B almost always satisfies the conditions checked in Phase II, thus completing the proof.

From Lemma 6 it can be deduced that no more than $O(q \log n)$ sources or sinks which start from one side of B can include more than $o(b)$ edges of B or nodes and edges on the opposite side of B. Thus $1-O(q \log n/n)=1-o(1)$ of all the sources and sinks will stay (for the most part) on the side of B from which they start. Note that those which do not can only contribute $o(n^2)$ to the total count of any edge, and can be safely disregarded henceforth.

We divide the remaining analysis into 2 cases: (i) the source-sink pairs that start from the same side of B, and (ii) the source-sink pairs that start from opposite sides of B.

Case (i). By the above observation, all but $o(1)$ of the n^2-n source-sink pairs that start on the same side of B stay (for the most part) on the same side. Applying the argument used in Lemma 3 (growing edges from the source and sink), we find that the mincut for such a pair consists solely of edges incident to the source or sink with probability $1-o(1)$. Hence for $1-o(1)$ of the graphs G, $1-o(1)$ of the mincuts of this type include only edges incident to the source or sink.

Since an edge can be on the frontier of a source or sink for at most $O(qn) = o(n^2)$ source-sink pairs, the preceding analysis means that edges not in B are included in at most $o(n^2)$ mincuts during Phase I of the algorithm for almost all G.

Case (ii). This is similar to the above case. In particular we can show that for almost all G, $1-o(1)$ of the n^2 mincuts consist precisely of the edges of B not in a source or sink along with the edges incident to a source or sink but on the opposite side of B from the origin of the source or sink, respectively. Hence, nonbisection edges are included in $o(n^2)$ mincuts for almost all G. On the other hand, the observation that every bisection edge is in at most $O(b)=o(n)$ sources or sinks implies that each edge of B is included in at least $n^2-o(n^2)$ mincuts. Hence B is distinguished for almost all G at the end of Phase I.

The analysis of Phase II is easier than that of Phase I since the sources and sinks are not allowed to cross B. We need only mimic the proof of Theorem 4, substituting higher capacity edges at the bisection in the proof of Lemma 3. Thus, with probability $1-o(1)$, all Phase II source-sink pairs have mincuts at B with size $b(1+1/d)$. ∎

3.3. Running time analysis

As stated, each pass of Algorithm 2 solves $O(n^2)$ flow problems on graphs with $2n$ nodes and dn edges. At most $O(dq)$ augmenting paths (each carrying $1/d$ unit of flow) need to be found for each flow problem, and each requires at most $O(dn)$ steps in the worst case. Hence Algorithm 2 can always be made to run in $O(d^2n^{4-1/\lfloor(d+1)/2\rfloor})$ steps. However, by modifying the algorithm slightly, this bound can be substantially improved. For example, by modifying the algorithm to check that most of the mincuts have $\Theta(q)$ edges at each pass before proceeding to the next value of q, the worst case running time can be improved to $O(d^2bn^3)$ where b is the bisection width of the graph. (This can be proved by showing that for $q \leqq b$, this condition is almost always satisfied, but for $q \gg b$ the condition is never satisfied.)

A more substantial improvement can be achieved by finding the mincuts for only a small random sample of the source-sink pairs. A more careful look at the proof of Theorems 5 and 6 reveals that only $\log n$ source-sink pairs are needed to insure the results with probability $1-o(1/n)$ for any graph. For the upper bounds on bisection, this probability can be incorporated into the $1-o(1)$ term in Theorem 6. The randomization has a more serious impact on the lower bound, however, since lower bound proofs would then only be correct with probability $1-o(1/n)$. In any case, the running time for the probabilistic version of the algorithm is $O(d^2bn \log n)$. If we only require correct answers with probability $1-o(1)$, then the $\log n$ term can be replaced by any increasing function. If we remove the lower bound portion of the algorithm entirely, then the expected time is $O(d^2bn)$.

Savings can also be made in the flow algorithm itself. By restricting ourselves to unit size flows in all but the bisection edges, one of the d factors can be removed. Although we do not yet have a proof, it is quite possible that even greater savings can be obtained by using the properties of random graphs to show that the augmenting paths are usually found in far fewer than $\Theta(dn)$ steps. In fact, it might be the case that the ith augmenting path can be found in $O(n/(b-i))$ steps. If so, this would replace

the *dbn* term in the preceding expressions with an $n \log b$ term. Hence, the expected time to find the upper bound might be as fast as $O(n \log b)$.

In practice, the probabilistic version of the algorithm runs very quickly. When computing the data in Section 4, the algorithm appeared to be much faster than both the Kernighan—Lin algorithm and simulated annealing.

4. Experimental data

We have fine-tuned and tested probabilistic versions of Algorithms 1 and 2 on over 100 graphs. Except for rare cases when a node was incident to more than $\lfloor (d-1)/2 \rfloor$ bisection edges or when an edge linked two nodes that were each incident to $(d-1)/2$ bisection edges, the algorithms always identified the correct bisection. Even in the few cases when a bisection was not precisely identified, a quick look at vertices near the cut revealed the irregularity and the optimal bisection. Such cases are identified by asterisks in Tables, I, II and III.

Bisections with less than $\sqrt{n}/2$ edges were verified (with very high probability) to be optimal. Larger bisections are also probably optimal but we did not verify them.

For comparison, we also tested the Kernighan-Lin, simulated annealing and greedy algorithms on many of the same graphs. As can be seen from the data, none of these algorithms performed very well for degree 3 graphs, although the Kernighan-Lin and simulated annealing algorithms performed dramatically better as the degree was increased. Experiment with graphs of higher degree showed that the performance of the greedy algorithm also increased but at a slower rate. The data was obtained using the standard Kernighan-Lin algorithm [15] and the Johnson [14] version of simulated annealing for graph bisection. We used the following version of the greedy algorithm in our experiments. The algorithm has several passes, each pass tries to improve the result of the previous pass. The algorithm will stop when no more improvement can be made. The algorithm starts with a random bisection. At each step in one pass of the algorithm, two vertices will be chosen, one in each side of the bisection. These vertices are chosen in such a way that when they are interchanged they will yield the largest reduction in the size of the bisection. Ties are broken arbitrarily. For simplicity each vertex in the pair is chosen independently, i.e., each vertex is the best choice in each half but as a pair they may not be the best choice over all pairs. By doing this we reduce the running time significantly, and experience shows that it does not affect the performance of the algorithm very much. Once a vertex has been chosen to be exchanged it will not be chosen again. A pass is finished when there is no pair that will give a positive reduction in the size of the bisection.

Since the Kernighan-Lin and greedy algorithms were discovered to be sensitive to the choice of the initial bisection, we ran these algorithms several times for each graph (each time starting with a different random bisection). The data for these algorithms represent the bisections found for the three or five initial bisections tried for each graph.

We have included some of the data obtained in Tables I, II and III. The data for 3-regular graphs is in Table I. Tables II and III contain the data for 4-regular and 5-regular graphs, respectively. Because of the difficulty in generating random d-regular graphs without loops and multiple edges for $d>3$, only the graphs in Table

Table I. Data for simple graphs randomly selected from $\mathcal{G}(n, 3, b)$

$2n$	b	*Alg. 2*	*S.A.*	*Greedy*	*K.L.*
100	2	2	2	(18, 24, 42)	(18, 2, 22, 8, 2)
	2	2	2	(46, 48, 24)	(2, 2, 2, 2, 2)
	2	2	2	(14, 32, 38)	(22, 2, 2, 14, 2)
	6	6*	6	(40, 32, 18)	(10, 14, 18, 20, 10)
	6	6	6	(32, 24, 26)	(12, 6, 6, 24, 10)
	6	6	14	(26, 24, 18)	(18, 8, 8, 6, 6)
200	2	2	20	(88, 48, 76)	(2, 36, 34, 44, 2)
	2	2	2	(84, 66, 68)	(36, 2, 10, 2, 2)
	2	2	14	(88, 50, 40)	(2, 14, 44, 2, 2)
	10	10*	38	(80, 50, 38)	(38, 34, 24, 20, 22)
	10	10*	10	(48, 42, 58)	(18, 38, 28, 18, 12)
	10	10*	10	(52, 68, 54)	(32, 34, 20, 24, 26)
400	2	2	2	(102, 168, 110)	(2, 74, 74, 72, 50)
	2	2	2	(94, 156, 138)	(2, 28, 14, 72, 2)
	2	2	42	(130, 128, 164)	(2, 82, 84, 72, 2)
	10	10	26	(82, 96, 168)	(74, 68, 24)
	10	10*	50	(94, 180, 90)	(78, 64, 10)
	14	14*	60	(92, 168, 86)	(74, 80, 44)
	14	14*	76	(94, 90, 98)	(80, 46, 78)
	16	16	70	(180, 82, 86)	(72, 42, 24)
	16	16*	44	(90, 102, 84)	(70, 32, 82)
	18	18*	74	(170, 98, 96)	(74, 42, 58)
	18	18	54	(100, 148, 124)	(46, 72, 60)
800	2	2	104	(286, 186, 314)	(94, 2, 26)
	2	2	104	(346, 186, 332)	(78, 8, 138)
	10	10	116	(314, 174, 190)	(142, 22, 102)
	10	10	102	(346, 190, 194)	(146, 10, 158)
	20	18*	126	(188, 198, 174)	(22, 50, 46)
	20	20	138	(202, 180, 180)	(152, 144, 98)
	24	24*	50	(174, 312, 176)	(112, 90, 156)
	24	24*	126	(236, 240, 186)	(78, 150, 54)

Table II. Data for graphs randomly selected from $\mathcal{G}^*(n, 4, b)$

$2n$	b	*Alg. 2*	*S.A.*	*Greedy*	*K.L.*
100	2	2	2	(50, 44, 44)	(2, 2, 2)
	6	6	6	(36, 40, 36)	(6, 6, 6)
	12	12*	40	(52, 50, 50)	(12, 12, 12)
200	2	2	2	(86, 106, 74)	(2, 2, 2)
	10	10	10	(82, 88, 100)	(10, 10, 10)
	20	20*	20	(90, 106, 110)	(20, 20, 20)
400	2	2	2	(400, 378, 366)	(2, 2, 2)
	14	14	14	(208, 184, 190)	(14, 14, 14)
	20	20	20	(196, 174, 184)	(20, 20, 20)
	24	24*	58	(200, 218, 192)	(24, 24, 24)
800	2	2	30	(400, 378, 366)	(2, 2, 2)
	14	14	14	(384, 404, 374)	(14, 14, 14)
	28	28	28	(404, 362, 390)	(28, 28, 28)
	50	50*	80	(404, 354, 384)	(50, 50, 50)

Table III. Data for graphs randomly selected from $\mathcal{G}^*(n, 5, b)$

2n	b	Alg. 2	S.A.	Greedy	K.L.
100	2	2	2	(2, 40, 2)	(2, 2, 2)
	14	14	14	(50, 76, 64)	(14, 14, 14)
	22	22	26	(62, 74, 64)	(22, 22, 22)
200	2	2	2	(124, 158, 140)	(2, 2, 2)
	20	20	20	(120, 116, 142)	(20, 20, 20)
	32	32	32	(152, 80, 116)	(32, 32, 32)
400	2	2	2	(272, 2, 224)	(2, 2, 2)
	30	30*	30	(266, 234, 314)	(30, 30, 30)
	36	36	40	(214, 180, 218)	(36, 36, 36)
800	2	2	32	(430, 2, 2)	(2, 2, 2)
	50	50	50	(634, 622, 492)	(50, 50, 50)
	94	94	110	(532, 456, 378)	(94, 94, 94)

I are simple. Based on our experiments with 3-regular simple graphs and 3-regular multigraphs, the existence of loops and multiple edges has no bearing at all on the data.

Values of b in Tables I, II and III denote the b used in generating graphs from $\mathcal{G}(n, d, b)$ and $\mathcal{G}^*(n, d, b)$. For all but one graph, this value was also the bisection width of the graph generated. Values of b were chosen between 2 and the largest values for which Algorithm 2 still works. Algorithm 2 breaks down for large values of b primarily because the probabilistic lemmas proved in Sections 2 and 3 do not hold for large b.

5. Remarks

The data in Section 4 suggest several interesting conjectures. First, it appears as through the Kernighan-Lin and simulated annealing algorithms almost always get the optimal bisection when the node degree is large. In fact, it seems that even the greedy heuristic almost always finds the minimum bisection when the node degree is sufficiently large. We suspect that this can be proved by showing that there are very few (and possibly only one) locally optimal bisections in such graphs.

The observation that standard algorithms like greedy, Kernighan—Lin and simulated annealing work better for larger degree graphs suggests an interesting variation of the algorithms for small degree graphs, namely to: 1) randomly contract edges out of the graph to increase the average node degree, 2) run the algorithm to find a bisection of the contracted graph, 3) uncontract the edges to obtain the original graph (the bisection of the contracted graph found in step 2 might not be a bisection of the original graph, in that case randomly adjust that bisection to obtain a bisection for the original graph), and 4) run the algorithm again, starting with this new bisection. We have run such modified versions of the algorithms on many of the test graphs used to generate the data in Section 4. The results were surprisingly consistent: even for degree 3 graphs, the modified algorithms almost always produced the optimal bisection. We suspect that this observation can be proved using the techniques developed in

Section 2 and 3. (A similar strategy is also used by Goldberg and Burstein in [12] to upgrade the performance of bisection algorithms, although less dramatic results are observed.)

We also suspect that the results of this paper can be extended to hold for graphs with arbitrarily large node degrees (provided that graphs with multiple edges and loops are also considered), and for graphs with larger bisections (provided that we no longer require the algorithm to find exactly the minimum bisection). It is also likely that the results can be extended to hypergraphs and graphs with recursive decompositions, such as bifurcators or separators [4]. The main barriers to proving stronger generalizations are

(i) the fact that only $e^{-\Theta(d^2)}$ of all d-regular graphs are simple combined with the fact that simple graphs are hard to work with directly, and

(ii) the fact that almost all graphs in $\mathcal{G}^*(n, d, b)$ for $b = \Omega(n^{1-1/\lfloor (d+1)/2 \rfloor})$ have bisection width less than b.

It would be nice to extend the results to classes of random graphs generated by inserting edges to achieve average node degree d, as opposed to uniform node degree d. Such graphs are far more common in empirical studies (e.g., see [12]) since they are very easy to generate and to work with mathematically. Graphs generated in this fashion suffer a serious drawback, however, since many of the nodes in such graphs have degree zero whenever the number of edges is linear in the number of nodes. In addition, it is not clear how to generate such graphs with known bisection uniformly, since minimum bisections for such graphs are usually nonunique, even if they have the same cardinality as the implanted bisection. Although the algorithms developed in the paper can probably be modified to work well for such graphs empirically, proving that they work optimally almost all the time may be very difficult.

Although theoretical extensions of the proofs in Sections 2 and 3 to include graphs with larger degrees and bisections may be difficult, it would at least be nice to extend the algorithms so that they worked on such graphs in practice. We don't yet know how to do this, although the flow-based techniques are very good at identifying unusually small cuts from one subset of nodes to the remaining nodes.

Lastly, it would be of great interest to find good approximation algorithms for the graph bisection problem. We suspect that flow-based techniques can be of some help in this respect but have not been able to prove it.

Acknowledgements. We gratefully acknowledge several helpful discussions of this work with Charles Leiserson and Gary Miller. We are also indebted to Christopher Heigham for providing some of the data in Section 4.

References

[1] B. BAKER, Approximation Algorithms for NP-complete Problems on Planar Graphs, *FOCS* 1983, 265—273.

[2] E. R. BARNES, An Algorithm for Partitioning the Nodes of a Graph, *IBM Technical Report* RC8690, 1981.

[3] E. R. BARNES and A. J. HOFFMAN, Partitioning, Spectra and Linear Programming, *IBM Research Report*, (unknown date).

[4] S. N. BHATT and F. T. LEIGHTON, A Framework for Solving VLSI Graph Layout Problems, *Journal of Computer and System Sciences*, **28** (1984).

[5] B. BOLLOBÁS, A Probabilistic Proof of an Asymptotic Formula for the Number of Labelled Regular Graphs, *European J. Combinatorics*, **1** (1980), 311—316.
[6] B. BOLLOBÁS and W. FERNANDEZ DE LA VEGA, The Diameter of Random Regular Graphs, *Combinatorica*, **2** (1982), 125—134.
[7] M. A. BREUER, Min-cut Placement, *J. Design Aut. and Fault Tol. Comp.*, **1** (1977), 343—362.
[8] W. E. DONATH and A. J. HOFFMAN, Lower Bounds for the Partitioning of Graphs, *IBM J. Res. Develop.*, **17** (1973), 420—425.
[9] C. M. FIDUCCIA and R. M. MATTHEYSES, A Linear-Time Heuristic for Improving Network Partitions, *Proceedings of the* **19***th Design Automation Conference, IEEE Computer Society Press*. 1982, 175—181.
[10] M. R. GAREY and D. S. JOHNSON, *Computers and Intractibility: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.
[11] M. R. GAREY, D. S. JOHNSON and L. STOCKMEYER, Some Simplified NP-complete Graph Problems, *Theoretical Computer Science*, **1** (1976), 237—267.
[12] M. K. GOLDBERG and M. BURSTEIN, Heuristic Improvement Techniques for Bisection of VLSI Networks, *unpublished manuscript*, 1984, Clarkson University.
[13] M. K. GOLDBERG and R. GARDNER, On the Minimal Cut Problem, *in: Progress in Graph Theory*, (ed: J. A. Bondy and U.S.R. Murty), Academic Press, 1984, 295—305.
[14] D. S. JOHNSON, *personal communication*.
[15] B. W. KERNIGHAN and S. LIN, An Efficient Heuristic Procedure for Partitioning Graphs, *The Bell System Tech. J.*, **49** (1970), 291—307.
[16] S. KIRKPATRICK, C. D. GELATT, Jr. and M. P. VECCHI, Optimization by Simulated Annealing, *IBM Research Report RC* **9355**, April 1982, Yorktown Heights, New York.
[17] R. J. LIPTON and R. E. TARJAN, Applications of a Planar Separator Theorem, *Proceedings of the* **18***th Symposium on the Foundation of Computer Science*, (1977), 162—170.
[18] R. J. LIPTON and R. E. TARJAN, A Separator Theorem for Planar Graphs, *SIAM J. Computing*, **36** (1979), 177—189.
[19] R. M. MACGREGOR, On Partitioning a Graph: A Theoretical and Empirical Study, *Ph. D. Thesis, University of California*, Berkeley, 1978, also appeared as Technical Report UCB/ERL M78/14.
[20] R. L. RIVEST, The "PI" (Placement and Interconnect) System, *Proceedings of the* **19***th Annual Design Automation Conference*, IEEE 1982, 475—481, *Computer Society Press*.

Thang Nguyen Bui
EECS Dept.
Mass. Inst. of Tech.
Cambridge, Mass. 02139
U.S.A.

current address:

Comp. Sci. Dept.
Penn State University
University Park, PA 16802
U.S.A.

Soma Chaudhuri
Math. Dept.
Mass. Inst. of Tech.
Cambridge, Mass. 02139
U.S.A.

current address:

Comp. Sci. Dept.
University of Washington
Seattle, Washington
U.S.A.

F. Thomson Leighton
Math. Dept. and Lab. for Comp. Sci.
Mass. Inst. of Tech.
Cambridge, Mass. 02139
U.S.A.

Michael Sipser
Math. Dept. and Lab. for Comp. Sci.
Mass. Inst. of Tech.
Cambridge, Mass. 02139
U.S.A.