New algorithmic aspects of the Local Lemma with applications to routing and partitioning

Tom Leighton* Satish Rao† Aravind Srinivasan‡

Abstract

The Lovász Local Lemma (LLL) is a powerful tool that is increasingly playing a valuable role in computer science. The original lemma was nonconstructive; a breakthrough of Beck and its generalizations (due to Alon and Molloy & Reed) have led to constructive versions. However, these methods do not capture some classes of applications of the LLL. We make progress on this, by providing algorithmic approaches to two families of applications of the LLL. The first improves recent results of Broder, Frieze & Upfal on constructing disjoint paths in expander graphs; the second provides constructive versions of certain applications of an extension of the LLL (modeling, e.g., hypergraph-partitioning and low-congestion routing problems). Our results can also be seen as constructive upper bounds on the integrality gap of certain packing problems. One common theme of our work is a "gradual rounding" approach.

1 Introduction.

The Lovász Local Lemma (LLL) is a powerful tool that can be used to show the existence of certain structures [12]. Let e denote the base of natural logarithms, and $E_1, E_2, \ldots, E_m$ be events with $\max_i \Pr[E_i] \le p$. If each E_i is mutually independent of all but at most D of the other events E_j and if $ep(D+1) \le 1$, then the key claim of the LLL (symmetric case) is that $\Pr[\bigwedge_{i=1}^m \overline{E_i}] \ge (1-ep)^m > 0$. See, e.g., [4, 27] for many applications of this. The term $(1-ep)^m$, though positive, is often "tiny"; so the result does not directly imply an efficient algorithm to produce a structure that avoids all the E_i. Breakthroughs in this direction have been made by Beck [6], and generalized by Alon [2] and Molloy & Reed [26]. However, as mentioned in [6, 2, 26], certain classes of applications of the LLL are not covered by these results. One such class arises because the approaches of [6, 2, 26] require $pD^3 = \mathrm{O}(1)$; they require something close to "$pD^9 = \mathrm{O}(1)$" for some situations. More importantly, as sketched in Appendix A, they do not guarantee polynomial-time algorithms in some settings where the underlying random variables have "large" ranges, even if $pD^c = \mathrm{O}(1)$ for an arbitrarily large constant c. Another such class involves some extensions of the LLL where $D \gg p^{-1}$. Here, we provide constructive approaches for certain instances of both of these classes; we now start with a framework that motivates many of the problems considered.

Let $[t] \doteq \{1, 2, \ldots, t\}$. The NP-hard *low-congestion routing* problem has attracted much attention, e.g., from the VLSI layout and network routing viewpoints [29, 16]. Given a graph $G = (V, E)$ and a collection $\mathcal{T} = \{(s_i, t_i) : i \in [k]\}$ of pairs of vertices in G, how do we construct an (s_i, t_i) path for each i, so that the *congestion* (maximum number of paths using any given edge) C is minimized? The multi-commodity flow relaxation of this problem is to send one unit of flow from s_i to t_i, for each i; the resultant minimum *fractional congestion* (maximum amount of flow using any edge) y^* is a lower bound on C. Since this relaxation is, e.g., a linear program (LP), the corresponding optimal multi-commodity flow can be found efficiently. How large can C be as a function of y^* and of some other parameters of G? (Even the special case $y^* = \mathrm{O}(1)$ is of much interest; $C \le \mathrm{O}(\log|V|/\log\log|V|)$ here [29]. This special case is related to several conjectures and results [32, 22, 25, 21].) Our second family (b) of results below yield new constructive results for generalizations of this question. Our first family of results, (a), are related to the question: how large can $y^* < 1$ be if we *require* that $C = 1$, i.e., if we need an edge-disjoint routing?

(a) Constructive sufficient conditions for disjoint paths. Suppose we require $C = 1$, as in the sentence above. Let $G = (V, E)$ with $|V| = n$ and $|E| = m$. Randomized rounding approaches show that if $y^* \le a_0/\sqrt{m}$, then $C = 1$ is achievable constructively; on the other hand, there are instances where $y^* =$

*Mathematics Department and the Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139. Supported in part by Army Grant DAAH04-95-1-0607 and DARPA Grant N00014-95-1-1246. E-mail: ftl@math.mit.edu.

†NEC Research Institute, 4 Independence Way, Princeton, NJ 08540. E-mail: satish@research.nj.nec.com.

‡School of Computing, National University of Singapore, Singapore 119260. Part of this work was done while visiting the NEC Research Institute, 4 Independence Way, Princeton, NJ 08540; part was supported by National University of Singapore Academic Research Fund Grants RP960620 and RP970607. E-mail: aravind@comp.nus.edu.sg.

$\Theta(1/\sqrt{n})$ and $C \geq 2$ [16]. Thus, the gap is large in general. An interesting alternative parametrization is based on d, the length of a longest path in a fractional optimal solution; the existentially optimal result that $C = 1$ if $y^* \leq a_1/d$, has been shown via the LLL [21]. (a_0, a_1, a_2 etc. denote positive constants.) It would be of much interest to make this constructive. As mentioned above, the approaches of [6, 2, 26] does not seem to extend here; this is discussed further in Appendices A and B. We get around this by an approach that generates a large number of paths according to a carefully chosen distribution. Though many of these paths could intersect, we show that if $y^* \leq a_2/(d \ln k)$, then we can locate an edge-disjoint collection within these paths with high probability (whp). This is our first main result, which we then apply to constructing disjoint paths in *expander graphs*.

Informally, an *expander* is a graph where any set of vertices that is not of "too high" cardinality, has a "reasonably large" number of edges leaving it. Since expanders possess very good routability and fault-tolerance properties [3, 5, 36], much attention has been paid to disjoint-path problems in various expander families [28, 9, 20, 17, 11, 21]. Let G be any bounded-degree n-vertex expander with sufficiently strong expansion properties. This will be formalized later; we just note for now that for large enough constant r, random r-regular graphs and the explicit r-regular Ramanujan graphs, e.g., of [24], are included here [9]. We will only consider *partial permutations* $\mathcal{T} = \{(s_i, t_i) : i \in [k]\}$; i.e., where all the s_i are distinct and all the t_i are distinct. A basic question is: what is the maximum $K = K(n)$ such that for *any* such $\mathcal{T}$ with K pairs (s_i, t_i), there is an edge-disjoint routing in G? (We need some criterion such as "partial permutation" for disjoint routability: clearly, no s_i and no t_i can appear in $\mathcal{T}$ with multiplicity more than the constant r. Also, since $r = O(1)$ and the average distance between pairs is $\Omega(logn)$, we have $K \leq O(n/\log n)$.) Two approaches have been developed for studying such problems. The first is based on random walks on expanders and their rapidly-mixing properties [9, 10]. The second is via multicommodity flow, exploiting the rich structure of low-congestion, short-length paths in expanders [20, 21]. Somewhat surprisingly, these very different ideas have both led, via the LLL and building on earlier results, to the result that $K \geq c_0 n/\log^2 n$, where c_0 depends on r and on the expansion properties of G, but not on n [10, 21]. Both these results were non-constructive; it is mentioned in [26] that their constructive version of the LLL does not seem to cover LLL-based methods such as those of [10]. Recently, a constructive disjoint-paths result for any fixed $\epsilon > 0$ and any $\mathcal{T}$ with up to $k = a_3 n/\log^{2+\epsilon} n$ pairs, has been developed [11]. Using our first main result above with some other ideas, we improve this to $k = a_4 n/(\log^2 n \log^{(s)} n)$ for any fixed $s \geq 1$. ($\log^{(s)}$ denotes the logarithm iterated s times; e.g., $\log^{(2)} n = \log\log n$.) Here, the constant r and the expansion properties of G are sufficiently high as functions of ϵ and s respectively. Also, a_3 and a_4 depend on r and the expansion properties of G, but not on n. Furthermore, we show that our result applies for a larger family of expanders than does the one of [11].

The "$a_4 n/(\log^2 n \log^{(s)} n)$" result has also been obtained independently by Frieze [13], through a random walk approach.

(b) Routing, partitioning, and minimax integer programs. Our second family of results concern *minimax integer programs* (MIPs) that generalize the low-congestion routing problem [35].

Definition 1.1. *An MIP has variables W and $\{x_{i,j} : i \in [n], j \in [\ell_i]\}$, for some integers $\{\ell_i\}$. Let $N = \sum_{i \in [n]} \ell_i$ and let x denote the N-dimensional vector of the variables $x_{i,j}$ (arranged in any fixed order). An MIP seeks to minimize a real W subject to: (i) $\forall i \in [n]$ $\sum_{j \in [\ell_i]} x_{i,j} = 1$; (ii) a system of linear inequalities $Ax \leq \vec{W}$, where $A \in [0,1]^{m \times N}$ and $\vec{W}$ is the m-dimensional vector with the variable W in each component, and (iii) $\forall i, j,\ x_{i,j} \in \{0,1\}$.*

Consider, e.g., a canonical NP-hard hypergraph-partitioning problem. We are given a set-system $H = (X, F)$, where $X = [n]$ and $F = \{S_1, \ldots, S_M\} \subseteq 2^X$. Given a positive integer ℓ, the problem is to partition X into ℓ parts, so that *each* S_j is "split well": we want a $\chi : X \to [\ell]$ that minimizes $W = \max_{j \in [M], k \in [\ell]} |\{i \in S_j : \chi(i) = k\}|$ [15]. This is easily written as an MIP with $m = M\ell$. The analog of d from the low-congestion routing problem in the context of MIPs, is the maximum number of nonzero entries in any column of A. Let $\kappa \geq 1$ be such that the minimum nonzero entry of A is κ^{-1}. As was for the low-congestion routing problem, an LP relaxation for an MIP is to remove the integrality constraints (iii); we also add the *valid constraint* $W \geq \kappa^{-1}$ (the validity of this is easy to check). The resulting LP optimum y^* is a lower bound on W. As MIPs are NP-hard, we seek approximation algorithms for them: we will present an efficient way of rounding an optimal solution to the LP. The current-best upper-bounds on W (as a function of y^* and other parameters) are due to [35], improving on earlier work of [29]. In particular, W has been shown to be "not much above" y^* if d is "small", i.e., if an MIP instance is *column-sparse.* However, these techniques are non-constructive and are based on an extension of the LLL. Here, we make progress on the algorithmic aspect, i.e., develop approximation algorithms for MIPs, by focusing on MIP instances where κ is not "too large": indeed, we have $\kappa = 1$ for MIPs with $\{0,1\}$ matrices A (e.g., for low-congestion routing and hypergraph-partitioning).

For $\mu > 0$ and $p \in (0,1)$, define $Q(\mu, p) = a_5 \frac{\log(p^{-1})}{\log(\log(p^{-1})/\mu)}$ if $\mu \leq \log(p^{-1})/2$, and to be $a_5\sqrt{\mu \log(\mu + p^{-1})}$ otherwise; $a_5 > 0$ is an absolute constant. The previous-best result for general MIPs was non-constructive, and showed that $W \leq y^* + \mathrm{O}(Q(y^*, 1/(2d))) + \mathrm{O}(1)$ [35]. Here, we present a randomized polynomial time algorithm that will produce an integral solution of value at most $y^* + \mathrm{O}(Q(y^*, 1/(2d\kappa))) + \mathrm{O}(1)$ whp. $Q(\mu, p)$ depends logarithmically on p^{-1}; thus, for families of MIPs where, e.g., κ grows at most as fast as $\mathrm{O}(\mathrm{poly}(d))$, we match the previous-best results *constructively*, to within a constant factor. Also, for any family of MIPs where $y^* = \omega(\log(d+\kappa))$ as $d\kappa \to \infty$, our algorithm will produce an integral solution of value $y^*(1+o(1))$ whp (e.g., in the hypergraph-partitioning problem, we can asymptotically constructively match the LP optimum if $\ell = o((\max_i |S_i|)/\log d)$, where d is the degree of H). For general MIPs, our improvement over the previous-best *constructive* result [29] is in essentially replacing m by $d\kappa$ in the integral solution produced: we get good improvements for MIP instances where $d\kappa \ll m$.

The approach of [26] is not directly applicable for us since the "dependencies" among the events are large, and since the underlying variables have large ranges. We get around these problems through a sequence of three different randomized rounding approaches; two of the ingredients of these are the algorithm of [26] and a "gradual rounding" technique used in our result (a). Lu [23] has independently achieved our result for the special case of MIPs with $\kappa = 1$.

Thus, we present new algorithmic approaches to some classes of packing problems, where the previous-best results were nonconstructive based on versions of the LLL.

2 Constructive sufficient conditions for disjoint paths.

Our first main result is Theorem 2.1. In all our applications of it, the parameter p will be $o(1)$ or bounded away from 1. So, the term "$\lceil \ln p^{-1} \rceil$" can be taken to be "$\ln p^{-1}$" for a first reading.

THEOREM 2.1. *There is a constant $c_1 > 0$ such that the following holds. Suppose we are given any (directed or undirected) graph G, any k (multi-)sets of paths in it, $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k$, and any parameter $p \in (0,1)$. Let the paths in $\mathcal{P}_i$ be denoted $P_{i,j}$; let d be the maximum, over all i,j, of the length of $P_{i,j}$. Let each $P_{i,j}$ carry some non-negative amount of flow such that: (i) the total flow in each $\mathcal{P}_i$ is 1, and (ii) for each edge in G, the total flow using it is at most $c_1/(d\lceil \ln p^{-1} \rceil)$. Then, there is a randomized polynomial algorithm to pick one path each from at least $k(1-p)$ of the $\mathcal{P}_i$, such that all paths chosen are edge-disjoint whp. (If we want one path from each of the $\mathcal{P}_i$, we can set, e.g., $p = 1/(2k)$.)*

COROLLARY 2.1. *There is a constant $c_1 > 0$ such that the following holds. Suppose we are given any (directed or undirected) graph G, any collection $\mathcal{T} = \{(s_i, t_i) : i \in [k]\}$ of vertex-pairs in G, and any parameter $p \in (0,1)$. Suppose there is a unit flow from s_i to t_i for each i, such that the fractional congestion is at most $c_1/(d\lceil \ln p^{-1} \rceil)$; d denotes the length of a longest flow-path. Then, there is a randomized polynomial algorithm to pick an (s_i, t_i)-path for at least $k(1-p)$ indices i, such that all paths chosen are edge-disjoint whp. (If we want one (s_i, t_i)-path for each i, we can set $p = 1/(2k)$.)*

Remark. Each of our algorithms will involve several random processes. For each process for which success or failure can be checked efficiently, we will just show that the process succeeds with constant probability. This suffices since, letting N denote the "input size" for the problem, we can repeat each such process, say $\mathrm{O}(N)$ times, to drive the failure probability down to $2^{-\Theta(N)}$. Thus, each of our algorithms will succeed whp.

We now present the algorithm and proof for Theorem 2.1. We will choose the constant $c_1 \in (0,1)$ sufficiently small. Let $z \doteq d\lceil \ln p^{-1} \rceil / c_1$; thus, the *fractional congestion* (maximum flow using any given edge) is at most $1/z$. Recall the notion of *randomized rounding* [29]: given a real $t \geq 0$, we round it to $\lceil t \rceil$ with probability $t - \lfloor t \rfloor$, and round it to $\lfloor t \rfloor$ with probability $1 - (t - \lfloor t \rfloor)$. There are three simple steps in the algorithm:

Step 1. Let n be the number of vertices in G. For a suitably large absolute constant $c_2 > 2$ (say 20), we scale all the flow values by $c_2 z \ln n$, and then conduct randomized rounding independently on each flow value (i.e., for each path $P_{i,j}$). By a standard analysis via the Chernoff-Hoeffding bounds [27], we have whp that:

(P1) the total flow f_i in each $\mathcal{P}_i$ is such that $c_2 z(\ln n)/2 \leq f_i \leq 2c_2 z \ln n$, **and**

(P2) the total flow on any edge is at most $2c_2 \ln n$.

(This can also be derandomized easily by the method of conditional probabilities.) Dividing the flow value on each $P_{i,j}$ by f_i, we restore the property that the total flow value in each $\mathcal{P}_i$ is 1. Furthermore, the fractional congestion now is at most

$$2c_2 \ln n (\min_i \ f_i)^{-1} \leq 2c_2 \ln n (c_2 z (\ln n)/2)^{-1}$$
$$= 4/z. \tag{2.1}$$

Let $x \doteq \max_i f_i$; (P1) shows that

$$c_2 z (\ln n)/2 \leq x \leq 2c_2 z \ln n. \tag{2.2}$$

Since the randomized rounding (of course) leads to integral flow values, we see, for any i, that the number of flow-paths $P_{i,j}$ carrying nonzero flow now is at most f_i; hence, the maximum number of flow-paths $P_{i,j}$ carrying nonzero flow now, for any i, is at most x.

Step 2. Suppose, after the scaling-rounding-rescaling process of Step 1, that flow path $P_{i,j}$ carries a flow of value $g_{i,j} \geq 0$. For each (i,j), we proceed as follows. Let $g'_{i,j}$ be the largest multiple of $1/(2x)$ that is no larger than $g_{i,j}$; replace $P_{i,j}$ by $2xg'_{i,j}$ copies of itself, each of the copies carrying a flow of precisely $1/(2x)$. We refer to the new paths as $P'_{i,\ell}$. Fix i. How many flow paths do we have in any $\mathcal{P}_i$ now? Since each flow value is now exactly $1/(2x)$ and the total flow in $\mathcal{P}_i$ is now at most 1, this number can be at most $2x$. On the other hand, recall that $\sum_j g_{i,j} = 1$, and, as seen above, that there are at most x (nonzero-flow) paths $P_{i,j}$. Since $g'_{i,j} \geq g_{i,j} - 1/(2x)$, we see by summing that $\sum_j g'_{i,j} \geq 1 - x/(2x) = 1/2$. Thus, since each of the new flow-paths carries a flow of exactly $1/(2x)$, there must be at least x many $P'_{i,j}$. So, for each i, the set A_i of flow-paths $P'_{i,j}$ is such that

$$x \leq |A_i| \leq 2x. \tag{2.3}$$

In the rest of the algorithm description, "flow paths" shall refer to these newly constructed flow paths $P'_{i,j}$.

Let y denote the maximum number of flow-paths $P'_{i,j}$ that use any given edge. Now, all paths $P'_{r,s}$ carry a flow of exactly $1/(2x)$. Thus, by (2.1),

$$y \leq (4/z)/(1/(2x)) = 8x/z \leq 8(2c_2 z \ln n)/z = 16c_2 \ln n, \tag{2.4}$$

the second inequality following from (2.2).

Step 3. Let $t \doteq \lceil \ln p^{-1} \rceil$. Independently for each $i \in [k]$, we do the following. For each i, we randomly select t flow-paths (without replacement) from among the collection of flow-paths $P'_{i,j}$. (The bounds $c_2 > 2$, (2.3), and (2.2) show that $|A_i| \geq z \ln n$. Now, $t = c_1 z/d \leq z \ln n$ since $c_1 < 1$. So our random-selection process is well-defined.) Call all the paths selected here *tentatively chosen*. We will say that two paths $P'_{a,b}$ and $P'_{r,s}$ are *neighbors* iff $a \neq r$ **and** $P'_{a,b}$ and $P'_{r,s}$ share a common edge. Call $P_{i,j'}$ *bad* iff it and some neighbor of it, are both tentatively chosen; call i dangerous iff all its t tentatively chosen paths $P'_{i,j}$ are bad.

THEOREM 2.2. *If $c_1 > 0$ is sufficiently small, then with probability at least $2/3$, the number of dangerous i is at most kp.*

Proof. Since each path passes through at most d edges and since each edge has at most y flow-paths using it (by definition of y), we see that each flow-path has at most dy neighbors. Fix i. We now bound the probability that i is dangerous. Suppose i is dangerous, and that it has a collection S of t bad paths $P'_{i,j}$. For each path in S, choose one among its bad neighbors arbitrarily. Suppose there are q such **distinct** bad neighbors $p_1, p_2, \ldots, p_q$ of the elements of S, where $1 \leq q \leq t$. We can represent this by an unordered q-tuple $\langle [S_1, p_1], [S_2, p_2], \ldots, [S_q, p_q] \rangle$, where: (i) the sets S_j form a partition of S, and (ii) p_j is a bad neighbor of all elements of S_j. Consider any such fixed tuple $T = \langle [S_1, p_1], [S_2, p_2], \ldots, [S_q, p_q] \rangle$. Suppose, for some distinct vertices $u_1, u_2, \ldots, u_r$, that the $t + q$ paths are in T are: ℓ_1 paths from A_{u_1}, ℓ_2 paths from A_{u_2}, ..., ℓ_r paths from A_{u_r}. (The positive integers r and $\ell_1, \ell_2, \ldots, \ell_r$ are such that $\ell_j \leq t$ and $\sum_j \ell_j = t + q$.) The probability of all of T's $(t+q)$ paths getting tentatively chosen is

$$\left(\prod_{j=1}^{r} \binom{|A_{u_j}| - \ell_j}{t - \ell_j} / \binom{|A_{u_j}|}{t} \right) \leq \prod_{j=1}^{r} (t/|A_{u_j}|)^{\ell_j}$$
$$\leq \prod_{j=1}^{r} (t/x)^{\ell_j}$$
$$= (t/x)^{t+q}; \tag{2.5}$$

the second inequality is a consequence of (2.3).

Fix q. We next upper-bound the number of possible unordered q-tuples $\langle [S_1, p_1], \ldots, [S_q, p_q] \rangle$. To do so, we will bound the number of such *ordered* q-tuples, and divide the result by $q!$. Let us now bound the number of ordered q-tuples. Clearly, we may assume that each S_j is nonempty. We first fix the cardinalities of the S_j: this can be done in $\binom{t-1}{q-1}$ ways, since $\sum_j |S_j| = t$ and as each S_j is nonempty. Suppose $|S_j| = t_j$. Recall from (2.3) that a member $P'_{i,\ell}$ of S_1 can be picked in at most $2x$ ways. Given this choice, since p_1 is a neighbor of $P'_{i,\ell}$, p_1 can be picked in at most dy ways. Next, since the remaining $t_1 - 1$ elements of S_1 are all neighbors of p_1, they can be picked in at most $\binom{dy}{t_1 - 1}$ ways. Thus, the number of ways of choosing $[S_1, p_1]$ is at most

$$(2x) \cdot (dy) \cdot \binom{dy}{t_1 - 1} \leq (2x) \cdot (dy)^{t_1}.$$

(Upper-bounding $\binom{dy}{t_1-1}$ by $(dy)^{t_1-1}/(t_1-1)!$, say, does not give asymptotic improvements. Also, we can improve our constants slightly by including an additional element of randomness in Step 3, motivated by a "random permutation" idea of [34]. We will take up these ideas in the full version of the paper.) Arguing similarly for $[S_2, p_2], \ldots, [S_q, p_q]$, we see, for each fixed choice of $t_1, t_2, \ldots, t_q$, that the number of ordered q-tuples is at most $\prod_{j=1}^{q}((2x) \cdot (dy)^{t_j}) = (2x)^q \cdot (dy)^t \leq 2^t x^q \cdot (dy)^t$. Next, as seen above, (i) there are

$$\binom{t-1}{q-1} \leq \binom{t}{q} \leq e^q t^q / q^q \leq e^t t^q / q^q$$

ways of choosing the t_i; (ii) the number of unordered q-tuples is the number of ordered q-tuples divided by $q!$. Combining with (2.5), the probability of existence of some q-tuple is at most

$$(2.6) c_3^t x^q (dy)^t (t/q)^q (t/x)^{t+q} / q! \leq c_4^t (t/q)^{2q} (tdy/x)^t,$$

where $c_3, c_4 > 0$ are absolute constants. (In going from the l.h.s. to the r.h.s., we used the fact that $q! \geq (q/e)^q \geq q^q/e^t$.) By calculus, it is evident that $(t/q)^{2q} \leq e^{2t/e}$. Next, by recalling that $z = dt/c_1$ and by using the lower bound on x and upper bound on y from (2.2) and (2.4), we see from (2.6) that the probability of existence of some q-tuple is at most $(c_5 c_1)^t$, where c_5 is an absolute constant. (c_5 is independent of c_1.) Finally, since q can only take t values (from 1 to t), the probability that i is dangerous is at most $t(c_5 c_1)^t$, which can be made, say, at most $p/3$ by taking c_1 appropriately small (recall that $t = \lceil \ln p^{-1} \rceil$). Thus, by Markov's inequality, the probability that there are more than kp dangerous i is at most $1/3$. This completes the proof of Theorem 2.2.

As usual, the above probability of $2/3$ can be boosted by repeating the random process sufficiently many times. Thus, we will be able to efficiently: (i) select some $T \subseteq [k]$ of cardinality at least $k(1-p)$, and (ii) pick one tentatively chosen $P'_{i,j}$ for each $i \in T$, such that all the paths picked are edge-disjoint. This proves Theorem 2.1.

2.1 Disjoint paths in expander graphs.

We show how to use Theorem 2.1 to construct disjoint paths in expander graphs, improving on a result of [11]. Given an undirected graph $H = (V, E)$ and any $S \subseteq V$, let $\delta(S)$ denote the set of edges of H that have precisely one end-point in S. Recall that the *edge-expansion* of H is

$$\min_{S \subseteq V : |S| \leq |V|/2} |\delta(S)|/|S|.$$

H is called a *β-expander* if its edge-expansion is at least β.

We now present the essence of [20] that is relevant here. Call H a *triple expander* if its edge-set can be efficiently partitioned into three spanning subgraphs, all three of which are 1-expanders. Let $H = (V, E)$ be an n-vertex triple expander with maximum degree some constant r. Given any partial permutation $\mathcal{T} = \{(s_i, t_i) : i \in [k]\}$ on H, the algorithm of [20] efficiently produces: (a) k (multi-)sets of paths in H, $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k$, such that each path $P_{i,j}$ in any $\mathcal{P}_i$ has length at most $b_0 \log n$, and (b) a non-negative flow on each $P_{i,j}$ such that the total flow in each $\mathcal{P}_i$ is 1; also, the total amount of flow on any edge of H is at most $b_1 k (\log n)/n$. Here, b_0 and b_1 are constants that depend only on r. We also have the key property:

(P3): "$\forall T \subseteq [k]$ and any selection of one path each from $\{\mathcal{P}_i : i \in T\}$ such that the selected paths are edge-disjoint, we can efficiently produce an edge-disjoint routing for $\{(s_i, t_i) : i \in T\}$ in H".

Next, we present some useful results and definitions from [9, 11]. Let $\exp(x)$ denote e^x. As in [11], call an r-regular n-vertex graph $H = (V, E)$ an (α, β, γ)-expander if for each $S \subseteq V$: (i) $\delta(S) \geq (1-\alpha) r |S|$ if $|S| \leq \gamma n$, and (ii) $\delta(S) \geq \beta |S|$ if $\gamma n < |S| \leq n/2$. Define $H(\gamma) \doteq ((1-\gamma)^{1-\gamma} \gamma^\gamma)^{-1}$ and $\psi(\epsilon) \doteq (1-\epsilon)\ln(1-\epsilon) + \epsilon$. As discussed in [11], a useful consequence of a result of [9] is

THEOREM 2.3. **([9, 11])** *Let $H = (V, E)$ be an r-regular n-vertex graph that is an (α, β, γ)-expander. For any $\epsilon \in (0, 1)$ and any integer $\ell \leq r/2$ such that $\beta > \ell(\gamma\psi(\epsilon))^{-1} \ln(H(\gamma))$, there is a randomized algorithm to partition the edge-set of H into ℓ subgraphs. With probability at least $1 - \exp(-n(\beta\gamma\psi(\epsilon)/\ell - \ln(H(\gamma)) - o(1)))$, all the ℓ subgraphs span V and have edge-expansion at least $\min\{\lfloor r/(2\ell) \rfloor - \alpha r, (1-\epsilon)\beta/\ell\}$; the $o(1)$ term tends to 0 as $n \to \infty$.*

Though the requirements on H look complicated, it is pointed out in [9] that r-regular Ramanujan graphs (see, e.g., [24]) and random r-regular graphs (whp) have the property that for any S with $|S| \leq n/2$, $\delta(S) \geq r(1-\lambda)(1-|S|/n)|S|$, where $\lambda \sim 2/\sqrt{r}$ for large (constant) r. In particular, as observed in [11], for any fixed $\ell \geq 1$, there is a sufficiently large constant r such that r-regular Ramanujan graphs and random r-regular graphs (whp): **(P4)** satisfy the pre-conditions of Theorem 2.3, and also have **(P5)** $\min\{\lfloor r/(2\ell) \rfloor - \alpha r, (1-\epsilon)\beta/\ell\} \geq 1$. Given any fixed integer $s \geq 1$, let $G = (V, E)$ be any r-regular graph that satisfies (P4) and (P5) for $\ell = 3s$. We now show how, given any partial permutation $\mathcal{T}$ on G with $|\mathcal{T}| \leq a_4 n/(\log^2 n \log^{(s)} n)$, we can efficiently construct an edge-disjoint routing for $\mathcal{T}$ on G. Recall the constants b_0 and b_1 introduced in the definition of triple expanders, and the constant c_1 of Theorem 2.1. $a_4 > 0$ is a constant depending only on b_0, b_1 and c_1. We assume that n is sufficiently large. Use Theorem 2.3 to split G into $\ell = 3s$ many

1-expanders, and group these as s triple expanders $G_1, G_2, \ldots, G_s$. Using (P3) above and Theorem 2.1 on G_1 with $p = (\log^{(s-1)} n)^{-2}$, we can whp route all but at most $np/\log^2 n$ pairs on G_1. We take the unrouted pairs and repeat the process on G_2 with $p = (\log^{(s-2)} n)^{-2}$; whp, the remaining number of unrouted pairs will be at most $\mathrm{O}(n(\log^{(s-2)} n)^{-2}/\log^2 n)$. Repeating this on $G_3, G_4, \ldots$, we will whp be left with just $t = \mathrm{O}(n/\log^4 n)$ pairs to be routed on G_s. All these pairs can be routed whp on G_s by invoking (P3) and Theorem 2.1 on G_s, with $p = 1/(2t)$.

As mentioned in Section 1, this improves on the "$\mathcal{T} \leq a_3 n/\log^{2+\epsilon} n$" result of [11]. We also improve on [11] as follows. Given any fixed ℓ and any large enough r, n, it is simple to check that there are constants $\alpha, \beta, \gamma > 0$ *that depend only on ℓ and not on r*, such that any r-regular (α, β, γ)-expander satisfies (P4) and (P5). However, the results of [11] require, e.g., $\alpha \leq \mathrm{O}(1/\log r)$; in other words, very high expansion is required for "small" sets in [11], if r is "large".

3 Constructive approximations for minimax integer programs.

We now present results for minimax integer programs (MIPs). Consider a generic MIP and its relaxation, as described in Section 1. We assume w.l.o.g. that the LP optimal solution is a *basic* feasible solution; this lets us assume

$$\sum_{i \in [n]} \ell_i \leq m. \tag{3.7}$$

Let $\{x^*_{i,j} : i \in [n], j \in [\ell_i]\}$ denote the given optimal solution to the LP relaxation. Starting with this, the algorithm will construct a sequence of feasible solutions to the LP relaxation; variables $x_{i,j}$ that get rounded to 0 or 1 will never have their values re-altered. At any point in the execution of the algorithm, suppose x denotes the current vector of the variables. We let:

- L be the maximum, over all $i \in [n]$, of the number of nonzero $x_{i,j}$;
- Δ be the maximum, over all $i \in [m]$, of the number of nonzero variables in the ith constraint of the linear system $Ax \leq \vec{W}$ of the MIP;
- $D = d \cdot \max\{L, \Delta\}$, and
- V be the objective function value of the current fractional solution (i.e., $V = \max_{i \in [m]} (Ax)_i$).

Initially, we have

$$L \leq L_0 = \max_{i \in [n]} \ell_i; \qquad \Delta \leq \Delta_0 = \sum_{i \in [n]} \ell_i;$$

$$D \leq D_0 = d \sum_{i \in [n]} \ell_i; \qquad V = V_0 = y^*.$$

We use a sufficiently large absolute constant $s_0 > 0$. We present our algorithm for the case $y^* \geq 1$; the details are similar for the case $y^* < 1$ and will be described in the full version. So suppose $y^* \geq 1$. We have three procedures: YSmall, YLarge, and FinalRound. If $y^* \leq \log m$, we run YSmall and then FinalRound; if $y^* > \log m$, we run YLarge and then FinalRound. The goal of both YSmall and YLarge is to construct a feasible solution to the LP such that whp,

$$V \leq y^* + \mathrm{O}(\sqrt{y^*}); \; L \leq \mathrm{polylog}(m); \; D \leq \mathrm{poly}(y^*, \kappa, d). \tag{3.8}$$

Procedure YSmall. Recall that this is run iff $y^* \leq \log m$. Starting with the LP's optimal solution, we will construct a sequence of feasible solutions to the LP relaxation; the goal of each iteration here is to round "many" of the currently nonzero variables to 0. For notational convenience, define a Boolean expression $\mathcal{E}$ by:

$$\mathcal{E} \equiv ((D > 2^{4s_0^2}) \text{ and } (d\kappa V^2 (\log^2 D) \cdot (1 + s_0/(V\sqrt{\log D})) \leq D^{3/4})).$$

Procedure YSmall is as follows (it uses a procedure Round, which will be described below):

if $D > 2^{4s_0^2}$ **then**
  {Run procedure Round; Update L, Δ, D and V.}
while $\mathcal{E}$ **do:**
  {Run procedure Round; Update L, Δ, D and V.}

Procedure Round is a two-step process that transforms one feasible fractional solution to another. Let the current feasible solution for the LP have variables denoted $\{x_{i,j}\}$, with objective function value V. Note that $V \geq y^* \geq 1$. The two steps of Round are as follows.

Step 1. For each i, j, set $x'_{i,j} := V(\log^2 D) x_{i,j}$. Consider conducting randomized rounding on the $x'_{i,j}$ in a manner similar to Step 1 in our proof of Theorem 2.1. That is, *independently* for each (i,j), round $x'_{i,j}$ to a random $z_{i,j} \in \{\lfloor x'_{i,j} \rfloor, \lceil x'_{i,j} \rceil\}$ as follows: $\Pr[z_{i,j} = \lceil x'_{i,j} \rceil] = x'_{i,j} - \lfloor x'_{i,j} \rfloor$, and $\Pr[z_{i,j} = \lfloor x'_{i,j} \rfloor] = 1 - \Pr[z_{i,j} = \lceil x'_{i,j} \rceil]$. Consider the system of "bad" events:

(B1) $(Az)_i > V^2 (\log^2 D) \cdot (1 + s_0/(V\sqrt{\log D}))$, $1 \leq i \leq m$, and

(B2) $|\sum_j z_{i,j} - V \log^2 D| > s_0 \sqrt{V \log^3 D}$, $1 \leq i \leq n$.

If s_0 is a sufficiently large absolute constant, the probability of each of these events is less than $(8D^9)^{-1}$, by a simple use of the Chernoff-Hoeffding bounds. Note that only the currently non-zero variables effectively take part in the above rounding. Using this and the definition of D, we can check that the "dependence" among the bad events above in the sense of the LLL, is at most D. Note next that the underlying independent random variables are, for each (i, j), the choice of setting $z_{i,j} \in \{\lfloor x'_{i,j} \rfloor, \lceil x'_{i,j} \rceil\}$. Thus, in the notation of Appendix A, we have $\gamma = 2$ here. So, as sketched in Appendix A, the approach of [26] can be used to set the $z_{i,j}$ efficiently in a way that avoids all the above bad events; we run this algorithm. With high probability, we would have avoided all the bad events in (B1) and (B2).

Step 2. We now restore the condition that the $x_{i,j}$ add up to 1 for each i. For all i, j, set $x''_{i,j} := z_{i,j}/\sum_u z_{i,u}$. This vector x'' is the new feasible solution that is output by ROUND.

LEMMA 3.1. *Suppose $D > 2^{4s_0^2}$ at some point, and that we run* ROUND *then. Whp, the following bounds hold for the new values L', Δ', D' and V' respectively of L, Δ, D and V.*
(i) $L' \le V(\log^2 D) \cdot (1 + s_0/\sqrt{V \log D})$.
(ii) $\Delta' \le \kappa V^2 \log^2 D(1 + s_0/(V\sqrt{\log D}))$.
(iii) $D' \le d\kappa V^2 \log^2 D(1 + s_0/(V\sqrt{\log D}))$.
(iv) $V' \le V(1 + 4s_0/\sqrt{V \log D})$.

Proof. (i) As seen above, all the bad events in (B1) and (B2) have been avoided whp. Let us assume that all these events have been avoided. For each $i \in [n]$, the $z_{i,j}$ are non-negative integers summing to at most $V(\log^2 D)\cdot(1+s_0/\sqrt{V \log D})$, since all the events in (B2) above have been avoided. This proves (i). Similarly, since all the events in (B1) have been avoided, the definition of κ yields (ii). Part (iii) follows since $D' = d \cdot \max\{L', \Delta'\}$. As for part (iv), since all the events in (B1) and (B2) have been avoided,

$$\begin{aligned} V' &\le V(1 + s_0/(V\sqrt{\log D})) \cdot (1 - s_0/\sqrt{V \log D})^{-1} \\ &\le V(1 + 4s_0/\sqrt{V \log D}); \qquad (3.9) \end{aligned}$$

the last inequality follows from the assumption "$D > 2^{4s_0^2}$" and from the fact $V \ge 1$. Thus we have proven Lemma 3.1.

THEOREM 3.1. *Whp: (a) The "while" loop of Procedure* YSMALL *terminates within $t = \mathrm{O}(\log\log D_0)$ iterations; (b) At the end of Procedure* YSMALL*, $V \le y^* + \mathrm{O}(\sqrt{y^*})$, $L \le$ polylog(m), and $D \le$ poly(y^*, κ, d).*

Proof. If the "if" statement of Procedure YSMALL was not executed, the proof is immediate. So, suppose the "if" statement was executed. Suppose the "while" loop was never executed. Since $D \le m^2$ by (3.7), we see that whp, $V \le y^* + \mathrm{O}(\sqrt{y^*})$ and $L \le \mathrm{O}(y^* \mathrm{polylog}(m))$ by parts (i) and (iv) of Lemma 3.1; as Procedure YSMALL is only run if $y^* \le \log m$, we infer that $L \le \mathrm{polylog}(m)$. The termination condition of the "while" loop implies $D \le \mathrm{poly}(y^*, \kappa, d)$.

Suppose the while loop was executed for at least one iteration. The precondition "$d\kappa V^2(\log^2 D) \cdot (1 + s_0/(V\sqrt{\log D})) \le D^{3/4}$" of the loop and part (iii) of Lemma 3.1 show that after each such iteration, D gets reduced whp to

$$D' \le D^{3/4}. \qquad (3.10)$$

This, along with the pre-condition "$D > 2^{4s_0^2}$", shows that Procedure YSMALL terminates within $\mathrm{O}(\log\log D_0)$ iterations whp. We now proceed to bound the final values of L, D and V. Let $D_0, D_1, \ldots, D_t$ be the sequence of values taken by D. Recall that $V_0 = y^*$. For $i = 0, 1, \ldots, t-1$, define

$$V_{i+1} = V_i(1 + 4s_0/\sqrt{V_i \log D_i});$$

we have by (3.9) that the final value of V is at most V_t, whp. We now upper-bound V_t. Let $0 \le i \le t-1$. Since $\log D_i \ge (4/3)^{t-1-i} \log D_{t-1}$ (see (3.10)), we have

$$\begin{aligned} V_{i+1} &\le V_i(1 + 4s_0/\sqrt{V_0 \log D_i}) \\ &\le V_i \exp(4s_0/\sqrt{V_0 \log D_i}) \\ &\le V_i \exp(4(\sqrt{3}/2)^{t-1-i} s_0/\sqrt{V_0 \log D_{t-1}}). \end{aligned}$$

A telescoping product gives

$$\begin{aligned} V_t &\le V_0 \exp(4s_0(\sum_{i=0}^{t-1} (\sqrt{3}/2)^{t-1-i})/\sqrt{V_0 \log D_{t-1}}) \\ &\le V_0 \exp(\mathrm{O}(1/\sqrt{V_0})) \\ &= V_0 + \mathrm{O}(\sqrt{V_0}) \\ &= y^* + \mathrm{O}(\sqrt{y^*}). \end{aligned}$$

Since $D \le m^2$ by (3.7), we see that whp, $L \le \mathrm{O}(y^* \mathrm{polylog}(m)) \le \mathrm{polylog}(m)$ by part (i) of Lemma 3.1. Finally, the termination condition of the "while" loop implies $D \le \mathrm{poly}(y^*, \kappa, d)$. This finishes the proof of Theorem 3.1.

Procedure YLARGE**.** As seen above, this is run iff $y^* > \log m$. The fact that $y^* > \log m$ will keep this procedure much simpler than YSMALL.

Recall that x^* is the given optimal solution to the LP; let $\lambda = 2my^*$. For each (i, j), define $\lfloor \lambda x^*_{i,j} \rfloor$ many new variables $\{v_{i,j,k} : 1 \le k \le \lfloor \lambda x^*_{i,j} \rfloor\}$, each having value $1/\lambda$. Replace each $x^*_{i,j}$ by $\sum_k v_{i,j,k}$, and let v denote the vector of the variables $\{v_{i,j,k} : i,j,k\}$. Since $\sum_k v_{i,j,k} \leq x^*_{i,j}$, we get $\max_{i\in[m]}(Av)_i \leq y^*$. Let ℓ'_i be the number of $v_{i,j,k}$, for each $i \in [n]$. Since $\sum_j x^*_{i,j} = 1$, the total number of $x^*_{i,j}$ is at most m (see (3.7)), and $\lfloor \lambda x^*_{i,j} \rfloor \geq \lambda x^*_{i,j} - 1$, we can check that $n'_i \geq \lambda - m$.

Independently for each $i \in [n]$, choose $\lceil \log m \rceil$ variables $v_{i,j,k}$ at random (without replacement) and set them all to 1; all other variables $v_{i,j,k}$ are set to 0. Let the resulting random analog of v be denoted w. Since $\Pr[w_{i,j,k} = 1] = \lceil \log m \rceil / n'_i \leq (\lceil \log m \rceil / \lambda) \cdot (1 + 1/y^*)$, it follows that for each $i \in [m]$, $\mathbf{E}[(Aw)_i] \leq y^* \lceil \log m \rceil (1 + 1/y^*)$. Furthermore, although the variables $w_{i,j,k}$ are not all independent, a "negative correlation" property enjoyed by them can be exploited via [30] to show that if s_0 is sufficiently large, then for each $i \in [m]$,

$$\Pr[(Aw)_i > y^* \lceil \log m \rceil (1 + 1/y^*) + s_0 \sqrt{y^*} \log m]$$

is at most $1/(3m)$. Thus, with probability at least $2/3$ (which can be boosted as usual), we have $(Aw)_i \leq y^* \lceil \log m \rceil (1 + \mathrm{O}(1/\sqrt{y^*}))$ for all i. Finally, we set $x_{i,j} := (\sum_k w_{i,j,k}) / \lceil \log m \rceil$ for each i, j.

THEOREM 3.2. *The parameters L, V and D of the vector x produced by* YLARGE *are bounded as follows. $L \leq \lceil \log m \rceil$; whp, $V \leq y^* + \mathrm{O}(\sqrt{y^*})$ and $D \leq poly(y^*, \kappa, d)$.*

Proof. The bound on L is immediate. From the above discussion, we have whp that $(Aw)_i \leq y^* \lceil \log m \rceil (1 + \mathrm{O}(1/\sqrt{y^*}))$ for all i. Two consequences of this are: (i) $V = \max_{i\in[m]}(Ax)_i \leq y^* + \mathrm{O}(\sqrt{y^*})$, and (ii) $\Delta \leq \mathrm{O}(\kappa y^* \lceil \log m \rceil)$ (using the definition of κ). So $D \leq \mathrm{O}(d \kappa y^* \lceil \log m \rceil)$. Since we would have run YLARGE iff $y^* > \log m$, we have $D \leq \mathrm{poly}(y^*, \kappa, d)$, completing the proof of Theorem 3.2.

Procedure FINALROUND**.** Recall that this is the final procedure run; let x be the current feasible solution to the LP. By Theorem 3.1 and Theorem 3.2, (3.8) holds whp. Assume that (3.8) holds. Define $D' = d\Delta L$; since (3.8) holds, we have

$$D' \leq d(D/d)^2 \leq \mathrm{poly}(y^*, \kappa, d). \tag{3.11}$$

Independently for each i, we choose exactly one $x_{i,j}$ to be set to 1: $\Pr[x_{i,j} = 1] = x_{i,j}$. Recall the function $Q(\cdot,\cdot)$ from Section 1. Consider the following m "bad" events: $(Ax)_i > V + s_0 Q(V, 1/(2D'))$, $1 \leq i \leq m$. A Chernoff-Hoeffding bound shows that the probability of each of these events is less than $(8(D')^9)^{-1}$, if s_0 is sufficiently large. It can also be checked that the "dependence" among these events is at most D'. Finally, note that we have n underlying independent random variables; the number of choices for each of these is at most L, by definition of L. In the notation of Appendix A, we have $\gamma = L$. Now note from (3.8) that $L \leq \mathrm{polylog}(m)$. So, as shown in Appendix A, exhaustive search in each "component" that arises from the algorithm of Appendix A can be conducted in

$$\gamma^{(\log m)^{2/5} \log\log m} \leq \mathrm{poly}(m)$$

time. Thus, we can efficiently apply the methodology of [26] to avoid all the above m bad events whp; this concludes the algorithm.

The objective function value of the final solution is at most $V + s_0 Q(V, 1/(2D'))$. The bound (3.11), combined with the facts that $y^* \geq 1$ and $V \leq y^* + \mathrm{O}(\sqrt{y^*})$ (which follows from (3.8)), leads to a simple calculation that shows that $V + s_0 Q(V, 1/(2D'))$ is upper-bounded by $y^* + \mathrm{O}(Q(y^*, 1/(2d\kappa))) + \mathrm{O}(1)$.

Acknowledgement. We thank Alan Frieze for his helpful comments, and for sending us copies of [11] and [14].

References

[1] N. Alon, *Eigenvalues and expanders*, Combinatorica, 6 (1986), pp. 83–96.

[2] N. Alon, *A parallel algorithmic version of the Local Lemma*, Random Structures & Algorithms, 2 (1991), pp. 367–378.

[3] N. Alon and F. R. K. Chung, *Explicit construction of linear sized tolerant networks*, Discrete Mathematics, 72 (1989), pp. 15–19.

[4] N. Alon and J. H. Spencer, *The Probabilistic Method*, Wiley–Interscience Series, John Wiley & Sons, Inc., New York, 1992.

[5] S. Arora, F. T. Leighton, and B. M. Maggs, *On-line algorithms for path selection in a nonblocking network*, SIAM J. Comput., 25 (1996), pp. 600–625.

[6] J. Beck, *An algorithmic approach to the Lovász Local Lemma*, Random Structures & Algorithms, 2 (1991), pp. 343–365.

[7] A. Broder, A. Frieze, S. Suen and E. Upfal, *Optimal construction of edge-disjoint paths in random graphs*, In Proc. ACM-SIAM Symposium on Discrete Algorithms, pages 603–612, 1994.

[8] A. Broder, A. Frieze, S. Suen and E. Upfal, *An efficient algorithm for the vertex-disjoint paths problem in random graphs*, In Proc. ACM-SIAM Symposium on Discrete Algorithms, pages 261–268, 1996.

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

[10] A. Broder, A. Frieze and E. Upfal, *Static and dynamic path selection on expander graphs: a random walk approach*, In Proc. ACM Symposium on the Theory of Computing, pages 531–539, 1997.

[11] A. Broder, A. Frieze and E. Upfal, Full version of [10]. (Personal communication from A. Frieze, June 1998.)

[12] P. Erdős and L. Lovász, *Problems and results on 3-chromatic hypergraphs and some related questions*, In *Infinite and Finite Sets*, A. Hajnal et al., eds., Colloq. Math. Soc. J. Bolyai 11, North Holland, Amsterdam, 1975, pp. 609–627.

[13] A. Frieze, *Disjoint paths in expander graphs via random walks: a short survey*, In Proc. Workshop on Randomization and Approximation Techniques in Computer Science, 1998.

[14] A. Frieze and L. Zhao, *Optimal construction of edge-disjoint paths in random regular graphs*, In Proc. ACM-SIAM Symposium on Discrete Algorithms, 1999.

[15] Z. Füredi and J. Kahn, *On the dimensions of ordered sets of bounded degree*, Order, 3 (1986), pp. 15–20.

[16] J. Kleinberg, *Approximation algorithms for disjoint paths problems*, PhD Thesis, Department of EECS, MIT, 1996.

[17] J. Kleinberg and R. Rubinfeld, *Short paths in expander graphs*, In Proc. IEEE Symposium on Foundations of Computer Science, pages 86–95, 1996.

[18] J. Kleinberg and É. Tardos, *Disjoint paths in densely embedded networks*, In Proc. IEEE Symposium on Foundations of Computer Science, pages 52–61, 1995.

[19] F. T. Leighton and S. B. Rao, *An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms*, In Proc. IEEE Symposium on Foundations of Computer Science, pages 422–431, 1988.

[20] F. T. Leighton and S. B. Rao, *Circuit switching: a multi-commodity flow based approach*, In Proc. Workshop on Randomized Parallel Computing, 1996.

[21] F. T. Leighton, S. B. Rao, and A. Srinivasan, *Multi-commodity flow and circuit switching*, In Proc. Hawaii International Conference on System Sciences, 1998.

[22] M. Lomonosov, *Combinatorial approaches to multiflow problems*, Discrete Applied Math., 11 (1985), pp. 1–94.

[23] C.-J. Lu, *A deterministic approximation algorithm for a minmax integer programming problem*, In Proc. ACM-SIAM Symposium on Discrete Algorithms, 1999.

[24] A. Lubotzky, R. Phillips and P. Sarnak, *Ramanujan graphs*, Combinatorica, 8 (1988), pp. 261–277.

[25] M. Middendorf and F. Pfeiffer, *On the complexity of the disjoint paths problem*, Combinatorica, 13 (1993), pp. 97–107.

[26] M. Molloy and B. Reed, *Further algorithmic aspects of the Local Lemma*, In Proc. ACM Symposium on Theory of Computing, pages 524–529, 1998.

[27] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.

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

[29] P. Raghavan and C. D. Thompson, *Randomized rounding: a technique for provably good algorithms and algorithmic proofs*, Combinatorica, 7 (1987), pp. 365–374.

[30] J. P. Schmidt, A. Siegel, and A. Srinivasan, *Chernoff-Hoeffding bounds for applications with limited independence*, SIAM Journal on Discrete Mathematics, 8 (1995), pp. 223–250.

[31] A. Schrijver, *Homotopic routing methods*, In *Paths, Flows, and VLSI-Layout*, B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, Editors, pages 329–371, Springer-Verlag, Berlin, 1990.

[32] P. D. Seymour, *On odd cuts and planar multicommodity flows*, J. London Math. Soc., 42 (1981), pp. 178–192.

[33] J. H. Spencer, *Ten Lectures on the Probabilistic Method*, SIAM, Philadelphia, 1987. (Second edition published in 1994.)

[34] J. H. Spencer, *The probabilistic lens: Sperner, Turán and Brégman revisited*, In *A Tribute to Paul Erdős*, A. Baker, B. Bollobás, A. Hajnal, editors, Cambridge University Press, pages 391–396, 1990.

[35] A. Srinivasan, *An extension of the Lovász Local Lemma, and its applications to integer programming*, In Proc. ACM-SIAM Symposium on Discrete Algorithms, pages 6–15, 1996.

[36] E. Upfal, *An* $O(\log N)$ *deterministic packet routing scheme*, In Proc. ACM Symposium on the Theory of Computing, pages 241–250, 1989.

A Previous constructive approaches to the LLL.

We now give a brief sketch of an approach of [26], which generalizes the basic algorithm of [6, 2] to cover many new applications. The reader is referred to these three beautiful papers for more details. Suppose we have a set $\mathcal{F} = \{f_1, f_2, \ldots, f_t\}$ of *independent* random variables, each f_i taking values in some domain of cardinality at most γ. Suppose we also have ("bad") events $E_1, E_2, \ldots, E_m$, such that each E_i is determined by the values of the variables in some $F_i \subseteq \mathcal{F}$. Define $u \doteq \max_i |F_i|$. Let us say that E_i *intersects* E_j iff $F_i \cap F_j \neq \emptyset$; we let D denote the maximum number of other E_j that any given E_i intersects. As mentioned in [26], we can typically (and certainly in all our applications) assume that $u \leq D$. Finally, let $p = \max_i \Pr[E_i]$. We need some further conditions on the efficient (exact or sufficiently good approximate) computability of certain conditional probabilities, all of which hold in our contexts. So, omitting these, one of the main results of [26] is that if

$$pD^9 < 1/8, \tag{1.12}$$

then we can find an assignment for all the f_i in randomized $\mathrm{poly}(m, t, \gamma^{uD \log\log m})$ time such that $\bigwedge_{i \in [m]} \overline{E_i}$ holds.

We now sketch this algorithm. A setting $\mathcal{A}$ of some of the random variables f_i is done in a way such that

$$\Pr[E_i \mid \mathcal{A}] \leq p^{2/3} \tag{1.13}$$

for all i. More importantly, the problem of setting the remaining f_i gets split into different "components", where each component has at most $D \log m$ events E_i whp. (We can handle each component separately.) An idea of [2] is useful now. If $D(\log m)p^{2/3} \leq 1/2$, say, then we have by (1.13) that in each component, a random setting will avoid all the corresponding E_i with probability at least 1/2 (which can be boosted as usual by repetition). So suppose $D(\log m)p^{2/3} > 1/2$; by (1.12), we see that $D \leq O((\log m)^{1/5})$. In this case, we repeat the above process and end up whp with components of size $O(D \log\log m)$. The LLL and (1.12) show that there is a setting of the variables in each component such that all the E_i are avoided. The helpful point now is that by definition of u, the number of f_i in any given component is at most $O(uD \log\log m)$; since $u \leq D \leq O((\log m)^{1/5})$ as seen above, this is at most $\ell = O((\log m)^{2/5} \log\log m)$. Since ℓ is so small, we can do exhaustive search over all possible settings of the f_i; by definition of γ, this takes time $\mathrm{poly}(m, t, \gamma^{uD \log\log m}) \leq \mathrm{poly}(m, t, \gamma^{(\log m)^{2/5} \log\log m})$ time overall.

In particular, if γ is "small", say bounded by a constant which is true for many applications of the LLL, the condition (1.12) that is somewhat stronger than the "$ep(D+1) \leq 1$" of the LLL, ensures a constructive version of the LLL. However, as pointed out in [26], if γ is "large", say $(t+m)^{\Theta(1)}$, this method does not guarantee polynomial-time algorithms.

B Comparison of Theorem 2.1 with previous methods.

Suppose we were to follow the methodology of [6, 2, 26] to prove Theorem 2.1. The reader is referred to Appendix A for a sketch of this approach. Motivated by the notation of Appendix A, let us define γ to be the maximum number of paths in any (multi-)set $\mathcal{P}_i$. In Theorem 2.1, consider the case where we wish to choose one path from each $\mathcal{P}_i$; i.e., $p = \Theta(1/k)$. Combined with the work of [21], the approach of Appendix A would require that the fractional congestion in Theorem 2.1 be $O(1/d\gamma^{7/9})$. (In contrast, Theorem 2.1 needs a fractional congestion of $O(1/(d \ln k))$.) While the exponent 7/9 can be reduced somewhat, it seems hard to reduce it below 2/3, and some basic features of the approaches of [6, 2, 26] make it potentially impossible to reduce the exponent below a certain positive constant. Thus, situations where γ is "large" pose problems if we use the approach of [6, 2, 26]. Via the approach of step 1 of our algorithm for Theorem 2.1, we can reduce γ to $O(d \log n)$, but reducing γ to $o(d)$, say, is a challenging open problem. More importantly, as pointed out in [26] and in Appendix A, situations where γ is "large", say $n^{\Theta(1)}$, do not admit of polynomial-time solutions via the approach of [6, 2, 26]. Thus, for instances where $\min\{\gamma, d\}$ is, say, $n^{\Theta(1)}$, it is not clear how to use the methods of [6, 2, 26] to derive polynomial-time algorithms.