Short Paths in Expander Graphs*

Jon Kleinberg[†] Ronitt Rubinfeld[‡]

Abstract

Graph expansion has proved to be a powerful general tool for analyzing the behavior of routing algorithms and the inter–connection networks on which they run. We develop new routing algorithms and structural results for bounded–degree expander graphs. Our results are unified by the fact that they are all based upon, and extend, a body of work asserting that expanders are rich in short, disjoint paths. In particular, our work has consequences for the disjoint paths problem, multicommodity flow, and graph minor containment. We show:

(i) A greedy algorithm for approximating the maximum disjoint paths problem achieves a polylogarithmic approximation ratio in bounded–degree expanders. Although our algorithm is both *deterministic* and *on-line*, its performance guarantee is an improvement over previous bounds in expanders.

(ii) For a multicommodity flow problem with arbitrary demands on a bounded–degree expander, there is a $(1 + \varepsilon)$–optimal solution using only flow paths of polylogarithmic length. It follows that the multicommodity flow algorithm of Awerbuch and Leighton runs in nearly *linear* time per commodity in expanders. Our analysis is based on establishing the following: given edge weights on an expander G, one can increase some of the weights very slightly so the resulting shortest-path metric is *smooth* – the min-weight path between any pair of nodes uses a polylogarithmic number of edges.

(iii) Every bounded–degree expander on n nodes contains every graph with $O(n/\log^{O(1)} n)$ nodes and edges as a minor.

1 Introduction

The concept of *graph expansion* has proved to be a powerful general tool for analyzing the behavior of routing algorithms and the inter–connection networks on which they run. The

*A preliminary version of this work appeared at the 1996 IEEE Symposium on Foundations of Computer Science.

[†]Department of Computer Science, Cornell University, Ithaca NY 14853. Email: kleinber@cs.cornell.edu. Supported in part by a David and Lucile Packard Foundation Fellowship, an Alfred P. Sloan Research Fellowship, an ONR Young Investigator Award, and NSF Faculty Early Career Development Award CCR-9701399. This work was conducted primarily while the author was a student in the MIT Laboratory for Computer Science, supported by an ONR Graduate Fellowship.

[‡]NEC Research Institute, Princeton, NJ 08540. Research conducted while at the Department of Computer Science, Cornell University and while visiting the MIT Lab for Computer Science under an NSF VPW award. This work is partially supported by ONR Young Investigator Award N00014-93-1-0590, the Alfred P. Sloan Research Award, and grant No. 92-00226 from the United States - Israel Binational Science Foundation (BSF), Jerusalem, Israel. Email: ronitt@research.nj.nec.com.

development of many fundamental routing algorithms for hypercube–derived networks, for example, has been based heavily on the good expansion properties of such graphs [33, 3, 21, 20]. As such, the development of new techniques for working with expander graphs, and with the consequences of expansion at a general level, is of value both from a basic algorithmic point of view and within the context of network routing.

In this paper, we develop a number of new routing algorithms and structural results for bounded–degree expander graphs. Our results are unified by the fact that they are all based upon, and extend, a body of work asserting that expanders are rich in short, disjoint paths. Let us first discuss some of this work, and then outline how our results fit into this framework.

For the sake of concreteness, we say that a graph $G = (V, E)$ is an *α–expander* if for every set X of at most half the vertices, we have $|\delta(X)| \geq \alpha|X|$; here $\delta(X)$ denotes the set of edges with one end in X and the other end in $V \setminus X$. We adopt the following notational device throughout this paper: we fix a natural number $\Delta \geq 3$ and a real number $\alpha > 0$; we say that G is *bounded–degree* if its degree is at most Δ, and an *expander* if it is an α-expander. All of our results in this paper will hold for arbitrary Δ and α; however, some of the previous work we discuss requires much stronger expansion guarantees, and we will indicate this when it is relevant.

Many of the previous results on disjoint paths in expanders can be roughly classified into two groups: those based on random walks, and those based on multicommodity flow.

• Using random walks in expanders, Broder, Frieze, and Upfal [12] showed that there is a constant θ so that in n–node graphs with strong expansion guarantees, any set of up to $O(n/\log^{\theta} n)$ pairs of nodes can be connected on edge–disjoint paths; this strengthened earlier work of Peleg and Upfal [26]. Random walk techniques were subsequently employed in a randomized on–line algorithm of Raghavan and Upfal [29]. They showed that in any bounded–degree expander graph, any set of terminal pairs in which each node appears at most once can be partitioned into $O(\log^2 n)$ subsets, each of which is routable on edge–disjoint paths. These techniques also provide a randomized on–line $O(\log^2 n)$–approximation for the maximum disjoint paths problem in any bounded–degree expander.

• The analysis of expansion properties via multicommodity flow was initiated by Leighton and Rao [23]. Among many other results, they showed that given any bounded–degree n-node expander G, any n–node bounded degree graph H can be embedded in G with congestion and dilation $O(\log n)$. That is, nodes of H are mapped bijectively to nodes of G, and edges of H realized as paths of length $O(\log n)$ in G. Aumann and Rabani [4] used this theorem to obtain the "partitioning" result of Raghavan and Upfal via a deterministic off–line algorithm.

It is also worth noting here that the recent multicommodity flow algorithm of Awerbuch and Leighton [7] has a running time that is heavily dependent on the length of the longest path used in a (nearly) optimal multicommodity flow. Thus, one would expect the algorithm to perform well in bounded–degree expander graphs. But despite the above work, there have not been results asserting that a $(1 + \varepsilon)$–approximately optimal multicommodity flow in a bounded–degree expander graph can be achieved using only "short" paths. In this paper we obtain such a result, with consequences for the analysis of the Awerbuch–Leighton algorithm.

Thus, in this paper, we prove three main results within the framework of the work just described.

The Greedy Algorithm for Disjoint Paths. The maximum disjoint paths problem (MDP) is this: given a graph G, and a set $\mathcal{T}$ of pairs of vertices in G, connect as many pairs in $\mathcal{T}$ as possible using edge–disjoint paths. (We call $\mathcal{T}$ the set of *terminal pairs.*) Probably the simplest approximation algorithm for the MDP is the *greedy algorithm* — proceed through the terminal pairs in sequence, routing each one if possible on the shortest free path. We investigate a closely related algorithm, the *bounded greedy algorithm* (BGA), which only uses a free path if it is "sufficiently" short. The BGA was first analyzed as one component of an on–line algorithm of the first author and Éva Tardos [19]; in that work, some (necessarily weak) bounds on its performance in general graphs were given. Specifically, the BGA is a deterministic, on–line algorithm, and hence lower bounds of Awerbuch et al. [6] and Blum et al. [10] imply that it cannot achieve a polylogarithmic approximation ratio in graphs such as trees and meshes.

We show here that the BGA is an $O(\log n \log\log n)$–approximation in bounded–degree expanders. Thus the BGA gives what is currently the best approximation ratio for the MDP in bounded–degree expanders; as noted above, the techniques of Raghavan and Upfal [29] and Aumann and Rabani [4] provide $O(\log^2 n)$–approximations. Our result also identifies, to our knowledge, the first non–trivial class of graphs in which deterministic on–line algorithms can achieve a polylogarithmic approximation ratio for the MDP; this is in contrast to the lower bounds of [6, 10], as well as the $\Omega(n^{\varepsilon})$ lower bound of Bartal, Fiat, and Leonardi that holds even for *randomized* on–line algorithms in certain graphs [8]. There is a natural extension of the BGA, and its analysis, which shows that it achieves the same approximation ratio for the more general problem of packing trees of bounded size in the on–line model.

Multicommodity Flow on Short Paths. The *multicommodity flow problem* is a fractional relaxation of the disjoint paths problem. We are given a graph G and a set $\mathcal{T}$ of terminal pairs, such that each terminal pair (u, v) has associated with it a positive *demand* ρ_{uv}. We will also refer to the terminal pair (u, v) as a *commodity.* The goal is to find the smallest ν such that there is a fractional flow routing ρ_{uv} units of demand between each commodity (u, v), and no more than ν units of flow cross any edge. We will refer to ν as the *congestion* of the flow, and denote the minimum congestion of any flow for $(G, \mathcal{T})$ by $\nu^*(G, \mathcal{T})$.

We prove that in any bounded–degree expander graph G, and for a multicommodity flow problem on G with arbitrary demands, there is a flow of congestion at most $(1+\varepsilon)$ times optimal that only uses flow paths of length at most $O(\varepsilon^{-1}\log^3 n)$. Our proof (see Section 4) is based on examining the linear programming dual of the multicommodity flow problem — within the context of the dual, we are faced with the following combinatorial problem. We are given edge weights on G; and we say that the shortest–path metric of G with respect to these weights is *smooth* if the minimum–weight path between any pair of nodes uses only a polylogarithmic number of edges. Now, an arbitrary shortest–path metric certainly need not be smooth; however, we show that if G is an expander, one can increase some of the edge weights very slightly, so that the resulting metric *is* smooth. By LP duality, this will imply our result on multicommodity flow on short paths.

Our result has the following consequence. The recent multicommodity flow algorithm of Awerbuch and Leighton [7] provides a $(1+\varepsilon)$–approximate guarantee and has running time

$O^*(d^2km)$, on a graph with m edges and k commodities, and where d is the maximum length of a flow path in the optimal solution. (Here and in what follows, we use $O^*()$ to suppress terms that are polynomial in $\log n$ and ε^{-1}.) However, it is in fact sufficient for their analysis that d be the maximum length of a flow path in any $(1 + \frac{1}{2}\varepsilon)$–approximate solution, and hence our result shows that in a bounded–degree expander graph, the Awerbuch–Leighton algorithm runs in time $O^*(km)$. This is nearly *linear* time per commodity; it is thus a significant improvement over previous running times [22] for the multicommodity flow problem on expanders.

Graph Minors. Finally, we show how the random–walk technique of Broder, Frieze, and Upfal [12], which produces short, disjoint paths in expanders, can be used to obtain results on graph minor containment. We say that a graph H is a *minor* of G if H can obtained from a subgraph of G by contracting edges; in this case, we say that G contains an H-minor. Minors appear in Kuratowski's famous theorem (see e.g. [11]), which characterizes planar graphs as precisely those which do not contain K_5 or $K_{3,3}$ as a minor; the exclusion of minors has since been used as a nice way to extend results for planar graphs to more general classes [2, 30], and as the basis for some very deep results in both structural and algorithmic graph theory [31, 32].

We show the following. For every $\alpha > 0$ there is a number $\kappa > 0$ such that the following holds: Every bounded–degree α–expander graph G on n nodes contains *every* graph H with $O(n/\log^\kappa n)$ nodes and edges as a minor. Moreover, we give a randomized polynomial–time algorithm, similar to that of [12], which explicitly finds a contraction of the expander G to the smaller graph H.

This result has the following immediate corollary. If G is a bounded–degree expander graph, and P is the graph consisting simply of a path on $O(n/\log^\kappa n)$ nodes, then our result implies that G contains a P-minor, and hence G has a long simple path. The existence of long simple paths in expanders has been a problem that has received some amount of attention; for example, a result of Pósa [28] asserts the following. If a graph G has the property that every set S of vertices of size at most k has at least $2|S| - 1$ neighboring vertices, then G contains a simple path of length k. A short proof of this is due to Lovász [25, Ch. 10]. This result was generalized by Friedman and Pippenger [14] to show that if every set S of size at most $2k - 2$ has at least $(d + 1)|S|$ neighboring vertices, then G contains every k-node tree with maximum degree d.

However, these results [28, 25, 14] require the stronger notion of *node expansion* (rather than edge expansion), and their proofs seem to inherently require an expansion factor of at least 2. On the other hand, our result, which is weaker by a polylogarithmic factor, holds in a graph with edge expansion α, for any constant $\alpha > 0$.

Our result is also related to work of Alon, Seymour, and Thomas [2] and Plotkin, Rao, and Smith [27] on the existence of large clique minors in graphs without small separators. The second of these papers shows that any bounded-degree expander contains a complete graph on $\Omega(\sqrt{n/\log n})$ nodes as a minor. This is a somewhat larger bound for clique minors than is provided by our result, which only guarantees cliques of size $\Omega(\sqrt{n}/\log^{O(1)} n)$ as minors. However, [2, 27] do not show that expanders contain *every* graph with $O(n/\log^{O(1)} n)$ nodes and edges as a minor. (It is worth noting that in order for a bounded-degree graph to contain every graph with k nodes and edges as a minor, it is not sufficient that it contain a clique minor on $\Theta^*(\sqrt{k})$ nodes. As an example, let G denote the graph obtained from the $\sqrt{n} \times \sqrt{n}$ grid by adding edges between each pair of nodes with a common neighbor, and let H denote a 3-regular expander on n^δ nodes, $\frac{2}{3} < \delta < 1$. Then G contains a clique on $\Theta(\sqrt{n})$ nodes as a minor, but it has no H-minor.)

A Result on Network Flow in Expanders. For the analysis in Section 3, we require a theorem on (single-commodity) network flows in expander graphs that appears to be novel and of interest in its own right. To understand the context, consider that in any bounded-degree graph G of expansion $\alpha > 1$, the max-flow min-cut theorem directly implies that any two subsets of equal cardinality, each containing at most half the vertices, can be linked by edge-disjoint paths. Our lemma asserts, essentially, that if the expansion α is a constant bounded above 1, then two such sets can be linked by edge-disjoint paths, each of length $O(\log n)$.

As a final comment, we note that a common element of all our results is that they hold for arbitrarily small constant values of the edge expansion $\alpha > 0$. For the sake of simplicity, we have not attempted to parametrize the bounds we obtain in terms of α, though this is not difficult to do.

2 A Result on Network Flow in Expanders

We begin with the following result about edge-disjoint paths linking two sets of nodes in a bounded-degree expander graph.

Theorem 2.1 *Let G be a multigraph whose edge expansion is at least 2 and whose maximum degree is a constant d. Let X and Y be sets of nodes in G of cardinality $r < n/2$. Then X can be linked to Y by edge-disjoint paths each of length $O(\log n)$.*

Proof. Let us attach a new node s to the nodes of X and a new node t to the nodes of Y. By the max-flow min-cut theorem, there is a feasible s-t flow of value r. Now, assigning a cost of 1 to each edge, we consider a minimum-cost X-Y flow of value r. By the integrality theorem for minimum-cost flow, this flow consists of r edge-disjoint s-t paths, each passing through a distinct node from X and Y. We claim that each of these paths has length $O(\log n)$.

To prove this, consider an optimal dual solution to the minimum-cost flow problem. It assigns an integer y_v to each vertex v, such that the following properties hold.

(1) If flow travels on an edge (u, v) from u to v, then $y_v - y_u \geq 1$,

(2) If no flow travels on an edge (u, v), then y_v and y_u differ by at most 1.

(3) We may assume without loss of generality that $y_s = 0$, $y_t \geq 0$, and for every j in the interval $[0, y_t]$, there exists a node v with $y_v = j$.

Let V_i denote the set of nodes $u \neq s$ with $y_u \leq i$. We claim that $|V_i| \geq (1 + 1/d)^{i-1}$, as long as $|V_{i-1}| \leq n/2$. (That is, the sets V_i grow exponentially in size.) This is true for $i = 1$ by fact (3) above – one choose the optimal dual solution in such a way that at least one vertex $v \neq s$ has $y_v \leq 1$. Now suppose that V_i has cardinality $a \leq n/2$. By the edge expansion of G, at least $2a$ edges leave the set V_i. Now, by property (1) of the dual solution, every flow path leaves V_i at most once; and since $|X \cap V_i| \leq |V_i| = a$, at most a flow paths pass through V_i at all. Thus, at least $2a - a = a$ of the edges leaving V_i carry no flow. By property (ii) of the dual solution, all these edges must go to nodes in $V_{i+1} - V_i$, and hence $V_{i+1} - V_i$ contains at least a/d nodes. Thus $|V_{i+1}| \geq (1 + 1/d)|V_i|$, and the claim follows.

It follows that for a number $q = O(d \log n)$, the set V_q contains more than half the vertices of G. That is, more than half the vertices v of G satisfy $y_v \leq O(d \log n)$. We now apply the same argument beginning at the node t instead of the node s, and conclude that more than half the vertices v of G satisfy $y_t - y_v = O(d \log n)$.

Thus we have $y_t - y_s = y_t = O(d \log n) = O(\log n)$. By property (1) of the optimal dual solution, each flow path passes through a sequence of nodes of strictly increasing y-values; thus each flow path is of length $O(\log n)$. ■

In Section 3, we combine this with the following result of Broder, Frizeze, and Upfal [13]. We say that a set $\mathcal{T}$ of terminal pairs is a *permutation* if each node of G appears in exactly one terminal pair; we say $\mathcal{T}$ is a *partial permutation* if each node appears in at most one terminal pair. We define the *congestion* of a set of paths in G to be the maximum number of paths that cross any single edge in G.

Theorem 2.2 (Broder-Frieze-Upfal) *There are constants c_1 and c_2 such that any partial permutation in G of size at most $c_1 n/\log n$ can be routed with congestion at most $c_2 \log \log n$.*

For our purposes in Section 3, we specifically wish to strengthen this theorem with the additional conclusion that all the paths in the routing are *short*. We use Theorem 2.1 to do this as follows.

Corollary 2.3 *There are constants c_1, c_2, and c_3 such that any partial permutation in G of size at most $c_1 n/\log n$ can be routed with congestion at most $c_2 \log \log n$ using paths of length at most $c_3 \log n$.*

Proof. We first consider how the paths in the proof of Theorem 2.2 are constructed. Let $k = c_1 n/\log n$, and let the partial permutation consist of pairs $\{(s_i, t_i) : i = 1, 2, \ldots, k\}$. Write $S = \{s_1, \ldots, s_k\}$ and $T = \{t_1, \ldots, t_k\}$.

In [13], Theorem 2.2 is proved by an algorithm that can be viewed as follows. It first replicates each of G's edges $2/\alpha$ times, obtaining a multigraph G'. It then selects two sets of random nodes A and B, each of size k. It finds edge-disjoint paths in G' linking S to A and T to B; we will refer to these sets of paths as *linkages* from S to A and from T to B. Write $A = \{a_1, \ldots, a_k\}$, indexed so that the path in the S-A linkage with one end equal to s_i has its other end equal to a_i. We index B analogously. The algorithm concludes by constructing a low-congestion set of paths in G', each of length $O(\log n)$, from each $a_i \in A$ to the corresponding $b_i \in B$. The superposition of all these portions of paths gives the routing for $\{(s_i, t_i)\}$.

Thus, to obtain the result in the corollary, we need only show that the S-A and T-B linkages can be constructed so that all their paths have length $O(\log n)$. But this follows directly from Theorem 2.1, since $|A|, |B|, |S|, |T| < n/2$, and G' is a multigraph of edge expansion at least 2 and maximum degree $2\Delta/\alpha = O(1)$. ■

3 The Greedy Algorithm for Disjoint Paths

We will be considering the *bounded greedy algorithm* (BGA) for the maximum disjoint paths problem. The BGA is a deterministic on–line algorithm that works as follows: as each terminal pair arrives, it routes on any free path in G that is sufficiently short, and rejects the pair if all free paths are too long. More precisely, the BGA is defined by an implicit paramter L and runs as follows.

(i) Proceed through the terminal pairs in one pass.

(ii) When (s_i, t_i) is considered, check whether s_i and t_i can be joined by a path of length of at most L. If so, route (s_i, t_i) on such a path P_i. Delete P_i from G and iterate.

As we mentioned above, lower bounds of [6, 10] show that *no* deterministic on–line algorithm can obtain a polylogarithmic approximation ratio when the underlying graph is a tree or a mesh. Thus, this applies to the BGA as well. What we show here is that the BGA is in fact an $O(\log n \log\log n)$–approximation in any bounded–degree expander graph.

We say that a set $\mathcal{T}$ of terminal pairs is a *permutation* if each node of G appears in exactly one terminal pair; we say $\mathcal{T}$ is a *partial permutation* if each node appears in at most one terminal pair. We define the *congestion* of a set of paths in G to be the maximum number of paths that cross any single edge in G. Our analysis of the BGA proceeds roughly as follows. Consider the problem defined by G and $\mathcal{T}$. We know, by Corollary 2.3, that there is a set of short paths, with low congestion, connecting a large fraction of the terminal pairs in $\mathcal{T}$. Since these paths are all short, the BGA could have used them if they were free; hence, if the terminal pairs associated with one of these short paths P is not routed by the BGA, the path P must meet a path constructed earlier by the BGA. Thus, we can charge unused short paths to paths constructed by the BGA, and conclude that the BGA must have constructed a lot of paths.

Recall the constants c_1, c_2, and c_3 from Corollary 2.3. We define $c' = \max(c_1^{-1}\Delta, c_3)$.

Theorem 3.1 *The* BGA *with parameter $L = c'\Delta \log n$ is an $O(\log n \log\log n)$–approximation for the* MDP *in G.*

Proof. Let $\mathcal{B}$ denote the set of paths routed by the BGA when run on G with terminal pairs $\mathcal{T}$, and let $\mathcal{O}$ denote a set of edge–disjoint paths of maximum cardinality. Let $\mathcal{O}' \subset \mathcal{O}$ denote the set of paths in $\mathcal{O}$ corresponding to terminal pairs not routed by the BGA. Since the paths in $\mathcal{O}'$ are edge-disjoint, at most Δ of the pairs routed by $\mathcal{O}'$ can have $v \in G$ as an endpoint. Thus, by Vizing's Theorem (cf. [11]), the pairs routed by $\mathcal{O}'$ can be partitioned into at most $\Delta + 1$ partial permutations; let $\mathcal{O}'_1 \subset \mathcal{O}'$ denote the set of paths corresponding to the largest of these partial permutations.

We now demonstrate the existence of a set of paths $\mathcal{O}''_1$ of congestion at most $c_2 \log\log n$ that routes at least half as many pairs as $\mathcal{O}'_1$ and uses only paths of length at most $c' \log n$.

(i) If $|\mathcal{O}'_1| > c_1 n/\log n$, then by the pigeonhole principle, at least half of the paths in $\mathcal{O}'_1$ have length at most $|E(G)|/(\frac{1}{2}|\mathcal{O}'_1|) \le c_1^{-1}\Delta \log n$. We define $\mathcal{O}''_1$ to be this set of paths.

(ii) If $|\mathcal{O}'_1| \le c_1 n/\log n$, then by the theorem of Broder, Frieze, and Upfal [13], the pairs routed by $\mathcal{O}'_1$ can instead be routed on paths of length at most $c_3 \log n$ and congestion at most $c_2 \log\log n$. We define $\mathcal{O}''_1$ to be this set of paths.

Now we relate the size of $\mathcal{O}_1''$ to the size of $\mathcal{B}$. First, we claim that every path in $\mathcal{O}_1''$ intersects some path in $\mathcal{B}$. For suppose that $Q \in \mathcal{O}_1''$ is disjoint from all the paths in $\mathcal{B}$. Then the BGA with parameter L would have routed the endpoints of $\mathcal{O}_1''$ on the path Q — a contradiction.

Given this, we charge each path in $\mathcal{O}_1''$ to some path in $\mathcal{B}$ that it intersects. Since the paths in $\mathcal{O}_1''$ have congestion at most $c_2 \log\log n$, and the paths in $\mathcal{B}$ have length at most $L = c'\Delta \log n$, each path in $\mathcal{B}$ is charged at most $c_2 c' \Delta \log n \log\log n$ times. Thus we have

$$\begin{aligned} |\mathcal{O}'| &\leq (\Delta+1)\cdot|\mathcal{O}_1'| \\ &\leq 2(\Delta+1)\cdot|\mathcal{O}_1''| \\ &\leq 2c_2c'\Delta(\Delta+1)\log n \log\log n \cdot |\mathcal{B}|. \end{aligned}$$

Since by the definition of $\mathcal{O}'$ we also have $|\mathcal{O}\setminus\mathcal{O}'| \leq |\mathcal{B}|$, the result follows. ∎

We can obtain a slight strengthening of Theorem 3.1 in the case in which $\mathcal{T}$ is a permutation. It is quite similar to the proof of Theorem 3.1, but relies on a theorem of Leighton and Rao [23] which in our case can be phrased as follows: There exist constants c_4 and c_5 such that every permutation in G can be routed with congestion at most $c_4 \log n$, using paths of length at most $c_5 \log n$.

Theorem 3.2 *The* BGA *with parameter $L = c_5 \log n$ is an $O(\log n)$–approximation for the* MDP *in G, when the set of terminal pairs is restricted to be a permutation.*

Proof. We define $\mathcal{B}$, $\mathcal{O}$, and $\mathcal{O}'$ as before. Let $\mathcal{T}'$ denote the partial permutation consisting of pairs not routed by the BGA. We define $\mathcal{Q}$ to be a set of paths, one for each pair in the partial permutation $\mathcal{T}'$, such that each path in $\mathcal{Q}$ has length at most $c_5 \log n$ and the set $\mathcal{Q}$ has congestion at most $c_4 \log n$. As in the previous proof, we observe that each path in $\mathcal{Q}$ must intersect some path in $\mathcal{B}$; thus, charging paths in $\mathcal{Q}$ to paths in $\mathcal{B}$, we have

$$|\mathcal{T}'| = |\mathcal{Q}| \leq c_4 c_5 \log^2 n |\mathcal{B}|.$$

Since we also have $|\mathcal{T}\setminus\mathcal{T}'| \leq |\mathcal{B}|$, it follows that

$$|\mathcal{B}| \geq \frac{|T|}{1 + c_4c_5\log^2 n} = \frac{\frac{1}{2}n}{1 + c_4c_5\log^2 n}. \tag{1}$$

Now, if

$$|\mathcal{O}'| \leq \frac{|E|}{\frac{1}{2}c_5\log n} = \frac{\Delta n}{c_5 \log n},$$

then by Equation (1), we have $|\mathcal{B}| \leq O(\log n)\cdot|\mathcal{O}|$, as desired. Otherwise, by the pigeonhole principle, there is a set $\mathcal{O}'' \subset \mathcal{O}'$ of size at least $\frac{1}{2}|\mathcal{O}'|$, consisting of paths of length at most $c_5 \log n$. Each such path must meet a path in $\mathcal{B}$; charging paths in $\mathcal{O}''$ to paths in $\mathcal{B}$ as in the previous proof, we have $|\mathcal{O}''| \leq c_5 \log n |\mathcal{B}|$. The bound now follows in this case as well, since

$$\begin{aligned} |\mathcal{O}| &\leq |\mathcal{O}'| + |\mathcal{O}\setminus\mathcal{O}'| \\ &\leq 2|\mathcal{O}''| + |\mathcal{B}| \\ &\leq (2c_5\log n + 1)|\mathcal{B}| \end{aligned}$$

■

In a similar fashion, we can also analyze the performance of the greedy algorithm for the problem of packing *trees* in G on a bounded number of leaves. Here the BGA, with parameter L, is defined as follows. A single request now consists of a set S of t vertices in G, which must be connected into a tree. The BGA connects S if there is spanning tree on S using at most $(t-1)L$ edges. By viewing each tree as a union of paths, and using Corollary 2.3, one can show the following. The proof is strictly analogous to that of Theorem 3.1.

Theorem 3.3 *Let t be a constant, and suppose that each request consists of at most t vertices. Then the* BGA *with parameter $\Theta(\log n)$ is an $O(\log n \log\log n)$ approximation for the tree-packing problem in G.*

It is not difficult to show that any deterministic on-line algorithm for this problem must have a performance guarantee that depends at least linearly on t, the maximum number of vertices in a single request.

4 Smooth Metrics and Multicommodity Flow on Short Paths

We now turn to the *fractional* version of the disjoint paths problem, namely, the multicommodity flow problem. As mentioned in the introduction, our goal is to show that in a bounded–degree expander graph, there is a near–optimal multicommodity flow that only uses flow paths of polylogarithmic length. To do this, we first focus on a natural combinatorial problem that arises in the analysis of the linear programming dual of the multicommodity flow problem. Since this combinatorial problem appears to be interesting in its own right, we develop it first below, and then indicate the application to the flow problem.

Let $G = (V, E)$ be a graph on n nodes. We assign a non–negative weight ℓ_e to each edge e of G, in such a way that $\sum_e \ell_e = 1$. This turns G into a finite metric space (G, ℓ), under the shortest–path metric with respect to ℓ. If P is a path in G, we use $\ell(P)$ to denote the total weight of P, with respect to ℓ; for $u, v \in V$, we use $\ell(u, v)$ to denote the minimum weight of a u-v path in G. Note that one can also talk about the fewest *number* of edges in a u-v path; we will use $d(u, v)$ to denote this. Let $B_r(v)$ denote the ball of radius r about v; that is, the set of $u \in V$ with $d(u, v) \le r$.

Definition 4.1 *(G, ℓ) is said to be s–smooth if for every $u, v \in V$, there is a minimum–weight u–v path using at most s edges.*

Our interest here is the case in which G is a bounded–degree expander. It is not necessarily the case that (G, ℓ) is $o(n)$–smooth, when ℓ is an arbitrary set of non–negative weights. For example, if G is Hamiltonian, then we can assign weight 0 to the edges of a Hamiltonian path in G, and positive weight to all other edges. The resulting metric spaces contains minimum–weight paths that use $\Omega(n)$ edges.

Our goal, however, is to show that for any edge–weighted expander G, we can increase the edge weights very slightly and produce a metric space that is $O(\log^3 n)$–smooth. In particular, we prove the following.

Theorem 4.2 *Let (G, ℓ) be an edge–weighted bounded–degree expander. For every $\varepsilon > 0$, there exists a set of weights $\{\ell'_e\}$ such that the following are satisfied.*

(i) For all edges e of G, $\ell'_e \geq \ell_e$.

(ii) $\sum_e \ell'_e \leq 1 + \varepsilon$.

(iii) (G, ℓ') is $O(\varepsilon^{-1} \log^3 n)$-smooth.

Proof. The proof consists of an algorithm to produce the "smooth" weights $\{\ell'_e\}$. As the algorithm contains a number of details, it is worth outlining the underlying ideas at the outset. At any given point in the execution of the algorithm, we have a current threshold $\gamma \in (0, 1)$. An edge e is "heavy" if its weight ℓ_e is at least γ, and "light" otherwise. Let G' denote the subgraph of G consisting of the light edges. We use a technique of Awerbuch [5] to partition the vertex set of the graph into pieces $X_1, \ldots, X_q$, such that each X_i has diameter $O(\log n)$, and $|\delta_{G'}(X_i)| \leq \frac{1}{4}\alpha|X_i|$. For all X_i but the largest, the expansion of G implies that $\delta_G(X_i)$ has at least $\alpha|X_i|$ edges; so it follows that at least $\Omega(\alpha|X_i|)$ of the edges in $\delta_G(X_i)$ are heavy. We now add $O(\gamma/\log n)$ to the weight of every edge incident to a vertex in X_i; by charging these to the large number of heavy edges in $\delta_G(X_i)$, we can argue that not too much total weight is added.

Now, when these iterations come to an end, why is the resulting metric space smooth? First of all, we may assume that all edge weights are at least $\Omega(\frac{\varepsilon}{n})$. Let P be a shortest u-v path, and let e and f be two edges of roughly (to within a factor of two) the same weight γ. We consider the last iteration in which they both belonged to the largest of the sets X_i; in this iteration, there was a path of $O(\log n)$ edges between them, and each of the edges on this path had weight $O(\gamma \log n)$. Over all the remaining iterations, the weights of these edges do not grow by more than a constant factor, and so there is a path from e to f in the final metric space of weight $O(\gamma \log^2 n)$. It will follow that P cannot contain more than $O(\log^2 n)$ edges whose weights are mutually within a factor of two of one another, and hence by the pigeonhole principle P contains $O(\log^3 n)$ edges.

Let us now present the proof at a detailed level. We choose constants ε_1 and ε_2 that are sufficiently small relative to ε. In particular, $\varepsilon_1 < \Delta^{-1}\varepsilon$ and

$$\varepsilon_2 < \frac{1}{4}\alpha\Delta^{-1}\ln(1 + \frac{\varepsilon}{4}) \cdot \frac{\log n}{\log(\varepsilon^{-1}n)} < \frac{1}{8}\alpha\Delta^{-1}\ln(1 + \frac{\varepsilon}{4}).$$

We first add $\frac{\varepsilon_1}{n}$ to the weight of each edge, obtaining a weight function $\ell^{(1)}$. Note that $\sum_e \ell_e^{(1)} \leq 1 + \frac{1}{2}\varepsilon_1\Delta \leq 1 + \frac{1}{2}\varepsilon$.

Now we partition the edges into classes by weight, so that edges in the same class differ in weight by at most a factor of two. Define $L_i = \frac{\varepsilon_1}{n} \cdot 2^{i-1}$. We say that an edge is in class i if its weight is between L_i and L_{i+1}. Note that there are at most $b \leq \log(\varepsilon_1^{-1}n)$ classes that initially contain any edges of G.

We now define a decomposition procedure, which is very similar to techniques of Awerbuch [5]. Let H be an arbitrary n–node graph, and $\beta > 0$ a parameter.

(i) Initially all vertices are unmarked.

(ii) For each unmarked vertex v, let $r(v)$ denote the minimum r such that $|\delta(B_r(v))| \leq \beta|B_r(v)|$.

(iii) While there is an unmarked node u for which $|B_{r(u)}(u)| \le \frac{1}{2}n$:

 (a) Delete $\delta(B_{r(u)}(u))$ from H.
 (b) Mark all vertices in $B_{r(u)}(u)$.
 (c) Update the values of $r(v)$ for all unmarked v.

When this procedure terminates, the graph H will consist of components $X_1, X_2, \ldots, X_q$. We index these in the reverse order in which they were found by the decomposition procedure. Thus, $X_2, \ldots, X_q$ consist of marked nodes, and each has size at most $\frac{1}{2}n$. X_1 is either the final collection of nodes to be marked, or it consists of unmarked nodes and has size greater than $\frac{1}{2}n$.

(A) The diameter of each component X_i is at most $2\beta^{-1}\Delta \log n$.

To prove (A), we first observe that at all times, the values $r(v)$ cannot exceed $\beta^{-1}\Delta \log n$. For consider a vertex v in some iteration of the algorithm, and let $r = r(v)$. For all $j < r$, we have

$$|B_{j+1}(v)| \ge (1 + \beta\Delta^{-1})|B_j(v)|.$$

Thus if $r > \beta^{-1}\Delta \log n$, we would have

$$B_r(v) > (1 + \beta\Delta^{-1})^{\beta^{-1}\Delta \log n} \ge n,$$

a contradiction.

Hence, each node in a marked component has a path of length at most $\beta^{-1}\Delta \log n$ to the node u that "grew" this component, and hence the diameter of this component is at most $2\beta^{-1}\Delta \log n$. If the nodes in X_1 are unmarked, then each node $v \in X_1$ can reach more than $\frac{1}{2}n$ nodes by a path of length at most $r(v) \le \beta^{-1}\Delta \log n$; and hence there is a path of length at most $2\beta^{-1}\Delta \log n$ between any two nodes in X_1. This completes the proof of (A).

We now use this decomposition algorithm to define our smoothing algorithm, which will produce the metric space (G, ℓ') from $(G, \ell^{(1)})$. The algorithm is as follows; it will iteratively update the value of the function ℓ, but the underlying graph G will remain the same.

Set $a = \log(\varepsilon_2^{-1} \log n)$.
For $k = 1, \ldots, b-2$:

(i) With respect to the current metric $\ell^{(k)}$, let E_k denote the set of edges in classes $1, \ldots, b-k$, and $G_k = (V, E_k)$.

(ii) Run the decomposition algorithm on G_k with parameter $\frac{1}{4}\alpha$, obtaining connected components $X_1, \ldots, X_q$. Assume these are indexed, as above, in the reverse order in which they were found by the decomposition procedure.

(iii) For each $i > 1$, and every edge e of G_k with at least one end in X_i, increase the weight of e by L_{b-k-a}. This gives a new metric $\ell^{(k+1)}$.

Finally, set $\ell' = \ell^{(b-1)}$ (i.e. the metric produced at the end of the final iteration).

In order to bound the total increase in weight, we show that the following property is maintained throughout the execution of the algorithm.

(*) At the start of iteration k, we have

$$\sum_e \ell_e^{(k)} \le (1 + \frac{1}{2}\varepsilon)\left(1 + \frac{4\Delta\alpha^{-1}\varepsilon_2}{\log n}\right)^{k-1}. \tag{2}$$

First, observe that (*) holds at the beginning of iteration 1, by the definition of $\ell^{(1)}$ above. Next we show:

(B) If (*) holds for iteration k, then it holds for iteration $k+1$.

We consider iteration $k+1$. For each $i > 1$, the component X_i was equal to a ball $B_r(v)$ in some iteration of the decomposition procedure; let F_i denote the set of edges removed from $\delta_{G_k}(X_i)$ in this iteration. Note that F_i is not necessarily equal to $\delta_{G_k}(X_i)$, since some edges in $\delta_{G_k}(X_i)$ may have already been deleted in some earlier iteration of the decomposition procedure. By definition, $|F_i| \le \frac{1}{4}\alpha|X_i|$, and so $\sum_{i>1}|F_i| \le \frac{1}{4}\alpha\sum_{i>1}|X_i|$. Note that the removal of $\cup_{i>1}F_i$ disconnects the graph G_k into the components $X_1, X_2, \ldots, X_k$, and so $\cup_{i>1}F_i = \cup_{i>1}\delta_{G_k}(X_i)$. But each X_i, for $i > 1$, has size at most $\frac{1}{2}n$, and hence $|\delta_G(X_i)| \ge \alpha|X_i|$. Hence $|\cup_{i>1}\delta_G(X_i)| \ge \frac{1}{2}\alpha\sum_i|X_i|$, since each edge in $\cup_{i>1}\delta_G(X_i)$ may appear in two different X_i.

Thus,

$$\left|\bigcup_{i>1}\delta_G(X_i)\right| - \left|\bigcup_{i>1}\delta_{G_k}(X_i)\right| \ge \frac{1}{4}\alpha\sum_i|X_i|;$$

since all these edges belong to $E\backslash E_k$, their total weight is at least $(\frac{1}{4}\alpha\sum_i|X_i|)L_{b-k}$. The additive increase in weight associated with edges incident to $\cup_{i>1}X_i$ is at most $(\Delta\sum_{i>1}|X_i|)L_{b-k-a}$. Hence he increase in weight is at most

$$4\Delta\alpha^{-1}\frac{L_{b-k-a}}{L_{b-k}} = \frac{4\Delta\alpha^{-1}\varepsilon_2}{\log n}$$

times the existing weight in G, and so (*) follows for iteration $k+1$.

Thus property (*) holds throughout all iterations of the algorithm. Setting $k = b-1$ in Equation 2, and using the definition of ε_2, one can show the following.

(C) $\sum_e \ell'_e \le 1 + \varepsilon$.

We now argue that the metric space is sufficiently smooth; this will complete the proof of the theorem. First we show a bound on the minimum weight of a path between vertices that are adjacent to edges that end up in the same class. We then use this to bound the number of edges taken on a shortest path between any two vertices.

(D) In (G, ℓ'), let u and v be nodes incident to edges in class i. Then the lightest path from u to v in (G, ℓ') has weight at most $32\alpha^{-1}\Delta L_{i+a+1}\log n$.

We first observe that since the diameter of G is at most $2\alpha^{-1}\Delta\log n$ and all edges have weight at most $2L_b$, the bound in (D) is immediate if $i \ge b-a-3$. So suppose $i < b-a-3$.

Let X denote the large component X_1 found in iteration $k' = b - i - a - 1$. We observe that any node which did not belong to the component X can be incident only to edges of weight at least $L_{b-k'-a} = L_{i+1}$. Thus u and v both belong to X. We know from (A) that this component has diameter at most $8\alpha^{-1}\Delta \log n$; let Q be a u-v path in $G[X]$ containing at most this many edges. Now, all the edges in the component X, and hence in Q, have weight at most $2L_{b-k'} = 2L_{i+a+1}$ at the start of iteration k'; and over the course of all remaining iterations, the weight of each can grow by at most

$$\sum_{j=k'}^{\infty} L_{b-j-a} \leq 2L_{b-k'-a} = 2L_{i+1}.$$

Thus the weight of any edge in Q in (G, ℓ') is at most $2L_{i+a+1} + 2L_{i+1} \leq 4L_{i+a+1}$, and hence (D) follows.

Now, let u and v be two arbitrary nodes in G, and P the shortest path between them.

(E) For each i, P contains $O(\varepsilon^{-1} \log^2 n)$ edges of class i.

Set $q = 64\alpha^{-1}\Delta\varepsilon_2^{-1} \log^2 n$, and suppose that P contains $q+1$ edges of class i. In a traversal of P from u to v, let x denote the first node of P incident to an edge of class i, and y the last such node. We know that the weight of the subpath of P between x and y is at least $(q+1)L_i$. But by (D), there is an x-y path in (G, ℓ') of weight at most $32\alpha^{-1}\Delta L_{i+a+1} \log n \leq qL_i$. This is a contradiction, since P is a shortest u-v path.

Since there are only $\log(\varepsilon_1^{-1} n) \leq 2 \log n$ weight classes that contain any edges of G, it follows from (E) that (G, ℓ') is $O(\varepsilon^{-1} \log^3 n)$–smooth. ∎

We now discuss the application of this theorem in the context of the multicommodity flow problem. We will use the definitions discussed in the introduction: there is a set $\mathcal{T}$ of terminal pairs (*commodities*) in a graph G, and each terminal pair (u, v) has a *demand* ρ_{uv}. We say that a flow for $(G, \mathcal{T})$ is *feasible* if it has congestion at most 1. The problem of finding the minimum–congestion flow for $(G, \mathcal{T})$ can be formulated as a linear program of polynomial size, and this has the following two consequences. First, it can be solved in polynomial time, by algorithms such as [17, 16]. Second, the optimal solution is well-characterized by linear programming duality. However, the linear programming algorithms for multicommodity flow are non–combinatorial, and their running times are fairly large. Thus, a number of faster, combinatorial $(1+\varepsilon)$–approximations have been developed for the problem. The fastest general $(1+\varepsilon)$–approximation algorithm is due to Leighton et al. [22], with a running time of $O^*(k^2mn)$, where k is the number of commodities.

A recent multicommodity flow algorithm due to Awerbuch and Leighton [7] works as follows: if there is a feasible flow $\mathcal{F}$ routing $(1+\varepsilon)\rho_{uv}$ units of each commodity (u, v), and using flow paths of length at most d, then the algorithm finds a flow $\mathcal{F}'$ routing ρ_{uv} units of each commodity (u, v). The running time of the algorithm is $O^*(d^2km)$; thus, it has the potential to be much faster than that of [22] when d is small. Also, note that by uniformly scaling all demands and using binary search, one can apply the Awerbuch–Leighton algorithm to approximate $\nu^*(G, \mathcal{T})$ to within a factor of $1 + \varepsilon$.

In what follows, we use Theorem 4.2 in conjunction with linear programming duality to show that the family of bounded–degree expanders provides a natural class of graphs on which this parameter d is small; in particular, we show that d can be taken to be polylogarithmic in this case. It follows that the multicommodity flow algorithm of Awerbuch and Leighton runs in nearly *linear* time per commodity on bounded–degree expanders. Thus, our result is the following.

Theorem 4.3 *Let G be a bounded–degree expander on n nodes, $(G, \mathcal{T})$ a multicommodity flow problem, and $\nu^* = \nu^*(G, \mathcal{T})$. Then for any $\varepsilon > 0$, there is a flow $\mathcal{F}'$ of congestion at most $(1+\varepsilon)\nu^*$, such that the maximum length of a flow path used by $\mathcal{F}'$ is $O(\varepsilon^{-1}\log^3 n)$.*

Proof. For the purposes of this proof, we will say that a path in G is *short* if its length is at most $O(\varepsilon^{-1}\log^3 n)$, where the constant inside the $O()$ is determined by the proof of Theorem 4.2.

The following gives an easy lower bound on the congestion of any flow for $(G, \mathcal{T})$. Suppose we assign a weight ℓ_e to each edge e in such a way that

(i) $\sum_e \ell_e = 1$.

(ii) $\sum_{(u,v)\in\mathcal{T}} \rho_{uv}\ell(u,v) = z$.

Then it would follow, by the pigeonhole principle, that the congestion of any flow for $(G, \mathcal{T})$ is at least z. Let us use $\ell(\mathcal{T})$ to denote the value of $\sum_{(u,v)\in\mathcal{T}} \rho_{uv}\ell(u,v)$ for a set of edge weights $\{\ell_e\}$. The duality theorem of linear programming implies that in fact the minimum congestion of a flow for $(G, \mathcal{T})$ is equal to the *maximum* possible value of $\ell(\mathcal{T})$ over all set of edge weights $\{\ell_e\}$ that sum to 1. Let z denote this maximum value.

Now, if $\{\ell_e\}$ is a set of edge weights, let $\ell^s(u,v)$ denote the minimum weight of a *short* path from u to v, and define $\ell^s(\mathcal{T}) = \sum_{(u,v)\in\mathcal{T}} \rho_{uv}\ell^s(u,v)$. One can again argue using duality that the minimum congestion of a flow for $(G, \mathcal{T})$ on short paths is equal to the maximum possible value of $\ell^s(\mathcal{T})$ over all set of edge weights $\{\ell_e\}$ that sum to 1. Let z^s denote this maximum value.

We are interested in showing that the minimum congestion of a flow on short paths is at most $1+\varepsilon$ times the minimum congestion of any flow for $(G, \mathcal{T})$. In view of the above argument, it is enough to show that $z^s \le (1+\varepsilon)z$. So let ℓ denote the set of weights that achieve the optimal value z^s. We apply Theorem 4.2 to the metric space (G, ℓ), obtaining a set of edge weights $\{\ell'_e\}$ such that the metric space (G, ℓ') satisfies the following properties.

(i) $\sum_e \ell'_e \le 1+\varepsilon$.

(ii) $(\ell')^s(\mathcal{T}) = \ell'(\mathcal{T})$.

(iii) For all e, $\ell'_e \ge \ell_e$, and therefore $(\ell')^s(\mathcal{T}) \ge \ell^s(\mathcal{T})$.

Now define a set of edge weights $\{\ell''_e\}$ by simply setting $\ell''_e = \ell'_e / \left(\sum_e \ell'_e\right)$. Since $\sum_e \ell''_e = 1$, we have

$$z^s = \ell^s(\mathcal{T}) \le (\ell')^s(T) = \ell'(\mathcal{T}) \le (1+\varepsilon)\ell''(\mathcal{T}) \le (1+\varepsilon)z.$$

The theorem follows. ■

Recall that given a multicommodity flow problem $(G, \mathcal{T})$, the algorithm of [7] requires, for some positive ε, that there is a feasible flow routing at least $(1+\varepsilon)\rho_{uv}$ units of each commodity (u,v). If G is a bounded–degree expander, then given such a flow, the above theorem implies that there is a feasible flow routing $1+\frac{1}{2}\varepsilon$ times the demand for each commodity on short paths. (Note that it is enough to argue that this flow exists; one does not have to construct it.) Thus we have

Corollary 4.4 *Let G be a bounded–degree expander, and $\mathcal{T}$ a set of k terminal pairs. The Awerbuch–Leighton algorithm runs in $O^*(km)$ time on $(G, \mathcal{T})$.*

5 Graph Minors

As in the introduction, we say that H is a *minor* of G (or that G has an H-minor) if H may be obtained from a subgraph of G by contracting edges. An equivalent way to phrase this as follows.

Definition 5.1 *We say that an* embedding *of H in G is a collection of connected subgraphs of G, $\{X_v : v \in V(H)\}$, and a collection of paths in G, $\{P_e : e \in E(H)\}$, such that*

(i) The subgraphs $\{X_v\}$ are mutually vertex–disjoint.

(ii) If $e \in E(H)$ has ends $u, w \in V(H)$, then P_e has its endpoints in X_u and X_w. It is vertex–disjoint from all other $P_{e'}$, and from all X_v with the exception of X_u and X_w.

It is easy to verify that H is a minor of G if and only if there is an embedding of H in G.

Our result here is the following. We prove it for Δ-*regular* expander graphs; to apply it to general bounded-degree expanders, we can add an appropriate number of directed self-loops at each node, so that the stationary distribution of the random walk on the graph becomes uniform.

Theorem 5.2 *For every $\alpha > 0$ there is a number κ and a constant c such that the following holds. Let G be a Δ-regular α–expander graph on n nodes, and H a graph on at most $cn/\log^{\kappa} n$ nodes and edges. Then H is a minor of G, and there is a randomized polynomial–time algorithm to find an embedding of H in G.*

We noted in the introduction that this has consequences for the construction of long, simple paths in graphs with constant edge expansion, for any positive constant (cf. [28, 25, 14]).

In the remainder of this section, we sketch the method to construct an embedding of H in G, where H and G are as in Theorem 5.2. The method will be a randomized polynomial-time algorithm that succeeds in constructing the embedding with probability at least $1 - \delta$, for a constant $\delta > 0$. The construction and its analysis follow very closely the disjoint paths algorithm of Broder, Frieze, and Upfal [12]; in particular, a number of the basic lemmas that we need are direct adaptations of lemmas from their work. Since the proofs of Claims 5.3,5.4,5.5 are essentially the same as in [12] we do not feel it makes sense to repeat them here. As in Broder et al. [12], we make use of several constants in the analysis. κ_2 and κ_3 are required to be sufficiently large constants by the analysis of Claim 5.4; κ_1 is required to be sufficiently large relative to κ_3, also for the analysis of Claim 5.4; and κ is required to be sufficiently large relative to κ_1 for the analysis of Claim 5.3. For some of our applications of the union bound, we require our constants to be somewhat larger than those used in Broder et al. [12].

First of all, we can assume with only a loss in the constants that H is a 3-regular graph. This follows from the transitivity of the minor relation and the fact that any graph with k nodes and edges is a minor of a 3–regular graph on $O(k)$ nodes.

Let h denote the number of nodes of H. Our first goal is to choose a set of h nodes, so that each node is distributed uniformly in G, and no two nodes are close. Let π denote the stationary distribution of G. First choose a set R of $2h$ nodes independently according to π from G. By a greedy algorithm, choose an arbitrary set $Q \subset R$, $|Q| = h$, with the property that the minimum distance between any pair of nodes is at least $3\kappa_1 \ln\ln n$. The following claim is given in the analysis of Phase 3 in [12].

Claim 5.3 *The probability that the greedy algorithm fails to produce a set of size at least h is at most $o(1)$.*

Next pick a random mapping σ of the nodes in H to the nodes in Q. Consider the set of pairs of nodes in G: $T = \{(\sigma(u), \sigma(v))|(u, v) \in E(H)\}$. Note that because H is 3-regular, the number of pairs in T is $q = \frac{3}{2}h$. Let us order the pairs in T from 1 to q, and use (a_i, b_i) to denote the i^{th} pair.

For each $(a_i, b_i) \in T$, construct a *bundle* $B(i)$ of $m = (\ln n)^2$ paths $P_{i,1}, \ldots, P_{i,m}$ connecting a_i to b_i, where each path is chosen via a random walk as in [12]: Let $\tau = \kappa_2 \ln n$. First choose a random midpoint $x_{i,j}$ according to π. Then choose a random $a_i \to x_{ij}$ trajectory W'_{ij} (resp. W''_{ij}) from the trajectories of length τ (resp. b_i to x_{ij}) (this can be done efficiently since the paths are short). Let the j^{th} path in the bundle $B(i)$, W_{ij}, be the concatenation of W'_{ij} and W''_{ij} reversed, with any cycles removed.

For each vertex $u \in Q$, define the set $B_u = \{W_{ij}|a_i = u \text{ or } b_i = u\}$. Prune the walks in B_u so that the set of walks that remains satisfies the following condition: outside of a $\geq (\kappa_1 - \kappa_3)\ln\ln n$-neighborhood of u, the minimum distance between any pair of walks is $\geq 2\kappa_3 \ln\ln n$ (in particular, they are vertex disjoint). (Then also prune the corresponding path in $B(i)$.) Finally prune any walks in $B(i)$ that get too close (within $\kappa_3 \ln\ln n$) to any node in $Q - \{a_i, b_i\}$. After all pruning is finished, rename the surviving paths in each bundle $B(i)$: $P_{i1}, \ldots, P_{im_i}$. The next claim follows by a direct adaptation of the analysis of Phases 5 and 6 in [12].

Claim 5.4 *For κ_2 and κ_3 sufficiently large, $m_i \geq m/2$ for all i, with probability $1 - o(1)$.*

We are interested in choosing a path from each bundle so that paths not sharing endpoints are node–disjoint; note that due to the pruning above, two paths that share an endpoint can only intersect near their (shared) endpoints. Such intersections will not pose a problem in the construction of the embedding, below. Thus, let F be the graph where nodes correspond to paths in $\cup_i B(i)$ (indexed by the bundle number and the number of the path within the bundle); that is, $V_F = \{(i, j)|i = 1, \ldots, q; j = 1, \ldots, m_i\}$. We say that two paths in $\cup_i B(i)$ are in *conflict* if (i) their endpoints are all distinct, and (ii) they share a node. We define the edge set E_F of F by including an edge between (i, j) and (i', j') if and only if the corresponing paths are in conflict. An independent set in F that includes an element of each "row" (an element (i, j) for each i) will be called an *independent transversal*; such a set corresponds to a choice of a paths in G for each pair in T such that the paths are mutually not in conflict.

The pruning and the choice of vertices in Q guarantees that no two paths in different rows intersect near their endpoints. Now, following the analysis of Phases 7 and 8 in [12], based on an argument via the Lovász Local Lemma, one can show

Claim 5.5 *With probability $1 - o(1)$, F contains an independent transversal. Moreover, again with probability $1-o(1)$, the maximum size of a connected component of F is $O(\ln n/\ln\ln n)$,* *and so such an independent transversal can be found by brute-force search in polynomial time.*

Thus, having found the independent transversal, we obtain paths $Z_1, \ldots, Z_q$, such that Z_i has endpoints a_i and b_i, and none of the pairs of paths are in conflict. Thus it follows that if $Z_i \cap Z_j \neq \phi$, then Z_i and Z_j must share an endpoint; and by the above pruning process, they can only intersect within their first $\kappa_1 \ln \ln n$ steps from this common endpoint.

We now depart from the structure of the Broder et al. [12] algorithm, as we must use these paths to define an embedding of H in G. For each Z_i, with endpoints $\sigma(u)$ and $\sigma(v)$, write Z_i^u to denote the first $\kappa_1 \ln \ln n$ nodes of Z_i starting from endpoint u, Z_i^v to denote a the first $\kappa_1 \ln \ln n$ nodes of Z_i starting from endpoint v, and $Z_i' = Z_i \setminus (Z_i^u \cup Z_i^v)$. For $u \in H$, we define the "super–node" X_u in Definition 5.1 to be the union $Z_i^u \cup Z_j^u \cup Z_k^u$, for the three paths Z_i, Z_j, Z_k that have $\sigma(u)$ as an endpoint. For $e = (u, v) \in E(H)$, we define the "super–edge" P_e in Definition 5.1 to be Z_i', where Z_i is the path with endpoints $\sigma(u)$ and $\sigma(v)$. We now need only verify that all X_u and P_e are vertex-disjoint (except that P_e can meet X_u when u is an end of e).

(i) $X_u \cap X_v = \phi$ follows from the fact that all vertices in Q are at least $3\kappa_1 \ln \ln n$ apart.

(ii) If e and f do not have a common end, then $P_e \cap P_f = \phi$ follows from the fact that two paths that are not in conflict, with no common endpoints, do not share vertices. If e and f do have a common end, then $P_e \cap P_f = \phi$ follows from the pruning step above.

(iii) Finally, consider P_e and X_u, where u is not an end of e. Suppose P_e is constructed from a subset of the path Z_i, with endpoints $\sigma(v)$ and $\sigma(w)$, where $u \neq v, w$. Let us assume by way of contradiction that $P_e \cap X_u \neq \phi$. Then Z_i intersects some path Z_j whose endpoints are $\sigma(u)$ and $\sigma(y)$. But as argued above, this can happen only if Z_i and Z_j have a common endpoint, and this intersection between Z_i and Z_j must occur within $\kappa_1 \ln \ln n$ steps of this common endpoint. Since $u \neq v, w$, it follows that y is equal to one of v or w (say v), and the intersection between Z_i and Z_j occurs within $\kappa_1 \ln \ln n$ steps of $\sigma(v)$. Because $\sigma(u)$ and $\sigma(v)$ are at least a distance of $3\kappa_1 \ln \ln n$ apart, this intersection cannot occur within distance $\kappa_1 \ln \ln n$ of $\sigma(u)$, and hence does not occur in X_u, a contradiction.

Thus, we have an embedding of H in G, and hence G contains H as a minor.

Acknowledgements

The first author thanks Éva Tardos for helpful discussions on the problem of multicommodity flow on short paths.

References

[1] N. Alon, F.R.K. Chung, "Explicit construction of linear–sized tolerant networks," *Discrete Math.*, 72(1988), pp. 15–19.

[2] N. Alon, P.D. Seymour, R. Thomas, "A separator theorem for non–planar graphs," *J. American Math. Soc.*, 3(1990), pp. 801–809.

[3] S. Arora, F.T. Leighton, N. Maggs, “On–line algorithms for path selection in a non–blocking network,” *Proc. 22nd ACM Symp. on Theory of Computing*, 1990.

[4] Y. Aumann, Y. Rabani, “Improved bounds for all-optical routing,” *Proc. 6th ACM-SIAM Symp. on Discrete Algorithms*, 1995, pp. 567–576.

[5] B. Awerbuch, “The complexity of network synchronization,” *Journal of the ACM*, 32(1985).

[6] B. Awerbuch, Y. Azar, S. Plotkin, “Throughput-competitive online routing,” *Proc. 34th IEEE Symp. on Foundations of Computer Science*, 1993, pp. 32–40.

[7] B. Awerbuch, F. T. Leighton, “Improved approximations for the multi–commodity flow problem and local competitive routing in networks,” *Proc. 26th ACM Symp. on Theory of Computing*, 1994, pp. 489–496.

[8] Y. Bartal, A. Fiat, S. Leonardi, “Lower bounds for on-line graph problems with application to on-line circuit and optical routing,” *Proc. 28th ACM Symp. on Theory of Computing*, 1996.

[9] J. Beck, “On size Ramsey numbers of paths, trees, and cycles,” *J. Graph Theory*, 7(1983), pp. 115–130.

[10] A. Blum, A. Fiat, H. Karloff, Y. Rabani, unpublished manuscript.

[11] B. Bollobas, *Extremal Graph Theory*, Academic Press, 1978.

[12] A. Broder, A. Frieze, E. Upfal, “Existence and construction of edge-disjoint paths on expander graphs,” *SIAM J. Computing*, 23(1994), pp. 976–989.

[13] A. Broder, A. Frieze, E. Upfal, “Static and dynamic path selection in expander graphs: A random walk approach,” *Proc. 29th ACM Symp. on Theory of Computing*, 1997, pp. 531-539.

[14] J. Friedman, N. Pippenger, “Expanding graphs contain all small trees,” *Combinatorica*, 7(1987), pp. 71–76.

[15] N. Garg, V. Vazirani, M. Yannakakis, “Primal-dual approximation algorithms for integral flow and multicut in trees, with applications to matching and set cover,” *Proc. International Colloq. on Automata, Languages, and Programming*, 1993, pp. 64–75.

[16] N. Karmarkar, “A new polynomial–time algorithm for linear programming,” *Combinatorica*, 4(1984).

[17] L. Khachiyan, “A polynomial algorithm for linear programming,” *Soviet Math. Dokl.*, 244(1979).

[18] P. Klein, A. Agrawal, R. Ravi, S. Rao, “Approximation through multicommodity flow,” *Proc. 31st IEEE Symp. on Foundations of Computer Science*, 1990.

[19] J. Kleinberg, É. Tardos, "Disjoint Paths in Densely Embedded Graphs," *Proc. 36th IEEE Symp. on Foundations of Computer Science*, 1995, pp. 52–61.

[20] F.T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*, Morgan Kaufman, 1992.

[21] F.T. Leighton, B. Maggs, "Fast algorithms for routing around faults in multibutterflies and randomly wired splitter networks," *IEEE Trans. Computers*, 41(1992).

[22] F.T. Leighton, F. Makedon, S. Plotkin, C. Stein, É. Tardos, S. Tragoudas, "Fast approximation algorithms for the multicommodity flow problem," *Proc. 23rd ACM Symp. on Theory of Computing*, 1991.

[23] F.T. Leighton, S. Rao, "An appoximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms," *Proc. 29th IEEE Symp. on Foundations of Computer Science*, 1988.

[24] F.T. Leighton, S. Rao, "Circuit switching: a multicommodity flow based approach," *Proc. 9th International Parallel Processing Symposium*, 1995.

[25] L. Lovász, *Combinatorial Problems and Exercises*, North–Holland, 1979.

[26] D. Peleg, E. Upfal, "Disjoint paths on expander graphs," *Combinatorica*, 9(1989), pp. 289–313.

[27] S. Plotkin, S. Rao, W. Smith, "Shallow excluded minors and improved graph decomposition," *Proc. 5th ACM-SIAM Symp. on Discrete Algorithms*, 1994.

[28] L. Pósa, "Hamiltonian circuits in random graphs," *Discrete Math.*, 14(1976), pp. 359–364.

[29] P. Raghavan, E. Upfal, "Efficient all-optical routing," *Proc. 26th ACM Symp. on Theory of Computing*, 1994.

[30] N. Robertson, P.D. Seymour, "Graph Minors VIII. A Kuratowski theorem for general surfaces," *J. Combinatorial Theory Ser. B*, 48(1990), pp. 255–288.

[31] N. Robertson, P.D. Seymour, "Graph Minors XIII. The disjoint paths problem," *J. Combinatorial Theory Ser. B*, 63(1995), pp. 65–110.

[32] N. Robertson, P.D. Seymour, "Graph Minors XX. Wagner's conjecture," submitted for publication.

[33] E. Upfal, "An $O(\log N)$ deterministic packet routing scheme," *Proc. 21st ACM Symp. on Theory of Computing*, 1989.