Inapproximability of edge-disjoint paths and low congestion routing on undirected graphs

Matthew Andrews* Julia Chuzhoy† Venkatesan Guruswami‡
Sanjeev Khanna§ Kunal Talwar¶ Lisa Zhang*

August 31, 2009

Abstract

In the undirected Edge-Disjoint Paths problem with Congestion (EDPwC), we are given an undirected graph with V nodes, a set of terminal pairs and an integer c. The objective is to route as many terminal pairs as possible, subject to the constraint that at most c demands can be routed through any edge in the graph. When $c = 1$, the problem is simply referred to as the Edge-Disjoint Paths (EDP) problem. In this paper, we study the hardness of EDPwC in undirected graphs.

Our main result is that for every $\varepsilon > 0$ there exists an $\alpha > 0$ such that for $1 \leqslant c \leqslant \frac{\alpha \log\log V}{\log\log\log V}$, it is hard to distinguish between instances where we can route all terminal pairs on edge-disjoint paths, and instances where we can route at most a $1/(\log V)^{\frac{1-\varepsilon}{c+2}}$ fraction of the terminal pairs, even if we allow congestion c. This implies a $(\log V)^{\frac{1-\varepsilon}{c+2}}$ hardness of approximation for EDPwC and an $\Omega(\log\log V/\log\log\log V)$ hardness of approximation for the undirected congestion minimization problem. These results hold assuming NP $\not\subseteq \cup_d$ZPTIME$(2^{\log^d n})$.

In the case that we do not require perfect completeness, i.e. we do not require that all terminal pairs are routed for "yes-instances", we can obtain a slightly better inapproximability ratio of $(\log V)^{\frac{1-\varepsilon}{c+1}}$. Note that by setting $c = 1$ this implies that the regular EDP problem is $(\log V)^{\frac{1}{2}-\varepsilon}$ hard to approximate.

Using standard reductions, our results extend to the node-disjoint versions of these problems as well as to the directed setting. We also show a $(\log V)^{\frac{1-\varepsilon}{c+1}}$ inapproximability ratio for the All-or-Nothing Flow with Congestion (ANFwC) problem, a relaxation of EDPwC, in which the flow unit routed between the source-sink pairs does not have to follow a single path, so the resulting flow is not necessarily integral.

Mathematics Subject Classification (2000) codes: 68Q17, 05C38.
Abbreviated title: Edge-disjoint paths in undirected graphs

*Bell Laboratories, Lucent Technologies, Murray Hill, NJ. {andrews,ylz}@research.bell-labs.com

†Toyota Technological Institute, Chicago, IL 60637. Email: cjulia@tti-c.org

‡**Corresponding author.** Current address: Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213. Email: guruswami@cmu.edu. This work was done when the author was at the Dept. of Computer Science and Engineering, University of Washington. Research supported in part by NSF CCF-0343672, and Sloan and Packard fellowships.

§Dept. of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104. Email: sanjeev@cis.upenn.edu. Supported in part by an NSF Career Award CCR-0093117 and by NSF Award CCF-0635084.

¶Microsoft Research, Silicon Valley Campus, Mountain View, CA 94043. Email: kunal@microsoft.com

1 Introduction

In the *edge-disjoint paths* (EDP) problem we are given a graph G and a set $\{(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)\}$ of pairs of vertices called terminals. The objective is to connect as many pairs as possible via edge-disjoint paths. Even highly restricted cases of EDP correspond to well-studied important optimization problems. For instance, EDP on stars is equivalent to the graph matching problem. EDP and its variants also have a host of applications to network routing, resource allocation, and VLSI design. It is then not surprising that EDP is one of the most well-studied problems in combinatorial optimization. In directed graphs, the problem becomes NP-hard even when we are given only two source-sink pairs [21]. In undirected graphs, the seminal work of Robertson and Seymour [36] gives a polynomial time algorithm for any constant number of pairs. These results are suggestive of the inherent differences between the undirected and directed versions of EDP. However, the tractability of undirected EDP with constant number of pairs does not hold once the number of pairs is allowed to grow as a function of the input size. In particular, the problem is NP-hard even on planar graphs [23].

Consequently, much of the recent work on EDP has focused on understanding the polynomial-time approximability of the problem. While constant or poly-logarithmic approximation algorithms are known for restricted classes of graphs such as trees, meshes, and expanders [7, 15, 20, 24, 29, 30], the approximability of EDP in general graphs is not well understood. The best approximation algorithm for EDP in directed graphs has a ratio of $\tilde{O}(\min(V^{2/3}, \sqrt{E}))$ [14, 31, 32, 38, 39] where V and E denote the number of vertices and edges respectively in the input graph. For undirected graphs and directed acyclic graphs, this factor improves to an $O(\sqrt{V})$-approximation ratio [13]. In directed graphs, the approximation ratio is matched by an $\Omega(E^{\frac{1}{2}-\varepsilon})$-hardness due to Guruswami *et al.* [25]. In contrast, only APX-hardness was known for undirected EDP until the work of [3] which showed an $\Omega(\log^{\frac{1}{3}-\varepsilon} V)$ hardness, unless NP $\subseteq$ ZPTIME$(n^{\mathrm{poly}(\log n)})$.

In this paper we study undirected EDP together with a natural generalization known as *edge-disjoint paths with congestion* (EDPwC), in which the goal is to route as many terminal pairs as possible subject to the constraint that at most c paths are routed through any edge. For directed and undireted graphs with constant congestion $c \geqslant 2$, there exists an $O(V^{1/c})$ approximation [8, 9, 32]. When the congestion is allowed to be $O(\log V / \log\log V)$ we get a constant approximation via randomized rounding [34]. For undireted planar graphs, when congestion 2 is allowed, an $O(\log V)$-approximation has recently been derived [11, 12]. We note that the performance of an approximation algorithm for EDPwC is measured with respect to an optimal solution with no congestion.

Another related problem is the *all-or-nothing* (ANF) flow problem where each routed demand is allowed to be routed on fractional paths. Thus ANF is a relaxation of EDP. Recent work has shown that in undirected graphs, ANF is $O(\log^2 V)$-approximable [10, 12]. The $\Omega(\log^{\frac{1}{3}-\varepsilon} V)$ hardness result in [3] extends to ANF as well. We also study the variant of ANF where congestion is allowed, referred to as *ANF with Congestion* (ANFwC).

The last problem that we discuss is the *Congestion Minimization Problem* (CMP) in which the goal is to find the minimum value of the congestion c such that all terminal pairs can be routed. The randomized rounding result of [34] implies that CMP can be approximated to within a factor $\Omega(\log V / \log\log V)$. In [4], it was shown that undirected CMP is hard to approximate to within

a factor $(\log\log V)^{1-\varepsilon}$. For directed graphs [6] showed a hardness ratio of $(\log V)^{1-\varepsilon}$. This was improved to the tight bound of $\Omega(\log V/\log\log V)$ in [17].

1.1 Our results

Our main results are as follows.

Theorem 1. *Consider an undirected graph. For every $\varepsilon > 0$ there exists $\alpha > 0$ such that for every integer-valued function $c : \mathbb{N} \to \mathbb{N}$ satisfying $1 \leqslant c(n) \leqslant \frac{\alpha \log\log n}{\log\log\log n}$ (and computable in time polynomial in n), it is impossible to distinguish in randomized polynomial time between the following cases:*

- [YES INSTANCES:] *There are edge-disjoint paths connecting all the terminal pairs.*
- [NO INSTANCES:] *With congestion $c = c(V)$, at most a $1/(\log V)^{\frac{1-\varepsilon}{c+2}}$ fraction of the terminal pairs can be routed.*

unless NP $\subseteq$ ZPTIME$(n^{\mathrm{poly}(\log n)})$.[1] *Hence undirected EDPwC is hard to approximate to within a factor $(\log V)^{\frac{1-\varepsilon}{c+2}}$ under the above complexity assumption.*

An important feature of the reduction implied by the above result is that it has perfect completeness, i.e. all terminal pairs are routed in the case of yes-instances. If we do not require perfect completeness then we obtain the following slightly stronger gap.

Theorem 2. *The reduction of Theorem 1 can be adapted so that for some parameter f it is impossible to distinguish in randomized polynomial time between the following cases:*

- [YES INSTANCES:] *There are edge-disjoint paths connecting an f-fraction of the terminal pairs.*
- [NO INSTANCES:] *At most an $f/(\log V)^{\frac{1-\varepsilon}{c+1}}$ fraction of the terminal pairs can be routed with congestion $c = c(V)$.*

unless NP $\subseteq$ ZPTIME$(n^{\mathrm{poly}(\log n)})$. *Hence undirected EDPwC is hard to approximate to within a factor $(\log V)^{\frac{1-\varepsilon}{c+1}}$ under the above complexity assumption.*

Note that in the above theorems n is an abstract parameter used to define the complexity class ZPTIME$(n^{\mathrm{poly}(\log n)})$. Recall that ZPTIME$(n^{\mathrm{poly}(\log n)})$ is the class of languages for which there is a randomized algorithm that always gives the correct answer and whose expected running time is $n^{\mathrm{poly}(\log n)}$. Since our reduction is quasipolynomial, n is not V, the size of the EDPwC instance.

We remark that simple modifications to the reductions used in the proofs of Theorems 1 and 2 immediately imply analogous hardness results for the Node-Disjoint Paths with Congestion problem in which the congestion constraint applies to the vertices of the graph instead of the edges. In

[1] The reduction will naturally have one-sided error, immediately giving hardness under the assumption that NP $\not\subseteq$ coRPTIME$(n^{\mathrm{poly}(\log n)})$. It is a standard fact that this assumption is implied by NP $\not\subseteq$ ZPTIME$(n^{\mathrm{poly}(\log n)})$; see Lemma 27.

addition, we also show that the result of Theorem 2 can be extended to give a hardness of $(\log V)^{\frac{1-\varepsilon}{2c}}$ for the ANFwC problem.

We now highlight two consequences of the above theorems that we believe are of particular interest. First, the perfect completeness of Theorem 1 means that it implies a gap in the amount of congestion required to connect all terminal pairs.

Lemma 3. *It is impossible to distinguish in randomized polynomial time between the following cases:*

- [YES INSTANCES:] *There are edge-disjoint paths connecting all the terminal pairs.*
- [NO INSTANCES:] *With congestion $c = \Omega(\log\log V/\log\log\log V)$, not all terminal pairs can be routed.*

unless NP $\subseteq$ ZPTIME($n^{polylog\,(n)}$). *Hence undirected CMP is hard to approximate to within a factor* $\Omega(\log\log V/\log\log\log V)$.

This improves the $(\log\log V)^{1-\varepsilon}$ hardness of [4]. Second, by setting $c = 1$ in Theorem 2 and in our result for ANFwC, we obtain the following hardness results for regular EDP and ANF.

Corollary 4. *For every constant $\varepsilon > 0$, undirected EDP and ANF are both hard to approximate to within a factor* $(\log V)^{\frac{1}{2}-\varepsilon}$.

This improves the $(\log V)^{\frac{1}{3}-\varepsilon}$ hardness of [3]. These corollaries imply that our results capture (and improve) in a *unified* way hardness results for EDP, ANF, EDPwC and CMP that were presented in the sequence of papers [3, 4, 2]. In particular, we believe that the current proof unifies the previous work also in terms of the proof techniques, by basing the reduction from the "correct" general constraint satisfaction problem — one defined over a large (non-Boolean) domain and that is hard with perfect completeness and near-optimal amortized query complexity [37, 27].

We also present a simple family of instances that shows that the integrality gap of the well-studied multicommodity flow relaxation is $(\log V)^{\Omega(1/c)}$ for both EDPwC and ANFwC.

Theorem 5. *For any congestion $2 \leqslant c \leqslant O\left(\frac{\log\log V}{\log\log\log V}\right)$, the integrality gap of the multicommodity flow relaxation for undirected EDPwC is $\Omega\left(\frac{1}{c}\cdot(\frac{\log V}{(\log\log V)^2})^{\frac{1}{c+1}}\right)$. For ANFwC, the integrality gap with congestion c is $\Omega\left(\frac{1}{c^2}\cdot(\frac{\log V}{(\log\log V)^2})^{\frac{1}{c+1}}\right)$. In particular, there exists a congestion $c = \Theta\left(\frac{\log\log V}{(\log\log\log V)^2}\right)$ for which the integrality gaps for both problems remain superconstant.*

We note that an immediate consequence of Theorem 5 is that for any fixed integer i, the gap between $(1/i)$-integral multicommodity flow (i.e. each flow path carries an integral multiple of $1/i$ units of flow) and fractional multicommodity flow is super-constant in undirected graphs. To our knowledge, prior to our work, it was not known if there was a superconstant gap even between half-integral flow and fractional flow in directed or undirected graphs. The instances used in establishing the integrality gap have a surprisingly simple structure.

1.2 Overview of Techniques

This paper represents a merging of three papers [26, 5, 16]. A preliminary version containing results from [5, 16] appeared in a conference paper [2]. The hardness construction presented here is based on [26] and may be seen as unifying all three constructions.

Our hardness construction is based on a hard-to-approximate Constraint Satisfaction Problem (CSP) due to [27] (and thus avoids an intermediate step taken by [3] of creating an independent set instance). The high-level idea of the reduction is as follows. In the CSP instance Ψ we start from, we have a set of variables and a collection of constraints on certain k-tuples of variables, each of which constrains the set of possible assignments to the variables in that k-tuple. The CSP has the property that given such an instance, it is hard to tell if it is satisfiable (i.e., some assignment satisfies *all* the constraints), or it is "very" unsatisfiable (in the sense that every assignment satisfies only a tiny fraction of the constraints — we skip the exact parameters for this discussion).

From such a CSP instance, we construct a graph G_Ψ which contains a sufficiently large collection of edge-disjoint paths for each (constraint, satisfying assignment) pair (call this a "local accepting configuration"). These paths are referred to as the *canonical paths* of the corresponding local accepting configuration. The canonical path collections for any two local accepting configurations that disagree on the value they assign to some common variable are made to *randomly intersect* with each other. The goal is to encode conflicting values to a variable by a gadget which ensures that low congestion routings cannot be based on too many conflicting values to variables. This implies that such routings can be "decoded" into a good assignment to the original CSP.

The graph G_Ψ serves as the input graph for the EDPwC instance. The source-sink pairs are formed by grouping together end-points of the canonical path collections corresponding to each constraint (over all its satisfying assignments). The idea is that (ideally) each source-sink pair picks a path corresponding to a satisfying assignment for the corresponding constraint. If the instance Ψ is satisfiable, by picking such paths, we can route all source-sink pairs on edge-disjoint paths. If very few constraints of Ψ are satisfiable, the construction ensures that if pairs are indeed routed on such "ideal" canonical paths, those paths must collide with high probability and create high congestion (because they must assign too many inconsistent values to some variable). However, these conflicts can be avoided if source-sink pairs are connected via paths that are not canonical. In order to deal with such non-canonical paths we employ an idea from [1] and construct the random graph G_Ψ so that on average, paths that deviate significantly from canonical paths are much longer than canonical paths and thus consume much more of the routing capacity of the graph. Therefore, there cannot be too many source-sink pairs routed on such non canonical paths. Together with the bound on number of pairs routed on (almost) canonical paths, we conclude that when only a small fraction of the constraints of Ψ are satisfiable, with high probability (over the randomness in constructing G_Ψ), only a small fraction of source-sink pairs can be routed in the graph G_Ψ (even if we allow some bounded congestion).

1.3 Organization

In Section 2, we present a simple integrality gap construction that establishes Theorem 5. Sections 3 and 4 present the hardness construction and analysis underlying Theorem 1 and Corollary 3. In Section 5, we establish Theorem 2 and Corollary 4 as well as describe an extension of our hardness results to node-disjoint paths problem.

2 Integrality Gap of the Multicommodity Flow Relaxation

In this section, we will show a family of instances that realize the integrality gap results stated in Theorem 5. The instances will be characterized by a parameter n, and we will construct, for each integer $c \leqslant O((\log\log n)/(\log\log\log n))$, an EDP instance of size $O(n \log n)$ for which the integrality gap is $\Omega((\frac{\log n}{(\log\log n)^2})^{1/c})/c)$ when congestion is restricted to be *strictly less than c*. Our construction will use two additional parameters, $\beta_1 = \frac{1}{4}\left(\frac{\log n}{150(\log\log n)^2}\right)^{1/c}$ and $\beta_2 = 6(2\beta_1)^{c-1}\ln\beta_1$. The integrality gap of our EDP instance will be $\Omega(\beta_1/c)$. Towards the end, we sketch how these results extend to ANF with congestion.

2.1 The Multicommodity Flow Relaxation

We start by presenting the standard multicommodity flow relaxation for EDP (ANF). Given a graph G and a set $\{(s_1,t_1),(s_2,t_2),\ldots,(s_k,t_k)\}$ of source-sink pairs, let $\mathcal{P}^{(i)}$ denote the set of all paths joining s_i and t_i in G. Also, let $\mathcal{P} = \cup_i \mathcal{P}^{(i)}$. The multicommodity flow relaxation uses two variables: (i) a variable $f(P)$ for each path $P \in \mathcal{P}$ that gives the amount of flow sent on P, and (ii) a variable x_i that indicates the total flow routed for the pair (s_i,t_i). We let $\bar{f}$ denote the flow vector with a component for each path P, and we denote by $|\bar{f}|$ the value $\sum_i x_i$. Then the LP relaxation for EDP (ANF) is as following.

$$\begin{aligned}
\max \sum_{i=1}^{k} x_i \quad & \text{s.t} \\
x_i - \sum_{P\in\mathcal{P}^{(i)}} f(P) &= 0 \qquad \forall i : 1 \leqslant i \leqslant k \\
\sum_{P:e\in P} f(P) &\leqslant 1 \quad \forall e \in E \\
x_i, f(P) &\in [0,1] \quad \forall i : 1 \leqslant i \leqslant k, P \in \mathcal{P}
\end{aligned}$$

When considering the integrality gaps for EDPwC and ANFwC, we restrict the fractional solution to obey the same constraints as above, but allow the integral solution to route up to $(c-1)$ paths through any edge e.

2.2 Auxiliary Hypergraph Construction

Our starting point is a random hypergraph H with vertex set $V(H) = \{v_1,\ldots,v_n\}$, and $\beta_2 n$ hyperedges, $h_1,\ldots,h_{\beta_2 n}$. Each hyperedge h_i, for $1 \leqslant i \leqslant \beta_2 n$, is a subset of c vertices chosen randomly and independently. Our EDP instance will be derived from the hypergraph H.

We now establish some properties of H. A set $S \subseteq V(H)$ of size n/β_1 is said to be a *bad* set if it contains none of the $\beta_2 n$ hyperedges in H. We say that *event* $\mathcal{E}_1$ *occurs* if there is at least one bad subset $S \subseteq V(H)$ of size n/β_1.

Lemma 6. *The probability that the event* $\mathcal{E}_1$ *occurs is at most* $1/4$.

Proof. Fix a set $S \subseteq V(H)$ of size n/β_1. The probability that a random hyperedge is contained in S is:

$$\frac{\binom{n/\beta_1}{c}}{\binom{n}{c}} = \frac{\frac{n}{\beta_1} \cdot \left(\frac{n}{\beta_1} - 1\right) \cdots \left(\frac{n}{\beta_1} - c + 1\right)}{n \cdot (n-1) \cdots (n-c+1)} \geqslant \left(\frac{\frac{n}{\beta_1} - c}{n}\right)^c \geqslant \frac{1}{(2\beta_1)^c}$$

since $c \leqslant n/(2\beta_1)$ for sufficiently large n.

Therefore,

$$\mathbf{Pr}[S \text{ is bad}] \leqslant \left(1 - \frac{1}{(2\beta_1)^c}\right)^{\beta_2 n} \leqslant e^{-\frac{\beta_2 n}{(2\beta_1)^c}}$$

Since the number of possible choices for the set S is $\binom{n}{n/\beta_1}$ which can be upper-bounded by $(e\beta_1)^{n/\beta_1} \leqslant \beta_1^{2n/\beta_1}$, using the union bound, we get that the probability that any set S of size n/β_1 is bad, is at most:

$$\beta_1^{\frac{2n}{\beta_1}} \cdot e^{\frac{-\beta_2 n}{(2\beta_1)^c}} \leqslant e^{\frac{n}{\beta_1}(2\ln\beta_1 - \frac{\beta_2}{2^c \beta_1^{c-1}})} = e^{-\frac{n \ln \beta_1}{\beta_1}} \leqslant \frac{1}{4}$$

□

Given a vertex $v \in V(H)$, we say that it is a *high-degree* vertex, if it participates in more than $10\beta_2 c$ hyperedges in H. We say that the *event* $\mathcal{E}_2$ *occurs*, if the number of high-degree vertices in H is greater than n/β_1. Using Chernoff bounds, we can bound the probability of this event as follows.

Lemma 7. *The probability that the event* $\mathcal{E}_2$ *occurs is at most* $1/4$.

Proof. A vertex v occurs in a random subset of size c with probability c/n. Thus the expected number of hyperedges in which a vertex is contained is $\beta_2 c$. By Chernoff bounds, for any $\delta \geqslant 2e-1$, the probability that a vertex is contained in more than $(1+\delta)\beta_2 c$ hyperedges can be bounded by $2^{-(1+\delta)\beta_2 c}$. Thus the probability that a vertex is contained in more than $10\beta_2 c$ hyperedges is at most $1/(4\beta_1)$.

Hence the expected number of high degree vertices is at most $n/(4\beta_1)$. By Markov's inequality, the probability that there are more than n/β_1 such vertices is at most $\frac{1}{4}$.

□

Thus with probability at least $\frac{1}{2}$, neither event $\mathcal{E}_1$ nor event $\mathcal{E}_2$ occurs in the random hypergraph H created above.

2.3 Integrality Gap Instances

The construction of the integrality gap instances for EDPwC and ANFwC is based on the hypergraph H defined above. The family of gap instances is identical for both problems. We focus on EDPwC, and describe at the end of this section the analysis for ANFwC.

The EDPwC instance is defined on a graph G constructed as follows. For each hyperedge $h_i : 1 \leqslant i \leqslant \beta_2 n$, it contains two vertices ℓ_i, r_i, which are connected by a *special edge*. Consider now some vertex $v \in V$, and assume that it participates in hyperedges $h_{i_1}, h_{i_2}, \ldots, h_{i_k}$, where $i_1 < i_2 < \cdots < i_k$. We add the following *regular edges* to graph G: $(s(v), \ell_{i_1})$, $(r_{i_k}, t(v))$, and for each $j : 1 \leqslant j \leqslant k-1$, we add a regular edge $(r_{i_j}, \ell_{i_{j+1}})$. Finally, for each vertex $v \in V(H)$, the EDP instance contains a source-sink pair $(s(v), t(v))$. For each source-sink pair $s(v)$-$t(v)$, we define a *canonical path* as follows: $P(v) = (s(v), \ell_{i_1}, r_{i_1}, \ldots, \ell_{i_k}, r_{i_k}, t(v))$. Note that the canonical path $P(v)$ traverses all hyperedges containing v in a monotonically increasing order.

Properties of the EDPwC Instance: We now establish that with high probability, the instance created above satisfies a set of properties that would allow us to establish the integrality gap.

Let G' be a graph obtained from G by shrinking each special edge (ℓ_i, r_i) into a vertex u_i. Note that each edge in G' corresponds to a unique and distinct regular edge in G. In particular, an edge (u_{i_1}, u_{i_2}) in G' maps to an edge (r_{i_1}, ℓ_{i_2}) if $i_1 < i_2$, and to edge (ℓ_{i_1}, r_{i_2}) otherwise. Let $g > 2$ be a *fixed integer* and let K_g be the total number of (simple) cycles of length at most g in G. We say that the *event* $\mathcal{E}_3$ *occurs* if $K_g > (6\beta_2 c^2)^{g+1}$. In the following Lemma 8 we first show a property that holds for any $g > 2$. In the subsequent section, we define a desirable value of g.

Lemma 8. *The probability that the event $\mathcal{E}_3$ occurs is at most $\frac{1}{4}$ for any fixed integer value of $g > 2$.*

Proof. A cycle C of length k in the graph G' corresponds to an ordered sequence of k vertices $u_{i_1}, \ldots, u_{i_k}$, where $i_k = \max\{i_1, \ldots, i_k\}$, and edges $e_1 = (u_{i_1}, u_{i_2}), \ldots, e_{k-1} = (u_{i_{k-1}}, u_{i_k}), e_k = (u_{i_k}, u_{i_1})$ belong to the cycle C in G'. Note that the indices $i_1, i_2, \ldots, i_k$ need not form an increasing sequence; they only satisfy the property that i_k is the largest integer in the sequence.

For each $j \in [1, \ldots, (k-2)]$, we first bound the probability that the edge e_j exists given the existence of edges $e_1, \ldots, e_{j-1}$. Let $A \subseteq V(H)$ be the subset of vertices that defines the hyperedge h_{i_j}. Note that $|A| = c$. If the edge e_j exists, then hyperedge $h_{i_{j+1}}$ must contain at least one vertex from A. The probability of this happening (given the existence of $e_1, \ldots, e_{j-1}$) is at most $\frac{c^2}{n}$.

We now bound the probability of edges e_k, e_{k-1} belonging to G', given the existence of $e_1, \ldots, e_{k-2}$. Consider the hyperedges $h_{i_1}, h_{i_{k-1}}$ of graph H. Let X, Y, Z be pair-wise disjoint subsets of $V(H)$ defined as follows: $X = h_{i_1} \setminus h_{i_{k-1}}$, $Y = h_{i_1} \cap h_{i_{k-1}}$, and $Z = h_{i_{k-1}} \setminus h_{i_1}$. We claim that the hyperedge h_{i_k} must have a non-empty intersection with at least two sets among X, Y, and Z. For instance, if the hyperedge h_{i_k} has a non-empty intersection only with the set Y but is disjoint from both X and Z, then at least one of the edges e_{k-1}, e_k does not belong to G'. This follows from the fact that for any vertex $v \in V(H)$ the canonical path of v traverses the hyperedges of H in a monotonically increasing order. So for each vertex $v \in Y$, the canonical path of v in G' visits $u_{i_1}, u_{i_{k-1}}, u_{i_k}$ in this order, and so edge (u_{i_1}, u_{i_k}) does not belong to G'. Similarly, we can rule out the possibility that the hyperedge h_{i_k} has a non-empty intersection only with X or only with Z. Thus in order for edges e_{k-1}, e_k to belong to G', the hyperedge h_{i_k} must intersect with at least two out of the three sets X, Y, Z. We bound the probability that it intersects with both X and Y. The probabilities of h_{i_k} intersecting with X and Z, and with Y and Z, can be bounded similarly.

Let $\mathcal{E}_X$, $\mathcal{E}_Y$ denote the events that $h_{i_k} \cap X \neq \emptyset$ and $h_{i_k} \cap Y \neq \emptyset$, respectively. Then

$$\begin{aligned}\mathbf{Pr}\left[\mathcal{E}_X \wedge \mathcal{E}_Y | e_1, \ldots, e_{k-1}\right] &= \mathbf{Pr}\left[\mathcal{E}_X | \mathcal{E}_Y, e_1, \ldots, e_{k-1}\right] \cdot \mathbf{Pr}\left[\mathcal{E}_Y | e_1, \ldots, e_{k-1}\right] \\ &\leqslant \mathbf{Pr}\left[\mathcal{E}_X | e_1, \ldots, e_{k-1}\right] \cdot \mathbf{Pr}\left[\mathcal{E}_Y | e_1, \ldots, e_{k-1}\right] \leqslant \frac{c^4}{n^2},\end{aligned}$$

where the inequality $\mathbf{Pr}\left[\mathcal{E}_X | \mathcal{E}_Y, e_1, \ldots, e_{k-1}\right] \leqslant \mathbf{Pr}\left[\mathcal{E}_X | e_1, \ldots, e_{k-1}\right]$ follows from the fact that $X \cap Y = \emptyset$, and so the hyperedge h_{i_k} is less likely to contain a vertex from X if it already contains a vertex from Y. Therefore, the total probability that both edges e_{k-1}, e_k belong to G' is at most $3\left(\frac{c^4}{n^2}\right)$, and the probability that cycle C of length k belongs to G' is at most $3\left(\frac{c^2}{n}\right)^k$.

The number of possible cycles of length k can be bounded by $(\beta_2 n)^k$. Thus the expected number of cycles of length k is at most $(3\beta_2 c^2)^k$. Summing up over all $k \in [3, \ldots, g]$, we get that $E[K_g] \leqslant (3\beta_2 c^2)^{g+1}$ for any integer $g > 2$, and using Markov's inequality, we get the claimed bound.

□

2.4 Integrality Gap Analysis

Let $g = 3\beta_1\beta_2 c^2$. By Lemmas 6, 7, and 8, with probability at least 1/4, none of the events $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}_3$ (for value $2g$) occur. We assume this from now on, and establish the integrality gap.

The fractional solution can route at least $\frac{n}{c}$ units of flow, by sending $\frac{1}{c}$ units of flow on each canonical path. This gives us a fractional solution of value (n/c) with congestion 1. Note that this is a feasible fractional solution for both EDPwC and ANFwC.

Gap Analysis for EDPwC: Consider now some integral solution whose congestion is at most $c-1$, and let $\mathcal{P}'$ denote the set of paths on which the source-sink pairs are routed in this integral solution. We partition $\mathcal{P}'$ into three subsets: $\mathcal{P}_1$ contains canonical source-sink paths, $\mathcal{P}_2$ contains non-canonical source-sink paths whose length is at least g, and $\mathcal{P}_3$ contains non-canonical source-sink paths whose length is smaller than g. We bound the size of each one of these sets separately.

$|\mathcal{P}_1| \leqslant \frac{n}{\beta_1}$ if event $\mathcal{E}_1$ does not happen. Otherwise, there must be c paths that go through a single special edge and the solution has congestion c.

$|\mathcal{P}_2| \leqslant \frac{n}{\beta_1}$ since the total number of edges in G is at most $3\beta_2 cn$. Thus even allowing a congestion of c, total routing capacity available in the graph is at most $3\beta_2 c^2 n$. Thus the number of paths of length at least g can be no more than $\frac{3\beta_2 c^2 n}{g} = \frac{3\beta_2 c^2 n}{3\beta_1\beta_2 c^2} = \frac{n}{\beta_1}$.

To analyze $|\mathcal{P}_3|$, we first remove from $\mathcal{P}_3$ all paths that correspond to vertices which occur in more than $10\beta_2 c$ hyperedges of H. Since event $\mathcal{E}_2$ does not happen, we discard at most n/β_1 paths. Let $\mathcal{P}'_3 \subseteq \mathcal{P}_3$ be the set that remains. For any $s(v)$-$t(v)$ pair routed in $\mathcal{P}'_3$, the length of its canonical path is at most $10\beta_2 c$. Let $p_1(v)$ denote the non-canonical path used for routing the pair $s(v)$-$t(v)$ in $\mathcal{P}'_3$, and let $p_2(v)$ denote the canonical path for the pair $s(v)$-$t(v)$. We consider the union of paths $p_1(v)$ and $p_2(v)$ in the graph G' (recall that G' is obtained from G by shrinking each special edge (ℓ_i, r_i) into a vertex u_i). This union must contain a simple cycle of length at most $g + 10\beta_2 c \leqslant 2g$. By Lemma 8, the number of cycles of length at most $2g$ in G' can be bounded by $(6\beta_2 c^2)^{2g+2}$. Since each edge in G' corresponds to a unique edge in G, and each edge in G is allowed a congestion of up to $(c-1)$, it follows that $|\mathcal{P}'_3| \leqslant 2gc(6\beta_2 c^2)^{2g+2}$ as each path in $\mathcal{P}'_3$ uses at least one edge associated

with such a cycle. Thus

$$|\mathcal{P}'_3| \leqslant 2gc(6\beta_2 c^2)^{2g+2} \leqslant (\beta_2 c^2)^{3g} \leqslant 2^{4g\log\beta_2} = 2^{72(4\beta_1)^c \cdot \ln^2 \beta_1} \leqslant \sqrt{n} \leqslant n/\beta_1$$

In total, $|\mathcal{P}'| \leqslant 4n/\beta_1$, and the integrality gap is at least $\frac{\beta_1}{4c}$, giving the bound in Theorem 5.

Gap Analysis for ANFwC: To show an integrality gap for ANF with congestion, consider the subset of pairs routed in some feasible solution. Recall that for each pair $s(v)$-$t(v)$ chosen in a feasible solution for ANFwC, a unit of flow needs to be routed (possibly fractionally) from $s(v)$ to $t(v)$. We refer to a routed pair as a *canonical pair* if more than $(c-1)/c$-fraction of the flow is routed on the pair's canonical path, and call it a *non-canonical pair* otherwise. It is easy to see that no more than $(c-1)$ canonical pairs can traverse a special edge without causing a congestion greater than $(c-1)$. Thus essentially the same analysis as given above for $\mathcal{P}_1$ applies to canonical pairs. For non-canonical pairs, we proceed as above for $\mathcal{P}_2$ and $\mathcal{P}_3$ noting that for each routed pair, we now have only $(1/c)$-fraction of the flow to be supported on non-canonical paths. Thus we can bound the number of pairs routed in any feasible solution for ANFwC by $\frac{4cn}{\beta_1}$, and hence the integrality gap is at least $\frac{\beta_1}{4c^2}$, giving the bound in Theorem 5.

3 The Hardness Construction

We now establish Theorems 1 and 2 by using a randomized reduction from a general constraint satisfaction problem (CSP) over large (non-Boolean) domains.

3.1 A Hard-to-Approximate Constraint Satisfaction Problem

An instance of p-ary k-CSP consists of *variables* $\{x_1, \ldots, x_N\}$ that take values in $\{1, 2, \ldots, p\}$ and *constraints* $C_1, \ldots, C_M$ on tuples of the variables of size k. Formally, a constraint C_i is simply a tuple τ_i of k variables along with a subset $S_i \subseteq \{1, 2, \ldots, p\}^k$ of "satisfying assignments." The constraint is *satisfied* if the k variables in τ_i are assigned a tuple of k values that belongs to S_i. The goal is to find an assignment to the x_i's that satisfies as many constraints as possible. Both p and k can be functions of the number of variables N.

The starting point for our reduction to EDPwC is the following strong hardness result for constraint satisfaction (over any domain $\{1, 2, \ldots, p\}$ with p prime) due to Håstad and Khot [27] (stated as Theorem 9 below). The two features about this result that are important to us are (i) perfect completeness, and (ii) the maximum number of satisfying assignments to any constraint is much smaller than the reciprocal of the soundness (below, the former is at most $p^{10\sqrt{\ell}}$ while the latter is at least $p^{\ell - 10\sqrt{\ell}}$, and for large ℓ the ratio of the logarithm for these quantities tends to 0). The result of Samarodnitsky and Trevisan [37], which was extended to larger, non-Boolean domains by Engebretsen [18], achieves the second property above but lacks perfect completeness.

Theorem 9. *For all large enough integers ℓ and every function $p : \mathbb{N} \to \mathbb{N}$ that takes prime values, there is a reduction from 3SAT to p-ary ℓ-CSP with the following properties:*

1. [Reduction complexity]: *For a 3SAT instance of size n, the reduction runs in time $n^{O(\ell \log p)} \cdot p^{p^{O(\ell)}}$, and produces a p-ary ℓ-CSP instance with at most $n^{O(\ell \log p)} \cdot p^{p^{O(\ell)}}$ vari-*

ables and constraints, where each constraint in the instance has at most $p^{10\sqrt{\ell}}$ satisfying assignments. (Here we use the shorthand $p = p(n)$.)

2. [PERFECT COMPLETENESS]: *If the original 3SAT instance is satisfiable, the p-ary ℓ-CSP instance is also satisfiable.*

3. [SOUNDNESS]: *If the original SAT instance is not satisfiable, then at most a fraction $p^{-\ell+10\sqrt{\ell}}$ of the constraints in the p-ary ℓ-CSP instance are satisfiable.*

While the above gives a quasi-polynomial time reduction for $p(n)$ up to polylog(n), we will use it with $p(n)$ at most $O((\log\log n)^3)$. In this case the reduction from 3SAT to the above CSP instance will run in $n^{O(\log\log\log n)}$ time.

3.1.1 Some details about the proof of Theorem 9

The above result is Theorem 1.5 in [27]. We say a few words about how the parameters claimed above arise in [27], since the complexity of the reduction is not spelled out explicitly there. (The focus in [27] was on the case of p being a constant, and in this case the reduction clearly had polynomial time complexity.) We discuss the high level details of the proof so as to convince the reader of the claimed bounds for non-constant functions $p(n)$.

The starting point for the above result, like so many other PCP results, is the u-parallel version of the basic 2-prover 1-round (2P1R) proof system for Gap3SAT(5). In a Gap3SAT(5) instance of 3SAT each variable occurs exactly in five clauses, and the goal is distinguish satisfiable instances from those that are most η_0-satisfiable for some absolute constant $\eta_0 < 1$[2]). In this u-parallel game, based on a Gap3SAT(5) instance ψ, a verifier picks u clauses of ψ at random and asks one prover (the "clause prover") for the assignment to the variables in those clauses. The verifier also picks u variables, one from each of the u chosen clauses, at random and asks the other prover (the "variable prover") for their values. The verifier accepts if the clause prover returns an assignment satisfying all the u clauses, and the variable prover's response is consistent with the clause prover's answer on the u common variables they both got asked about. By Raz's parallel repetition theorem [35], the parallelized 2P1R has soundness c_0^u for some absolute constant $c_0 < 1$. The size of the 2P1R instance (viewed naturally as a bipartite graph) is $n^{O(u)}$, if n is the size of the Gap3SAT instance ψ, and the answers from each of the provers consist of at most $3u$ bits.

To get the hardness for p-ary CSP, one gives a *PCP verifier* V which randomly checks ℓ locations of a proof consisting of suitably encoded versions of the two provers' answers in the above u-parallel game. Each location of the proof must consist of a value in $\{1, 2, \ldots, p\}$. Specifically, the verifier expects as proofs the encodings of the two provers' answers by the p-ary *long code*. Each of these long codes has a position for every function taking values in $\{1, 2, \ldots, p\}$ on a domain of size $2^{O(u)}$, and thus has size $p^{2^{O(u)}}$. The variables of the CSP naturally correspond to positions of these long codes (and a proof corresponds to a p-ary assignment to the variables). The total number of variables in the CSP instance is at most

$$n^{O(u)} p^{2^{O(u)}} , \tag{1}$$

[2]There exists such η_0 for which the problem is NP-hard, according to the PCP theorem.

since there are $n^{O(u)}$ long codes each of length $p^{2^{O(u)}}$.

Håtsad and Khot first give a verifier that makes 5 queries and has completeness 1 and soundness at most $\frac{1}{p} + c_0^{\Omega(u)} p^{O(1)}$ (this is the result of Lemma 4.9 in [27]). Then for any integer $f \geqslant 1$, they analyze an "f-iterated" verifier that performs f^2 copies of this test, reusing many queries between the tests for a total of $\ell = 4f + f^2$ queries into the proof. This verifier inherits the perfect completeness of the basic 5 query verifier, and Theorem 4.10 of [27] shows that the iterated verifier has soundness $p^{-f^2} + c_0^{\Omega(u)} p^{O(1)}$ (our f is called k in their notation). (In other words, the f^2 tests effectively behave as if they are independent even though they share many queries.) To summarize, if the Gap3SAT instance ψ is satisfiable, all constraints checked by the verifier can be simultaneously satisfied, whereas in the soundness case when ψ is at most η_0-satisfiable, at most a fraction $p^{-f^2} + c_0^{\Omega(u)} p^{O(1)}$ of the constraints can be satisfied by any assignment.

By picking $u = d_0 \ell \log p$ for a large enough (absolute) constant $d_0 > 0$, the soundness can be made at most $2p^{-f^2} \leqslant p^{-\ell + 10\sqrt{\ell}}$. By a simple inspection of their construction (Section 4.2.2 in [27]), the total number of different query patterns of the f-iterated verifier (which correspond to constraints in the CSP view) can be seen to be at most

$$n^{O(u)} 2^{O(uf)} p^{f 2^{O(u)}} \ . \tag{2}$$

For the choice $u = \Theta(\ell \log p)$, (1) and (2) imply that the size of the CSP instance, and the time complexity of the reduction, are both $n^{O(\ell \log p)} p^{p^{O(\ell)}}$ as claimed.

3.2 A new CSP by serial product

We will boost the soundness of the above CSP by forming a new instance via a λ-fold serial product for an integer parameter $\lambda = \lambda(n)$. Specifically, we will form a new p-ary k-CSP instance for $k = \lambda\ell$, where there is a constraint for each λ-tuple of constraints in the original CSP (with repetitions), and this constraint is satisfied iff all the λ constituent constraints are satisfied. Note that the number of constraints of the new CSP is bounded by $n^{O(\ell\lambda \log p)} 2^{p^{O(\ell)} \lambda \log p}$. An assignment that satisfies all constraints of the p-ary ℓ-CSP instance satisfies all constraints of the p-ary k-CSP instance, so perfect completeness is preserved. Also, an assignment σ to the variables that satisfies a fraction ρ of the constraints of the original CSP satisfies a fraction ρ^λ of the constraints in the product CSP (since each of the λ chosen constraints must be satisfied by σ, and these choices are made independently). The soundness therefore gets raised to the λ'th power, and is at most $p^{-(\ell - 10\sqrt{\ell})\lambda}$.

3.3 The Reduction to EDPwC instance

The overall reduction first constructs the p-ary ℓ-CSP instance guaranteed by Theorem 9, followed by taking a λ-fold serial product.

Suppose an arbitrary constant $\varepsilon > 0$ is given and we seek hardness for congestion $c(n)$ where the function $c(n)$ satisfies $1 \leqslant c(n) \leqslant \frac{\log \log n}{10 \log \log \log n}$. We first specify the parameters $\ell, p = p(n)$ and $\lambda = \lambda(n)$ to construct the p-ary $k = \lambda\ell$-CSP instance as above. For convenience, we also define

two other parameters h, b that are closely related to the congestion $c = c(n)$.

$$\begin{aligned}
\ell &= \ell(\varepsilon) \text{ is a large enough constant, } \ell(\varepsilon) > 10000/\varepsilon^2 && (3)\\
h &= c = c(n) && (4)\\
b &= c + 1 && (5)\\
p &= \text{Any prime such that } b^2 < p < b^3 \text{ (note that such a prime always exists)} && (6)\\
\lambda &= \left\lceil \frac{\log\log n}{(b+1)\log p} \right\rceil && (7)\\
k &= \lambda\ell \ . && (8)
\end{aligned}$$

We note that since $c(n) \leqslant \frac{\log\log n}{10\log\log\log n}$, $\frac{\log\log n}{(b+1)\log p} > 1$ so that $\lambda = \left\lceil \frac{\log\log n}{(b+1)\log p} \right\rceil \leqslant 2\frac{\log\log n}{(b+1)\log p}$.

We now describe the reduction from the resulting p-ary k-CSP to routing in undirected graphs. Let N be the number of variables and M the number of constraints. Recall that we have

$$M \leqslant n^{O(\ell\lambda\log p)} 2^{p^{O(\ell)}\lambda\log p} \leqslant n^{O(\log\log n)} \ . \tag{9}$$

We define some more notation concerning the p-ary k-CSP instance.

- Let J be an upper bound on number of satisfying assignments to any of the constraints (we have $J \leqslant p^{10\sqrt{\ell}\lambda} = p^{10k/\sqrt{\ell}}$).
- Let B_i denote the number of constraints in which variable x_i participates in. Note that $\sum_i B_i = kM$. Let $T = \max_i B_i$ be the maximum number of constraints any variable participates in.

The reduction will use two other integer parameters Y, Z which will be defined in Section 3.4.

For a positive integer K, we will use the notation $[K]$ to denote the set $\{1, 2, \ldots, K\}$. In what follows, $i \in [N]$ will typically be used for a variable index, $j \in [M]$ for a constraint index, and $q \in [p]$ to refer to a possible value assigned to a variable. For each $q \in [p]$ and for constraint C_j containing x_i, let Γ_{ijq} be the set of satisfying assignments to C_j that set x_i to q and let Γ_{iq} be the set of all pairs (C_j, γ) such that C_j contains x_i and $\gamma \in \Gamma_{ijq}$.

The main building blocks in the construction are certain variable gadgets. For each variable x_i, we give a randomized construction of a gadget called G_i that we describe in detail below. Recall that $B_i \leqslant T$ denotes the number of occurrences of x_i in the constraints. For notational simplicity, we omit the subscript i in B_i and refer to it as simply B when i is obvious from the context(we do **not** assume that all B_i's are equal). For each $z \in [Z]$, the variable gadget G_i has a matching $M_z^{(i)}$ consisting of YJB *special* edges $e^i_{z,s} = (u^i_{z,s}, v^i_{z,s})$ for $s \in [YJB]$ — the vertex $u^i_{z,s}$ (resp. $v^i_{z,s}$) will be referred to as the left (resp. right) endpoint of the edge $e^i_{z,s}$.

These disjoint matchings will be strung together by $Z + 1$ intermediate levels of *connector vertices* in a random way as described below. Let $\tau = (C_j, \gamma, y)$ be a triple where γ is a satisfying assignment for C_j and $y \in [Y]$; we call such a triple an *accepting interaction*. For each $z \in [Z + 1]$ and $q \in [p]$, we have a set $W^{(i)}_{q,z}$ of $Y|\Gamma_{iq}|$ connector vertices, each labeled by an accepting interaction (C_j, γ, y) where the pair (C_j, γ) belongs to Γ_{iq} and $y \in [Y]$ (that is, there are Y vertices for each pair (C_j, γ) where C_j contains x_i and the satisfying assignment γ to C_j assigns the value q to x_i).

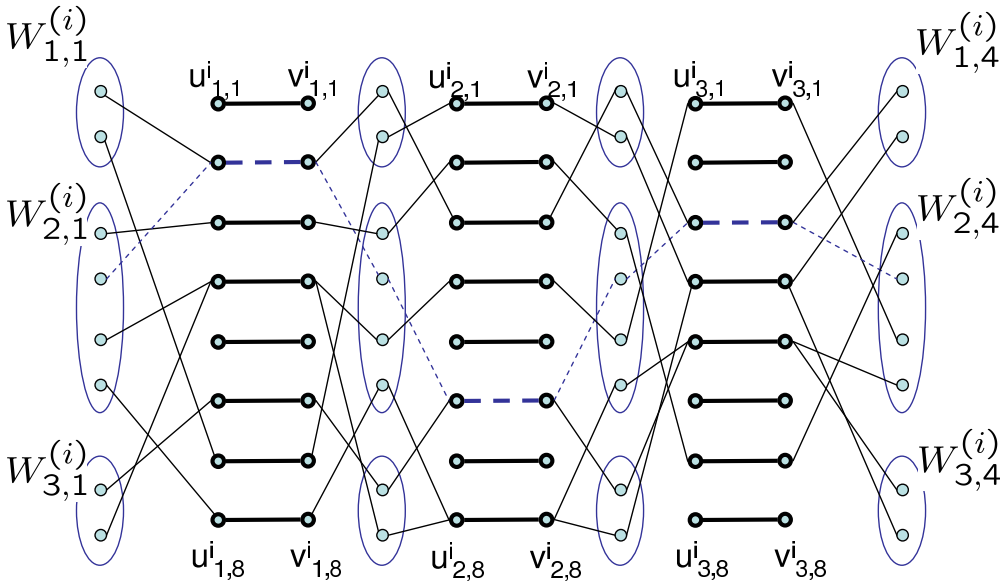

Figure 1: Gadget G_i for $p = 3$, $Z = 3$ and $Y = 2$. The special edges are shown in bold. One of the canonical paths is shown dotted.

We will denote by $w^{i,z}_{C_j,\gamma,y}$ the vertex in $W^{(i)}_{q,z}$ that is labeled by the accepting interaction (C_j, γ, y). Note that, for each z, we have $\sum_{q\in[p]} |W^{(i)}_{q,z}| \leqslant YJB$. Now comes the crucial interconnection of the different matchings via the connector vertices. For each $q \in [p]$ and $z \in [Z]$, pick **independently and uniformly at random** a subset $S^{(i)}_{q,z}$ of the matching $M^{(i)}_z$ of size $|W^{(i)}_{q,z}|$. Connect the left endpoints of the edges in $S^{(i)}_{q,z}$ to the vertices $W^{(i)}_{q,z}$ via a **random matching**. If the left endpoint of an edge in $S^{(i)}_{q,z}$ is connected to the vertex labeled $w^{i,z}_{C_j,\gamma,y}$, then the right endpoint of that edge is connected to the corresponding node $w^{i,z+1}_{C_j,\gamma,y}$ in the $(z+1)$'th level. Moreover, we will call this special edge as $f^{i,z}_{C_j,\gamma,y}$. Note that the collection of the edges $f^{i,z}_{C_j,\gamma,y}$ as (C_j, γ) ranges over Γ_{iq} and y ranges over Y is precisely the submatching $S^{(i)}_{q,z}$ of $M^{(i)}_{q,z}$.

We now identify and give names to certain collections of "canonical" paths through the variable gadget G_i that correspond to the p possible value assignments to x_i. Fix a value q for x_i. For each pair $(C_j, \gamma) \in \Gamma_{iq}$ and for each $y \in [Y]$, we can define a canonical path $P_i[j, \gamma, y]$ as the unique path going through the connecting vertices labeled $w^{i,z}_{C_j,\gamma,y} : z \in [Z+1]$. Note that for each $q \in [p]$ and $i \in [N]$, the at most YJB_i canonical paths $P_i[j, \gamma, y] : (C_j, \gamma) \in \Gamma_{iq}, y \in [Y]$ are edge disjoint. Moreover, at each level $z \in [Z]$, $P_i[j, \gamma, y]$ passes through a special edge $e^i_{z,s}$ where s is distributed uniformly in $[YJB]$.

This defines a variable gadget. The graph is constructed by linking together the variable gadgets as follows. For each constraint C_j there are Y source-sink pairs $(S_{j,y}, T_{j,y})$. Let $x_{i_1}, \ldots, x_{i_k}$ be the k variables in constraint C_j in some order. For each such assignment γ that satisfies C_j, we string together from $S_{j,y}$ to $T_{j,y}$ by connecting:

1. $S_{j,y}$ to the vertex $w^{i_1,1}_{C_j,\gamma,y}$ by a weighted edge of weight Z, "implemented" as a simple path of

length Z (with the internal vertices of this path being degree two vertices that are exclusive to this path)[3],

2. $w^{i_t,Z+1}_{C_j,\gamma,y}$ to $w^{i_{t+1},1}_{C_j,\gamma,y}$ for $t \in [k-1]$ by an edge, and

3. $w^{i_k,Z+1}_{C_j,\gamma,y}$ to $T_{j,y}$ by a weight Z edge, again implemented by a simple path of length Z (again, the internal vertices of this path have degree two).

This naturally defines the canonical paths $P[j,\gamma,y]$ connecting $S_{j,y}$ to $T_{j,y}$ that pass through $P_{i_t}[j,\gamma,y]$ for each $t \in [k]$. Note that for each $(S_{j,y}, T_{j,y})$ pair, we have a choice among the canonical paths $P[j,\gamma,y]$ corresponding to which assignment γ satisfies C_j. Each of these canonical paths has $O(kZ)$ edges.

Note that the number of source-sink pairs is MY, each of which has at most J canonical paths of length $O(kZ)$ connecting them. Therefore, the total number of edges in the graph is $O(MYJkZ)$, and the graph can be constructed in time polynomial in the number of edges. This completes the construction.

3.4 The parameters

For easy reference, recall again the following parameters concerning the instance of p-ary k-CSP instance we start with:

- $p = p(n)$ is a prime in the range (b^2, b^3) where $b = c(n)+1$, and $1 \leqslant c(n) \leqslant \frac{\log\log n}{10\log\log\log n}$ is the congestion for which we seek hardness.
- $k = \lambda\ell$ where $\lambda = \left\lceil \frac{\log\log n}{(b+1)\log p} \right\rceil$ and $\ell = \ell(\varepsilon)$ is a large enough constant, $\ell(\varepsilon) > 10000/\varepsilon^2$.
- N, the number of variables
- M, the number of constraints, $M = n^{O(\log\log n)}$
- J, the maximum number of satisfying assignments to any constraint ($J \leqslant p^{10k/\sqrt{\ell}}$)
- B_i, the number of constraints in which variable x_i participates. We have $\sum_{i=1}^{N} B_i = kM$.

We now discuss the values for the other parameters Y, Z that were used in the above reduction, and define some other parameters that will be useful in the analysis. Note the parameters are

[3]The use of a long path instead of an edge for this connection will be very convenient in a later step in our analysis.

defined in order of dependence.

$$
\begin{aligned}
r &= p^{k/3} \geqslant (\log n)^{\frac{\ell}{3(b+1)}} && (10)\\
\rho &= pkJr && (11)\\
Z &= 128b\rho^b && (12)\\
g &= Z\rho && (13)\\
Y &= 4M^g g^{2g+1} && (14)\\
X_i &= YJB_i \text{ (for } i \in [N]) && (15)\\
X &= \max_i X_i && (16)\\
A_i &= 2X_i/\rho\ . && (17)
\end{aligned}
$$

We give some intuition for the parameters here. The final gap we get is r and ρ is roughly (not much larger than) r. The best gap we can hope for is the soundness of the p-ary k-CSP, which is $p^{-(\ell-10\sqrt{\ell})\lambda}$ and surely at most $p^{-k/3}$. This determines our choice of the gap r.

Since there is a source-sink path for every assignment to the corresponding constraint, two such paths form a cycle. Since each cycle allows a demand using an edge on the cycle to avoid a particular heavily loaded edge, we need the notion of almost-canonical paths. The capacity argument we use for flows that use several detours requires us to set the threshold for almost-canonical to be ρ detours.

Z has to be roughly ρ^{b-1} for the measure concentration part of the balls-and-bins argument, that guarantees that for every setting of flows, the number of bins with too many balls is close to its expectation. However, since we allow almost-canonical paths to deviate on about ρ edges, the expected number of bad bins has to be large enough, which requires us to set Z to be about ρ^b.

Once again, to make a capacity argument, we need to set the threshold g defining a long cycle to be about $Z\rho$. The number of short cycles then is exponential in g, which requires us make Y exponential in g as well. Thus Y is roughly $2^{\rho^{b+1}}$ and the hardness we get is about $\log^{\frac{1}{b+1}} Y$. The total number of vertices, say V, in the final graph H is $O(MYJkZ)$ which is at most Y^2, so we get a gap of about $\Omega(\log^{\frac{1}{c+2}} V)$ as a function of the number V of vertices. For future reference, we formally record the following bound on r.

Lemma 10. *With the above choice of parameters, we have $r = \Omega(\log^{\frac{1-\varepsilon}{c+2}} V)$, where V is the number of vertices in the graph H output by the reduction.*

Proof. Let us bound $\log Y$ from above in terms of r. We have $\log Y = O(g(\log M + \log g)$. By definition, $g = b^{\Theta(k)}$. Since $b \leqslant \log\log n$ and $k = O(\log\log n)$, $g \leqslant \log^{O(1)} n$. Combined with the fact that $M = n^{O(\log\log n)}$, we have $\log Y = O(g\log^2 n)$. Again by definition, $g = 128b\rho^{b+1} = O(\rho^{b+1}\log n)$ so that $\log Y = O(\rho^{b+1}\log^3 n)$. We have

$$\rho = pkJr \leqslant p^{20k/\sqrt{\ell}} r = r^{1+60/\sqrt{\ell}} \leqslant r^{1+\varepsilon/2}\ ,$$

if $\ell \geqslant 10000/\varepsilon^2$. Also $\log^3 n \leqslant r^{(b+1)9/\ell} \leqslant r^{(b+1)\varepsilon/2}$. Plugging these, we get $\log Y = O(r^{(b+1)(1+\varepsilon)})$, which gives $r = \Omega((\log Y)^{\frac{1-\varepsilon}{b+1}})$. Recalling that $V \leqslant Y^2$ and $b = c+1$, we have the claim. □

3.5 Time complexity

The time complexity of the reduction is polynomial in Y, which for the above choice of parameters is easily seen to be $2^{\log^{O(\ell)} n}$ if n is the size of the original SAT instance. Therefore, we have a quasi-polynomial time reduction.

4 Hardness Gap Analysis

We now analyze the hardness gap achieved by the preceding reduction.

4.1 Completeness

When there is a satisfying assignment to the p-ary k-CSP instance, all MY pairs $(S_{j,y}, T_{j,y})$ can be routed on edge-disjoint paths. Indeed, let a be an assignment that satisfies all constraints, and let γ be the projection of a to constraint C_j. Then, for each $y \in [Y]$, connect $(S_{j,y}, T_{j,y})$ via the canonical path $P[j, \gamma, y]$.

4.2 Soundness

We now show that if no assignment satisfies more than a small fraction of the constraints, then it is impossible to route many of the $(S_{j,y}, T_{j,y})$ paths, even if congestion c is allowed. This part is complicated with several steps and using several of the ideas developed by Andrews and Zhang in their paper [4] on hardness of congestion minimization.

Due to the fact that we are aiming for perfect completeness, we have to handle an additional complication. We call a source-sink pair $(S_{j,y}, T_{j,y})$ *risky* if for some assignments $\gamma \neq \gamma'$ satisfying C_j, the canonical paths $P[j, \gamma, y]$ and $P[j, \gamma', y]$ intersect. We now argue that the number of risky paths is small with high probability.

We call a canonical path $P_i[j, \gamma, y]$ *risky* if for some $\gamma' \neq \gamma$, $P_i[j, \gamma, y]$ intersects $P_i[j, \gamma', y]$. Note that γ' and γ must assign different values to x_i for them to intersect. In this case their matching edges $e^i_{z,s}$ are chosen independently at each level, and hence the probability that $P_i[j, \gamma, y]$ intersects $P_i[j, \gamma', y]$ is at most $\frac{Z}{YJB_i}$. Taking a union bound over the at most J possible γ''s, the probability that $P_i[j, \gamma, y]$ is *risky* is at most $\frac{Z}{YB_i} < \frac{Z}{Y}$. A demand $(S_{j,y}, T_{j,y})$ is *risky* if one of its canonical paths is risky. Taking a union bound over the k variables and the at most J assignments γ, we get an overall bound of $\frac{kJZ}{Y}$. Thus the expected number of *risky* demand pairs is $MkJZ$. Using Markov's inequality, with probability 99/100, the number of *risky* demand pairs is at most $100MkJZ$.

It is easy to verify that for the above setting of parameters, this quantity is $M\text{polylog}(Y)$, which is negligible compared to MY, the total number of demands. Moreover, it is easy to find all *risky* demand pairs in an instance, and delete them. The resulting graph still has $MY(1 - o(1))$ source-sink pairs, inherits the perfect completeness, and has no *risky* demands. For the rest of the paper, we assume that we have an instance with no *risky* demands.

We take an arbitrary routing of some subset of the $S_{j,y}$–$T_{j,y}$ pairs and divide the paths used for the routing into two classes: almost-canonical and non-canonical, defined as follows.

Definition 11. For $j \in [M]$ and $y \in [Y]$, an $S_{j,y}$–$T_{j,y}$ path Π is said to be *almost-canonical* if there is a satisfying assignment[4] $\gamma \in [J]$ to C_j such that for each variable x_i present in constraint C_j, the path Π uses more than $Z - \rho$ out of the Z matching edges in M_z^i, $z \in [Z]$, that are used by $P[j, \gamma, y]$. In other words, Π deviates from the canonical path $P[j, \gamma, y]$ in at most ρ special edges in each variable gadget.

For such an almost-canonical path Π, we say that the value *highlighted* by Π for x_i equals the value that satisfying assignment γ assigns to x_i (note that each almost-canonical path highlights exactly one value for each variable present in the associated constraint).

Definition 12. For $j \in [M]$ and $y \in [Y]$, an $S_{j,y}$–$T_{j,y}$ path is said to be *non-canonical* if it is not almost-canonical.

We will bound the number of almost-canonical and non-canonical paths separately.

4.3 Bounding almost-canonical paths

We divide almost-canonical paths into two categories: heavy and light, defined below, and bound each of these in turn.

Definition 13 (Heavy and Light Paths)**.** For $i \in [N]$ and $q \in [p]$, a variable-value pair (x_i, q) is said to be *heavy* if more than A_i almost-canonical paths highlight the value q for x_i.

An almost-canonical path Π connecting $S_{j,y}$ to $T_{j,y}$ is said to be *heavy* if (x_i, q_i) is heavy for each variable x_i in C_j where q_i is the value highlighted by Π for x_i.

An almost-canonical path that is not heavy is said to be *light*.

4.3.1 Light almost-canonical paths

It is very easy to bound the number of light almost-canonical paths. Indeed this number is at most $p \sum_i A_i$, since for a pair (x_i, q) at most A_i almost-canonical paths can be light because of it. We record this as:

Lemma 14. *In any routing of a subset of the $S_{j,y}$–$T_{j,y}$ pairs, there can be at most $p \sum_{i=1}^{N} A_i$ light almost-canonical paths.*

4.3.2 Heavy almost-canonical paths

Definition 15 (Over-ambiguous variables)**.** Define a variable x_i to be *over-ambiguous* if the variable-value pair (x_i, q) is heavy for more than h values of $q \in [p]$.

The above notion is an important one for our analysis. On the one hand, if there are no over-ambiguous variables, then the soundness of the original CSP implies a bound on number of heavy almost-canonical paths (Lemma 16 below). On the other hand, the existence of an over-ambiguous variable x_i implies that with high probability the almost-canonical paths that pass through the

[4]There can be more than one such γ. If so, we pick one of them arbitrarily.

variable gadget G_i create congestion more than $h = c$ (provided parameters such as Z are picked appropriately). The intuition behind the latter phenomenon can be explained as follows. For each value q for which (x_i, q) is heavy, the heavy almost-canonical paths pass through a random subset of the matching $M_z^{(i)}$ of size A_i. For different values q, these subsets are chosen independently for each z. If more than h such subsets of size A_i (out of X_i) are chosen, then with high probability they will all collide on some element creating congestion $h+1$, provided Z is large enough. This is the "balls and bins" intuition behind our reduction.

Lemma 16. *If there are no over-ambiguous variables in a routing of a subset of $S_{j,y}$–$T_{j,y}$ pairs, then the number of heavy almost-canonical paths is at most $p^{10k/\sqrt{\ell}}(h/p)^k MY$.*

Proof. Let $\tilde{C}$ denote the set of constraints C_j for which for some $y \in [Y]$ there is a heavy almost-canonical flow path connecting $S_{j,y}$ to $T_{j,y}$. Consider the following assignment to each x_i. Pick an element from the set $V_i = \{q \mid (x_i, q) \text{ is heavy}\}$ uniformly at random. Note that for each constraint in $\tilde{C}$, this assignment satisfies it with probability at least $1/h^k$, since $|V_i| \leqslant h$ for all x_i. Therefore, the expected number of constraints satisfied by this assignment is at least $|\tilde{C}|/h^k$. This quantity must be at most $p^{\lambda(-\ell+10\sqrt{\ell})}M = p^{-k+10k/\sqrt{\ell}}M$ due to the soundness of p-ary k-CSP. It follows that the total number of heavy almost-canonical paths is at most $|\tilde{C}|Y \leqslant p^{10k/\sqrt{\ell}}(h/p)^k MY$. □

We now show that if there is an over-ambiguous variable, then we must get congestion at least $h+1$, with high probability over the construction of our instance.

Consider a variable x_i and let $\alpha_1, \ldots, \alpha_b$ be $b = h+1$ distinct possible values of x_i. Denote by $X_i = YJB_i$ the number of matching edges in each of the matchings M_z^i (recall that $B_i \leqslant T$ is the number of occurrences of variable x_i in the constraints C_j, $j \in [M]$). For convenience, we shall omit the subscript i in the rest of this section and use A and X to refer to A_i and X_i respectively. For $s \in [b]$, let I_s be a set of A triples (j, γ, y) such that C_j uses x_i and γ assigns the value α_s to x_i. Thus by construction, the set of matching edges used by (the canonical paths corresponding to) triples in I_s is a uniformly random subset of size A. We expect about $(A/X)^b X$ of the matching edges to be used by a triple from each of the I_s's.

Definition 17. We say that $I_1, \ldots, I_b$ are *highly congesting* at level $z \in [Z]$ if at least $(\frac{A}{2X})^b X$ matching edges in M_z^i are used by a triple from each $I_1, \ldots, I_b$ (in other words, they have congestion b).

We say bad event $B(\alpha_1, \ldots, \alpha_b, I_1, \ldots, I_b, z)$ occurs if the sets $I_1, \ldots, I_b$ are *not* highly congesting at level z.

Lemma 18. $\mathbf{Pr}[B(\alpha_1, \ldots, \alpha_b, I_1, \ldots, I_b, z)] \leqslant 2e^{-\frac{A^b}{16(2X)^{b-1}}}$.

Proof. The event that the sets $I_1, \ldots, I_b$ are *not* highly congesting at level z has the same probability as the following event: For each color $1, \ldots, b$, we are throwing A balls into X distinct bins and we want to compute the probability that fewer than $(\frac{A}{2X})^b X$ bins have a ball of every color. Let $\beta_s = (\frac{A}{2X})^s$.

Suppose that we have thrown balls for s of these colors and assume inductively that the probability that there are at least $\beta_s X$ bins with q balls each is at least $1 - \sum_{t=1}^{s} e^{-\frac{\beta_t X}{8}}$. Suppose that $\beta_s X$ bins indeed have s balls each. Let Y_j be the number of $(s+1)$-loaded bins after we have

thrown the jth ball of color $(s+1)$ and let $Y'_j = Y_{j+1} - Y_j$. As long as $Y_j \leqslant 2\beta_{s+1}X$, Y'_j is 1 with probability at least $(\beta_s - 2\beta_{s+1})$. Thus the probability that $Y_A < \beta_{s+1}X$ is bounded above by the probability that the sum of A independent 0-1 random variable each with mean $(\beta_s - 2\beta_{s+1})$ is less than $\beta_{s+1}X$. Assuming $A < \frac{X}{10}$ and using Chernoff bounds, the latter probability is at most $e^{-\frac{\beta_{s+1}X}{8}}$. Hence the induction holds. The claim follows. □

Let $B(\alpha_1, \ldots, \alpha_b, I_1, \ldots, I_b)$ be the event that the above bad event happens at no less than $\frac{Z}{2}$ of the Z levels. Then

$$\begin{aligned}\mathbf{Pr}[B(\alpha_1, \ldots, \alpha_b, I_1 \ldots, I_b)] &\leqslant \binom{Z}{\frac{Z}{2}} e^{-\frac{ZA^b}{32(2X)^{b-1}}} \\ &\leqslant 2^Z e^{-\frac{ZA^b}{32(2X)^{b-1}}}.\end{aligned}$$

Let $B(\alpha_1, \ldots, \alpha_b)$ be the event that there exist sets $I_1, \ldots, I_b$ such that this bad event occurs. Then

$$\begin{aligned}\mathbf{Pr}[B(\alpha_1, \ldots, \alpha_b)] &\leqslant \binom{X}{A}^b 2^Z e^{-\frac{ZA^b}{32(2X)^{b-1}}} \\ &\leqslant (\frac{eX}{A})^{bA} 2^Z e^{-\frac{ZA^b}{32(2X)^{b-1}}} \\ &\leqslant e^{bA(\log X + 1 - \log A) + Z\ln 2 - \frac{ZX}{16}(\frac{A}{2X})^b}\end{aligned}$$

Plugging in the values of A in terms of X, the negative of the exponent is at least $X(\frac{Z}{16\rho^b} - \frac{2b(\log\rho+1)}{\rho} - \frac{Z\ln 2}{X})$. Since $Z \geqslant 128b\rho^b \geqslant 64b\rho^{b-1}(\log\rho+1)$, the first term is at least twice the second. The third term is easily seen to be much smaller. The negative of the exponent is thus $\omega(X)$.

Thus the probability that for a given variable x_i there is a set of values $\alpha_1, \ldots, \alpha_b$ for which $B(\alpha_1, \ldots, \alpha_b)$ occurs is at most p^b as much, which is still $o(\exp(-X))$. Taking a union bound over the n variables, the probability of any bad event is $o(1)$.

Suppose that no bad events occur, but that a variable x_i is over-ambiguous with $b = h+1$ heavy values $\alpha_1, \ldots, \alpha_b$. Let P_s be a set of A almost-canonical paths that highlight value α_s for x_i. Let I_s be the set of triples corresponding to the almost canonical paths in P_s; by definition, each path in P_s deviates from the corresponding canonical path in I_s in at most ρ levels. On the other hand, the event $B(\alpha_1, \ldots, \alpha_b, I_1, \ldots, I_b)$ did not occur and hence at least $\frac{Z}{2} \geqslant 16b\rho^b$ levels are highly congested by the canonical paths in $I_1, \ldots, I_b$. At each such level, at least $(\frac{A}{2X})^b X$ of the paths in $\cup_{s=1}^b P_s$ must deviate from the corresponding canonical path, or else some edge at this level has congestion b. The total number of deviations is thus $(\frac{A}{2X})^b X(16b\rho^b)$ which is at least $8bA\rho$. Since we are looking at only bA paths, one of them must deviate at 8ρ levels, which is a contradiction since almost-canonical paths deviate from the associated canonical path at most ρ times. We have thus proved:

Lemma 19. *With high probability (over the choice of random matchings in the construction), if a routing of some subset of $S_{j,y}$-$T_{j,y}$ pairs has an over-ambiguous variable, then it creates congestion at least $h+1$ on some edge.*

In summary, combining Lemmas 14, 16 and 19, we conclude the following:

Lemma 20. *With high probability over the construction of the graph H from the p-ary k-CSP instance, the following holds when the k-CSP instance is at most $p^{-k+10k/\sqrt{\ell}}$-satisfiable: The number of $S_{j,y}$-$T_{j,y}$ pairs which can be routed using almost-canonical paths using congestion c is at most*

$$p^{10k/\sqrt{\ell}}(h/p)^k MY + p\sum_{i=1}^{N} A_i \ .$$

4.4 Bounding non-canonical paths

At a high level we bound the number of non-canonical paths by classifying them into long and short ones. The long ones are few because of volume arguments. The short ones are few for they create short cycles in a certain random graph that has few short cycles in expectation. The details of the overall analysis are technical, mainly due to "guaranteed" cycles in H comprising of two different canonical paths between an $S_{j,y}$-$T_{j,y}$ pair. Specifically, following [4], we argue about cycles in an auxiliary graph, called the *incidence graph*.

We define an auxiliary (undirected) graph G based on the instance, say H, of edge-disjoint paths constructed in Section 3.3. The nodes in G consist of:

- A demand node $d_{j,y}$ for each $S_{j,y}$-$T_{j,y}$ pair
- A path node p for each canonical path $P[j, \gamma, y]$
- A special edge node f for each edge in the matchings $M_z^{(i)}$ for $i \in [N]$ and $z \in [Z]$.

The edges in G are defined as follows:

- An edge $(d_{j,y}, p)$ for each demand node $d_{j,y}$ and path node p corresponding to $P[j, \gamma, y]$ for all satisfying assignments γ to C_j. Note that each path node is connected to a unique demand node, namely the one whose source and destination it connects.
- An edge between a path node p and special edge node f if f belongs to p.

(Thus G is obtained from H by shrinking each special edge, identifying each source sink pair and identifying all connector nodes corresponding to each canonical path.)

A route in the original graph H maps into a route in G in a natural way. Each route of length 2 that connects two special edges f_1 and f_2 (via a connector vertex) along canonical path p maps to a two-edge route $f_1 \to p \to f_2$ in G. The two-edge route between a source node $S_{j,y}$ to a special edge f (through some connector vertex) along canonical path p corresponds to a two-edge route $d_{j,y} \to p \to f$ in G. Likewise, the route between special edge f and node $T_{j,y}$ via a canonical path p corresponds to a two-edge route $f \to p \to d_{j,y}$ in G.

The following easy lemma says that long routes in G correspond to long routes in the original graph H.

Lemma 21. *An $S_{j,y}$-$T_{j,y}$ route in H that passes through x special edges corresponds to a route of length at most $2x + 2$ in G.*

We are now in a position to tackle non-canonical paths used for routing in H. Consider a demand $d = d_{j,y}$ and let π be the (simple) path that d is routed along in H. This path maps naturally to a (non-simple) path in G and thus induces a subgraph of G. There are two scenarios to consider, depending on whether or not this subgraph is cyclic — this is the precise dichotomy of non-canonical paths used to bound them.

4.4.1 π forms a cycle in G

This case is further divided into two subcases, depending on whether the cycle is of length at most g or more than g.

Lemma 22. *With probability at least $3/4$, in the graph G the number of demands which are within distance g of a cycle of length less than g is at most $4M^g g^{2g+1}$.*

Proof. For the sake of simplicity, let us shrink all *d-p* edges in G thus getting a bipartite (multi)graph G'. Since the demands d in G are not *risky* , every cycle in G uses path/demand nodes from at least two demands. Thus each cycle in G is still a cycle (of no larger length) in G'. Each d node is now connected to at most J of the X_i f-nodes for each variable i participating in its constraint, at each level. Let us estimate the expected number of cycles in G' containing $g' < g$ nodes. The number of prospective cycles is at most $(2ZYJ\sum B_i)^{g'} = (2ZYJkM)^{g'}$, since G' has no more than $2Z\sum X_i$ nodes. Conditioned on at most $g' \leqslant Y/2$ other edges, each edge of this cycle occurs with probability no more than $\max_i J/(X_i - g') \leqslant 2J/X_i \leqslant 2/Y$. Thus the expected number of cycles of length g' is at most $(2ZYJkM)^{g'}(2/Y)^{g'} \leqslant (4kZJM)^{g'}$.

Therefore, the expected number of cycles of length less than g is at most $(4kZJM)^g$, and so with probability at least $3/4$ the number of cycles of length less than g is at most $4(4kZJM)^g$. The maximum degree of the graph is at most $\max\{kJZ, p\} = kJZ$, so at most $g(kJZ)^g$ nodes are within distance g of any particular cycle of length less than g. This leads to a total of at most $4g(4k^2J^2Z^2M)^g$ demands which are within distance g of some cycle of length less than g. Now $2kJ \leqslant pkJ \leqslant \rho$ and $g = Z\rho$, so we have $4g(4k^2J^2Z^2M)^g \leqslant 4g(g^2M)^g$, which gives the desired upper bound. □

In view of the above lemma, we conclude the following. Except for possibly $4M^g g^{2g+1}$ demands, all demands routed that fall under this case are connected by paths of length at least $g - kZ \geqslant g/2$. Since the total capacity of all special edges in a congestion c routing is at most c times the number of special edges (which is $O(MYJkZ)$), there can be at most $O(MYJkZc/g) = O(MYJkc/\rho)$ routed demands that fall in this category. We remark that demands of this form force us to take g to be about $Z\rho$.

4.4.2 π does not form a cycle in G

In this case, we have the following crucial lemma.

Lemma 23. *If a demand is routed along a path π (in H) which does not form a cycle in G, then either π is almost-canonical or π has length at least $Z\rho$.*

Proof. The proof of this lemma relies on the following claim:

Claim 24. *Suppose that the path π in H maps to an acyclic subgraph in G. Let $\hat{\pi}$ be the canonical path to which the first edge of π belongs. Then*

- *π passes through each connector node in $\hat{\pi}$ (in H).*
- *Each component of $\pi \setminus \hat{\pi}$ (in H) has length at least Z.*

Assuming the claim, we now prove the lemma. Let χ be the number of components in $\pi \setminus \hat{\pi}$. Since π is simple and contains all the connector nodes of $\hat{\pi}$ it deviates from π on at most χ special edges. If $\chi \leqslant \rho$, the path π must be almost-canonical. On the other hand, since each component of $\pi \setminus \hat{\pi}$ has length at least Z, if $\chi \geqslant \rho$, the total length of π is at least $Z\rho$. □

Proof of claim: We now prove claim 24. Consider the image of π in G: since it induces an acyclic subgraph and s and t are both mapped to the same node d, it is a closed walk along a tree. Thus

Fact 25. *If the image of π leaves a node u in G along the edge (u, v), it returns to u along the edge (v, u).*

Let the nodes of $\hat{\pi}$ in H be $s, w_1, u_1, v_1, \ldots, w_i, u_i, v_i, w_{i+1}, \ldots w_L, t$, where the w_i's are connector nodes and $f_i = (u_i, v_i)$ are special edges on it. Suppose that π does not traverse (v_i, w_{i+1}) for some i and let w_{i+1} be the first such node. Thus π enters w_i using (v_{i-1}, w_i) and hence must leave using (w_i, u_i). In G this corresponds to traversing the edge $(\hat{\pi}, f_i)$. Thus it must traverse this edge in the opposite direction and since π is a simple path in H, the reverse traversal of the edge $(f_i, \hat{\pi})$ must correspond to the edge (v_i, w_{i+1}) in H, contradicting the assumption. This proves the first part of the claim.

Consider now a component of $\pi \setminus \hat{\pi}$. Its image must leave $\hat{\pi}$ along the edge (f_{i^*}, p') for some $p' \neq \hat{\pi}$. Let p' be the path $s', w'_1, u'_1, v'_1, \ldots, w'_i, u'_i, v'_i, w'_{i+1}, \ldots w'_L, t'$ and let $f'_i = (u_i, v_i)$. Since the graph is leveled and edge has the same index in any path it belongs to. Hence, $f_{i^*} = f'_{i^*}$. The edge (f'_{i^*}, p') can correspond to either edge (u'_{i^*}, w'_{i^*}) or to the edge (v'_{i^*}, w'_{i^*+1}) in H. The two cases are analogous and we assume the former. Since π is a simple path, the edge (u'_{i^*}, w'_{i^*}) must be immediately followed by the edge (w'_{i^*}, v'_{i^*-1}) which maps to the edge (p', f'_{i^*-1}) in G. Thus we must revisit p' using the edge (f'_{i^*-1}, p'). Since π is a simple path in H, this must correspond to using the edge (u'_{i^*-1}, w'_{i^*-1}) and must therefore be followed by the edge (w'_{i^*-1}, v'_{i^*-2}) which maps to the edge (p', f'_{i^*-2}) in H. Inducting in this fashion, π must eventually use the edge (p', d'). Using fact 25 again, π must use the edge (d', p'). By simplicity of π in H, these edges must correspond to edges (w'_1, s') and (t', w'_L) in H respectively. Hence the component of $\pi \setminus \hat{\pi}$ must go from s' to t'. Since these nodes are distance Z apart in H, this implies that the component of $\pi \setminus \hat{\pi}$ must have length at least Z as desired.

Since the edge (w'_i, s') has weight Z (i.e., the actual edge is a path of length Z), this means that this component of $\pi \setminus \hat{\pi}$ must have length at least Z, as desired.

□

It follows that there can be at most $O(MYJkZc/Z\rho) = O(MYkJc/\rho)$ non-canonical paths that are routed within congestion c and which fall in this category.

In summary, we record the following bound on the total number of non-canonical paths.

Lemma 26. *With probability at least 3/4, the number of demands that can be routed on non-canonical paths using congestion c is at most*

$$O(MYkJc/\rho) + 4M^g g^{2g+1} \ .$$

4.5 Final Accounting with Chosen Parameters

By combining the bounds on almost-canonical and non-canonical paths from Lemmas 20 and 26, we can bound from above the total number of demands routed with congestion c by

$$p^{10k/\sqrt{\ell}}(h/p)^k MY + p\sum_{i=1}^{N} A_i + O(MYkJc/\rho) + 4M^g g^{2g+1} \ . \tag{18}$$

We now bound each of the terms in (18) above one by one. We shall show that each of the terms is $O(MY/r)$.

1. The first term $p^{10k/\sqrt{\ell}}(h/p)^k MY$. Since $p > h^2$, this term is at most

   $$p^{10k/\sqrt{\ell}-k/2}MY \leqslant p^{-k/3}MY = MY/r \ .$$

   if $\ell \geqslant 3600$.

2. The second term $p\sum_i A_i$. Recall that $\sum_i B_i = kM$ and $A_i = 2X_i/(pkJr) = 2YJB_i/(pkJr)$. So $p\sum_i A_i = 2MY/r$.

3. The third term $O(MYkJc/\rho)$. Since $\rho = pkJr$ and $p \geqslant c$, this term is at most $O(MY/r)$.

4. The fourth term $4M^g g^{2g+1}$. By definition, this equals Y. Now $r \leqslant (\log n)^\ell \ll M$, so this term is at most MY/r as well.

Recalling the lower bound on r from Lemma 10, we get the claimed gap in terms of the number of vertices of the graph. Also, since $V \leqslant 2^{\log^{O(\ell)} n}$ and the result holds for congestion $c(n) \leqslant \frac{\log\log n}{10\log\log\log n}$, in terms of V, we get a result for congestion up to $\frac{\alpha \log\log V}{\log\log\log V}$ where α is a constant that depends on ℓ (and hence on ε).

This proves the gap that we claim in Theorem 1. Note however that our reduction has one-sided error. For the soundness component of the proof we only show that the value of the optimal solution is small with high probability. Hence we have only proved the gap under the assumption that $NP \not\subseteq \text{coRPTIME}(n^{poly(\log n)})$. We now show that this can be converted into an assumption that $NP \not\subseteq \text{ZPTIME}(n^{poly(\log n)})$ which implies the full statement of Theorem 1. This last proof is similar to a standard argument that $NP \subseteq coRP$ implies $NP \subseteq ZPP$.

Lemma 27. *If $NP \subseteq \text{coRPTIME}(n^{poly(\log n)})$ then $NP \subseteq \text{ZPTIME}(n^{poly(\log n)})$.*

Proof. If $NP \subseteq \text{coRPTIME}(n^{\log^a n})$ then $RP \subseteq NP \subseteq \text{coRPTIME}(n^{\log^a n})$. Taking complements we have $coRP \subseteq coNP \subseteq RPTIME(n^{\log^a n})$. Now consider a language $L \in \text{coRPTIME}(f(n))$ for some function $f(n) \geqslant n$. For any string α let $g(\alpha)$ be the string that is obtained by padding α with $f(|\alpha|) - |\alpha|$ zeros. Let $g(L) = \{g(\alpha) : \alpha \in L\}$. Clearly we can obtain a coRP algorithm for $g(L)$ by simply running the coRPTIME($f(n)$) algorithm for L. By the above inclusions this implies that $g(L) \in RPTIME(n^{\log^a n})$. Since we can trivially create $g(\alpha)$ from α in linear time, this in turn implies that $L \in RPTIME(f(n)^{\log^a f(n)})$.

We have just shown that under our initial assumptions $\text{coRPTIME}(f(n)) \subseteq RPTIME(f(n)^{\log^a f(n)})$. If we take $f(n) = n^{\log^a n}$ then this shows that $\text{coRPTIME}(n^{\log^a n}) \subseteq RPTIME(n^{\log a(\log^{a+1} n)^a}) \subseteq RPTIME(n^{\log^{2a^2} n})$. Hence $NP \subseteq \text{coRPTIME}(n^{\log^a n}) \cap RPTIME(n^{\log^{2a^2} n}) \subseteq \text{ZPTIME}(n^{\log^{2a^2} n})$.

□

The claim of Theorem 1 now follows.

5 Non-Perfect Completeness, Node-Disjoint Paths and All-or-Nothing Flow

In this section we extend our results to the cases of non-perfect completeness, Node-Disjoint Paths and All-or-Nothing flow.

5.1 Non-Perfect Completeness

First, we show how to prove Theorem 2, i.e. we show that in the case that perfect completeness is not required, we can improve the inapproximability factor for EDPwC from $(\log V)^{\frac{1-\varepsilon}{c+2}}$ to $(\log V)^{\frac{1-\varepsilon}{c+1}}$. In particular we wish to adapt the reduction such that for some parameter f, in the case of yes-instances there are edge-disjoint paths connecting an f-fraction of the terminal pairs and in the case of no-instances we can connect at most an $f/(\log V)^{\frac{1-\varepsilon}{c+1}}$ fraction of the terminal pairs, even if we allow congestion c.

Proof of Theorem 2. The proof is extremely similar to the proof of Theorem 1 for the perfect completeness case. In the following we list the main differences.

- We "split up" the canonical paths so that each canonical path $P[j, \gamma, y]$ has a separate source node $S_{j,\gamma,y}$ and destination node $T_{j,\gamma,y}$. We now have at most MYJ demands, each of which is now associated with a *single* canonical path.
- We define $Z = 128b\rho^{b-1}$ instead of $Z = 128b\rho^b$. This change in the definition of Z allows us to obtain the improved hardness factor. As we explained in Section 3.4, Z needs to be more than ρ^{b-1} in order for the balls-in-bins argument to work. However, we had to make it larger than that (about ρ^b) since we allowed almost-canonical paths to deviate from canonical paths. Without almost-canonical paths, Z can be kept at roughly ρ^{b-1}.
- For yes-instances we can route MY demands on edge-disjoint canonical paths. Note that the total number of demands in the system is more than MY and so we no longer have perfect completeness.

- We no longer concern ourselves with the concept of an almost-canonical path that can deviate from a canonical path a limited number of times. We simply classify demands according to whether they use their entire canonical path or some other path.
- The analysis of canonical paths is almost identical to the analysis in Section 4.3. The only difference is that we no longer have to account for deviations from the canonical path which allows us to use the new definition of Z.
- The analysis of non-canonical paths is simpler under the new reduction. Note that in the proof of Theorem 1, each demand had multiple canonical paths. These paths created unavoidable cycles since they shared the same source and destination node. A large part of the analysis of Section 4.4.2 was concerned with showing that these cycles cannot be used to create short non-canonical paths. In the new reduction each demand has a single canonical path and so those cycles do not appear. In this case the analysis of Section 4.4.2 gives us the following simpler analog of Lemma 23.

  Lemma 28. *If a demand is routed along a path π (in H) which does not form a cycle in G, then the path π must be a canonical path.*

  This implies that if π is a non-canonical path then we only need to consider the case that π forms a cycle in G. This can be handled in the same manner as in Section 4.4.1. The remainder of the proof is exactly the same as for Theorem 1. In particular we obtain that for no-instances we can connect at most $MY/(\log V)^{\frac{1-\varepsilon}{c+1}}$ terminal pairs.

□

5.2 Node-Disjoint Paths with Congestion

In the undirected Node-Disjoint Paths with Congestion (NDPwC) problem we wish to route as many demands as possible subject to the constraint that at most c paths pass through any node. The reductions that we used to prove Theorems 1 and 2 for EDPwC apply directly to NDPwC. For yes-instances it is easy to see that the canonical paths that correspond to the satisfying assignment of the CSP instance are node-disjoint as well as edge-disjoint. The results for no-instances follow directly from the fact that any solution with edge-congestion c automatically has node-congestion at least c.

5.3 All-or-Nothing Flow with Congestion

All-or-Nothing Multicommodity Flow with Congestion is a relaxation of EDPwC in which each demand is allowed to be routed on fractional paths subject to the constraint that the total demand routed through an edge is at most c. The objective is to maximize the number of demands for which the *entire* demand is routed. There is no natural analog of our perfect completeness result (Theorem 1) for ANFwC since the multicommodity flow relaxation gives a polynomial-time algorithm to decide whether or not it is possible to route all demands fractionally with congestion c.

We can however adapt our non-perfect completeness result (Theorem 2). Note that in our construction without perfect completeness each demand has exactly one canonical path. For the

case of yes-instances we can route MY demands integrally on edge-disjoint paths. For the case of no-instances we classify any routed demand according to two types. We say that a routed demand is a *Type 1* demand if strictly more than half the demand is routed along its canonical path, otherwise it is a *Type 2* demand. Since the capacity of each edge is c, the total number of Type 1 demands whose canonical paths can be routed through any edge is at most $2c - 1$. Our canonical path analysis therefore implies that the total number of Type 1 demands is at most $MY/(\log V)^{\frac{1-\varepsilon}{2c}}$. Moreover, an almost identical argument to the analysis of Section 4.4 implies that the total amount of demand routed along non-canonical paths is at most $MY/(\log V)^{\frac{1-\varepsilon}{c+1}}$, *even if this demand is fractionally routed*. This in turn allows us to say that the total number of Type 2 demands is at most $2MY/(\log V)^{\frac{1-\varepsilon}{c+1}}$. The combination of these two results implies that for any ε, ANFwC is hard to approximate to within $(\log V)^{\frac{1-\varepsilon}{2c}}$.

6 Concluding remarks

We have shown a $\Omega(\log^{1/2-\varepsilon} V)$ inapproximability result for EDP, ANF, and node-disjoint paths (NDP) on undirected graphs, for any constant $\varepsilon > 0$. When congestion c, $1 \leqslant c \leqslant O(\frac{\log\log V}{\log\log\log V})$, is allowed in the routing, we obtained a hardness factor of $\log^{\frac{1-\varepsilon}{c+1}} V$ (and a slightly weaker $\log^{\frac{1-\varepsilon}{c+2}} V$ factor with perfect completeness, i.e., when one is promised that the instance has an edge-disjoint routing of *all* the source-destination pairs).

There is still a large gap between the hardness result for EDP and NDP and the best known ratio of $V^{\Omega(1)}$ achieved by polynomial time approximation algorithms. Closing this gap remains a central open question. For ANF, there is less of a gap, since a factor $O(\log^2 V)$ approximation algorithm is known [10, 12]. Another interesting point is that for the *edge-disjoint cycles* (EDC) problem on undirected graphs, where the goal is to pack a maximum number of cycles that are edge-disjoint, there is a $O(\sqrt{\log V})$ approximation algorithm [33]. It has been shown in [22] that our techniques for EDP hardness, specifically the version in [16], also yield a tight inapproximability factor of $\log^{1/2-\varepsilon} V$ for EDC. This is a rather surprising approximation threshold for a natural optimization problem and also highlights some limitations of our techniques. In order to improve our hardness factor for EDP, we need to develop techniques that will *not* work for EDC, and in order to get a hardness factor that is greater than $\text{poly}(\log V)$, we need techniques that are less general and won't apply for ANF. These intricacies make the challenge of pinning down the approximability of EDP on undirected graphs all the more important and exciting.

References

[1] M. Andrews. Hardness of buy-at-bulk network design. In *Proceedings of the 45th Annual Symposium on Foundations of Computer Science*, pages 115 – 124, Rome, Italy, October 2004.

[2] M. Andrews, J. Chuzhoy, S. Khanna, and L. Zhang. Hardness of the undirected edge-disjoint paths problem with congestion. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 226–244, 2005.

[3] M. Andrews and L. Zhang. Hardness of the undirected edge-disjoint paths problem. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 276–283, 2005.

[4] M. Andrews and L. Zhang. Hardness of the undirected congestion minimization problem. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 284–293, 2005.

[5] M. Andrews and L. Zhang. Hardness of the edge-disjoint paths problem with congestion. Manuscript, 2005. Available at: `http://cm.bell-labs.com/~andrews/pub.html`.

[6] M. Andrews and L. Zhang. Logarithmic hardness of the directed congestion minimization problem. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 517–526, 2006.

[7] Y. Aumann and Y. Rabani. An $O(\log k)$ approximate min-cut max-flow theorem and approximation algorithm. *SIAM Journal on Computing*, 27(1):291–301, February 1998.

[8] Y. Azar and O. Regev. Strongly Polynomial Algorithms for the Unsplittable Flow Problem. In *Proceedings of the 8th Integer Programming and Combinatorial Optimization Conference* pages 15-29, 2001.

[9] A. Baveja and A. Srinivasan. Approximation algorithms for disjoint paths and related routing and packing problems. *Mathematics of Operations Research*, Vol. 25, pp. 255–280, 2000.

[10] C. Chekuri, S. Khanna, and F. B. Shepherd. The all-or-nothing multicommodity flow problem. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 156–165, 2004.

[11] C. Chekuri, S. Khanna, and F. B. Shepherd. Edge-disjoint paths in Planar graphs with constant congestion. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 757-766, 2006.

[12] C. Chekuri, S. Khanna, and F. B. Shepherd. Multicommodity flow, well-linked terminals, and routing problems. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 183–192, 2005.

[13] C. Chekuri, S. Khanna, and F. B. Shepherd. An $O(\sqrt{n})$-approximation and integrality gap for disjoint paths and UFP. *Theory of Computing*, Vol. 2, 137–146, 2006.

[14] C. Chekuri and S. Khanna. Edge Disjoint Paths Revisited. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 628–637, 2003.

[15] C. Chekuri, M. Mydlarz, and F. B. Shepherd. Multicommodity Demand Flow in a Tree and Packing Integer Programs. Submitted. Preliminary version in *Proc. of ICALP*, 2003.

[16] J. Chuzhoy and S. Khanna. New hardness results for undirected edge disjoint paths, Manuscript, 2005. Available at: `http://www.cis.upenn.edu/~sanjeev/postscript/edpwc.ps.gz`

[17] J. Chuzhoy, V. Guruswami, S. Khanna, and K. Talwar. Hardness of routing with congestion in directed graphs. In *Proceedings of 39th Annual ACM Symposium on Theory of Computing*, pages 165-178, June 2007.

[18] L. Engebretsen. The nonapproximability of non-boolean predicates. *SIAM Journal on Discrete Mathematics*, 18:114–129, 2004.

[19] P. Erdös and H. Sachs. Reguläre graphen gegebener Taillenweite mit minimaler Knotenzahl. *Wiss. Z. Uni. Halle-Wittenburg (Math. Nat.)*, 12:251–257, 1963.

[20] A. Frank. Edge-disjoint paths in planar graphs. *J. of Combinatorial Theory, Ser. B.*, No. 2 (1985), 164-178.

[21] S. Fortune, J. Hopcroft and J. Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, Vol. 10, No. 2 (1980), pp. 111–121.

[22] Z. Friggstad and M. R. Salavatipour. Approximability of packing disjoint cycles. In *Proceedings of 18th International Symposium on Algorithms and Computation*, 2007, to appear.

[23] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, 1979.

[24] N. Garg, V. Vazirani, and M. Yannakakis. Primal-Dual Approximation Algorithms for Integral Flow and Multicut in Trees. *Algorithmica*, 18(1):3-20, 1997.

[25] V. Guruswami, S. Khanna, R. Rajaraman, F. B. Shepherd, and M. Yannakakis. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. *J. Comput. Syst. Sci.*, 67(3):473–496, 2003.

[26] V. Guruswami and K. Talwar. Hardness of low congestion routing in undirected graphs. *Manuscript*, 2005.

[27] J. Håstad and S. Khot. Query efficient PCPs with perfect completeness. *Theory of Computing*, 1(7):119-148, 2005.

[28] J. Håstad and A. Wigderson. Simple Analysis of Graph Tests for Linearity and PCP. *Random Structures and Algorithms*, Vol 22, no. 2, pp 139-160, 2003.

[29] J. M. Kleinberg and É. Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. *Journal of Computer and System Sciences*, 57:61–73, 1998.

[30] J. M. Kleinberg and É. Tardos. Disjoint Paths in Densely Embedded Graphs. *Proc. of FOCS*, pp. 52–61, 1995.

[31] J. M. Kleinberg. Approximation algorithms for disjoint paths problems. PhD thesis, MIT, Cambridge, MA, May 1996.

[32] S. G. Kolliopoulos and C. Stein. Approximating Disjoint-Path Problems Using Greedy Algorithms and Packing Integer Programs. IPCO 1998: 153-168.

[33] M. Krivelevich, Z. Nutov, and R. Yuster. Approximation algorithms for cycle packing problems. In *Proceedings of the 16thACM-SIAM Symposium on Discrete Algorithms*, pages 556-561, 2005.

[34] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, 1987.

[35] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.

[36] N. Robertson and P. D. Seymour. Outline of a disjoint paths algorithm. In B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, Eds., *Paths, Flows and VLSI-Layout*. Springer-Verlag, Berlin, 1990.

[37] A. Samorodnitsky and L. Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings of the 32nd Annual ACM Symposium on theory of Computing*, pages 191-199, 2000.

[38] A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. In *Proceedings of the 38th Symposium on Foundations of Computer Science*, pp. 416–425, 1997.

[39] K. Varadarajan and G. Venkataraman. Graph Decomposition and a Greedy Algorithm for Edge-disjoint Paths. *Proc. of SODA*, 2004.