

Hardness of the Undirected Congestion Minimization Problem

Matthew Andrews
andrews@research.bell-labs.com

Lisa Zhang
ylz@research.bell-labs.com

Bell Laboratories
600-700 Mountain Avenue
Murray Hill, NJ 07974

July 26, 2005

Abstract

We show that there is no $\frac{\gamma \log \log M}{\log \log \log M}$ -approximation for the undirected congestion minimization problem unless $NP \subseteq ZPTIME(n^{\text{polylog } n})$, where M is the size of the graph and γ is some positive constant.

1 Introduction

Consider a graph G with M edges and a set $\{(s_i, t_i)\}$ of source-sink pairs. The congestion minimization problem aims to connect all these pairs while minimizing the *edge congestion*, i.e. the maximum number of demands that go through the same edge in G . The problem is known to be NP-hard. The famous result of Raghavan and Thompson [6] states that by applying randomized rounding to a linear relaxation of the problem we obtain an $O(\log M / \log \log M)$ approximation for both directed and undirected graphs. On the hardness side, Chuzhoy and Naor show an $\Omega(\log \log M)$ hardness for directed graphs [3]. For undirected graphs, the only previously known hardness result is a factor $2 - \varepsilon$ hardness, for any $\varepsilon > 0$. (This follows from the fact that it is NP-hard to determine if all the terminals can be routed on edge-disjoint paths [5].)

In this paper we present a hardness result for undirected graphs that almost matches Chuzhoy and Naor's result for directed graphs. We refer to this congestion minimization problem on undirected graphs as MINCONGESTION. We show that for some constant $\gamma > 0$, MINCONGESTION is $\frac{\gamma \log \log M}{\log \log \log M}$ -hard to approximate unless $NP \subseteq ZPTIME(n^{\text{polylog}(n)})$.¹ Our proof also shows that the integrality gap for the natural linear programming relaxation of MINCONGESTION is $\Omega\left(\frac{\log \log M}{\log \log \log M}\right)$.

Raz Verifier. To show the hardness of MINCONGESTION we construct a reduction using the Raz verifier for MAX3SAT(5). A 3SAT(5) formula has n variables and $5n/3$ clauses where each variable appears in exactly 5 distinct clauses and each clause contains exactly 3 literals. The MAX3SAT(5) problem aims to find an assignment that maximizes the number of satisfied clauses. A 3SAT(5) formula is called a *yes-instance* if it is satisfiable; it is called a *no-instance* if no assignment satisfies more than a $1 - \varepsilon$ fraction of the clauses for some constant $\varepsilon > 0$. It follows from the PCP theorem [2] that it is NP-hard to distinguish between yes-instances and no-instances.

¹Recall that $ZPTIME(n^{\text{polylog}(n)})$ is the set of languages that have randomized algorithms that always give the correct answer and have expected running time $n^{\text{polylog}(n)}$.

A Raz verifier with ℓ repetitions is defined as follows [7, 3]. A verifier interacts with 2 provers, a clause prover (c-prover) and a variable prover (v-prover). Given a 3SAT(5) formula ϕ , the verifier sends the c-prover a clause query (c-query) that consists of ℓ clauses c_1, \dots, c_ℓ chosen uniformly at random. It also sends the v-prover a variable query (v-query) that consists of one variable v_1, \dots, v_ℓ chosen uniformly at random from each of the ℓ clauses. The c-prover sends back the assignment of every variable in clauses c_1, \dots, c_ℓ and the v-prover sends back the assignment of the variables v_1, \dots, v_ℓ . The verifier *accepts* ϕ if all the ℓ clauses are satisfied and the two provers give a consistent assignment to the ℓ variables. The verifier *rejects* ϕ otherwise.

Suppose ϕ has n variables, then ϕ has $5n/3$ clauses. Clearly, a Raz verifier with ℓ repetitions has $Q_c := (5n/3)^\ell$ distinct c-queries each of which has $A_c := 7^\ell$ answers. It also has $Q_v := n^\ell$ distinct v-queries each of which has $A_v := 2^\ell$ answers. Since the verifier only queries variables that appear in a c-query, the number of distinct clause-variable query pairs is $R := (5n)^\ell$. Each clause appears in $R/Q_c = 3^\ell$ c-v query pairs and each variable appears in $R/Q_v = 5^\ell$ c-v query pairs. These parameters are summarized in Section 2.2.

Theorem 1 [7] *There is a universal constant $\alpha > 1$ such that if ϕ is a yes-instance there is a proof system in which the verifier always accepts; if ϕ is a no-instance the verifier never accepts with probability higher than $\alpha^{-\ell}$.*

2 Construction

Given a 3SAT(5) formula ϕ we first construct the two-prover interactive proof system and represent it in a graph that we call the *proof system graph*, P . We then transform this proof system into a MINCONGESTION instance on a graph that we call the *transformed graph*, T . We show that if ϕ is a yes-instance then all demands can be routed with congestion 1. If ϕ is a no-instance then with high probability for any routing of the demands some edge has high congestion.

In defining these two graphs, we do not use the convention of specifying their node sets and edge sets. Instead we first specify a set of paths that exist in the graph. We then add edges to the graph to ensure that these paths are realizable. Our graph construction involves many parameters which we define in Section 2.2. We give an overview of the analysis in Section 2.3. Throughout the paper we use subscript c and v when discussing quantities pertaining to clauses and variables, and we omit the subscript when we do not distinguish them.

2.1 Graph Construction

In the proof system graph P , for each possible answer a we have an answer edge which we also denote a . For each query q we group together all the possible answers to q and refer to this group as a query blob which we also denote q . A clause query blob (c-blob) contains A_c edges that we call *answer* edges and a variable query blob (v-blob) contains A_v answer edges. For each c-v query pair r we have Y demands $d_{r,y}$, $1 \leq y \leq Y$, each of which has a source node $s_{r,y}$, a destination node $t_{r,y}$ and routes one unit of flow. For each accepting interaction (r, a_c, a_v) demand $d_{r,y}$ has a special path p that we refer to as a *canonical path*.

This path starts at node $s_{r,y}$, passes through edges a_c, a_v and ends at node $t_{r,y}$. In order for this to be possible, we place a *center* edge between a_c and a_v , a *demand* edge between $s_{r,y}$ and a_v , and a demand edge between a_c and $t_{r,y}$. Note that each demand has multiple canonical paths. Note also that we allow parallel demand edges and center edges so that no two canonical paths share a common edge other than answer edges.

To facilitate the analysis we label the canonical paths as follows. For each accepting interaction (r, a_c, a_v) there is a random Y -element permutation σ . The canonical path for demand $d_{r,y}$ that corresponds to accepting interaction (r, a_c, a_v) is labeled $p_{r,a_c,a_v,\sigma(y)}$. That is, σ induces a *random* matching between the demands $d_{r,y}$ and the canonical paths corresponding to the accepting interaction (r, a_c, a_v) .

Let us use an example to illustrate the above construction of P , shown in Figure 1. Let the repetition parameter $\ell = 1$. Let $\phi = (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee y \vee w)$. For a query pair r that queries clause $x \vee y \vee \bar{z}$ and variable x , there are 7 accepting interactions, namely the 7 satisfying assignments for the clause and the assignment of x consistent with the clause. Suppose $Y = 1$. We define one demand $d_{r,1}$ which has 7 canonical paths defined by the 7 accepting interactions. These 7 canonical paths are shown in solid lines in Figure 1. The demand for query pair r' that queries clause $\bar{x} \vee y \vee w$ and variable x also has 7 canonical paths and they share the answer edges in the v-blob x with demands $d_{r,1}$. These canonical paths are shown in dotted lines in Figure 1. We did not draw 4 other demands and their canonical paths.

If $Y > 1$ then for each query pair r we define Y demands, each of which has its own source node and destination node, and its own 7 canonical paths. The canonical paths for these Y demands share the answer edges in c-blob $x \vee y \vee \bar{z}$ and v-blob x , but have distinct center edges and demand edges.

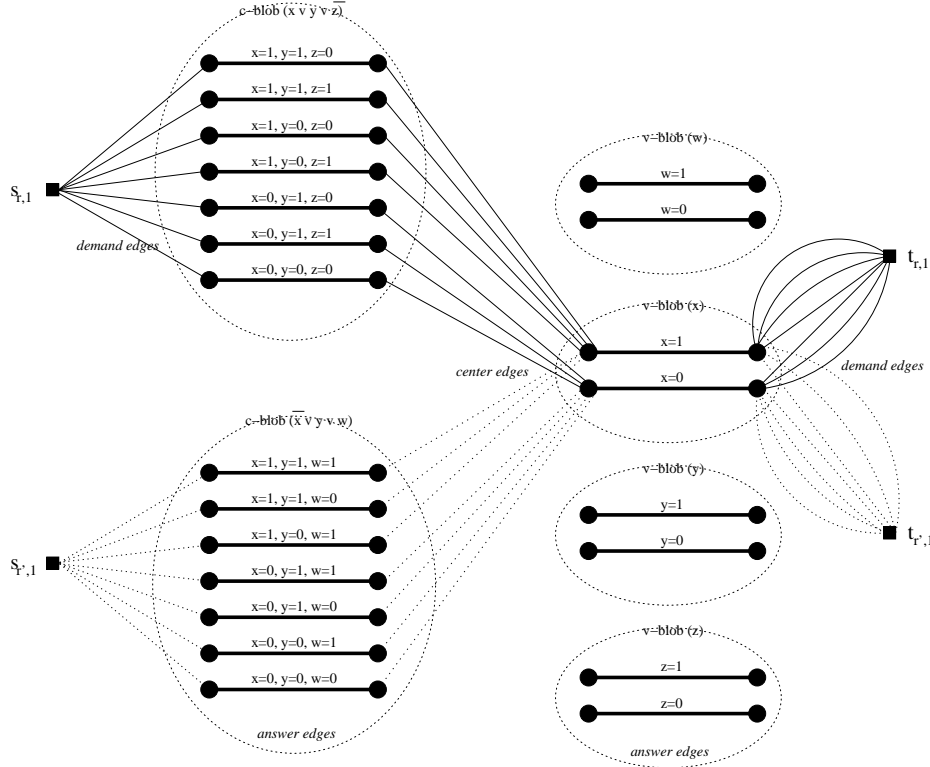


Figure 1: Graph P for a proof system, $\phi = (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee y \vee w)$, $\ell = 1$ and $Y = 1$. The figure shows 2 out of 6 demands.

We construct a MINCONGESTION instance on the transformed graph T . Like P we have Y demands $d_{r,y}$ for each c-v query pair r . Each demand has a source node $s_{r,y}$ and a destination node $t_{r,y}$. The graph T consists of Z levels. Each level of T consists of Q_c c-blobs and Q_v v-blobs and they correspond one-to-one to those in P . Each c-blob in T consists of I_c image edges and each v-blob consists of I_v image edges. We use $b_{q,z}$ to denote the blob at level z of T that corresponds to query blob q of P ; we use $f_{q,z,j}$ to denote the j th image edge in $b_{q,z}$.

In the following we describe how to map canonical paths from P to T . For any clause query blob q_c in

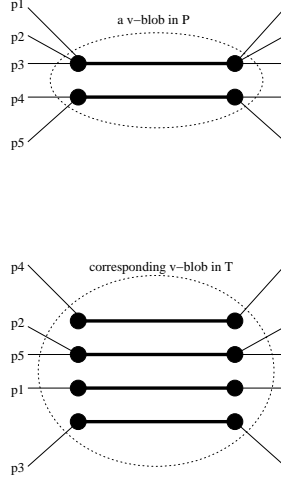


Figure 2: A possible mapping of 5 canonical paths from a v-blob in P to a v-blob in T . Paths p_1 , p_2 and p_3 are mapped to distinct image edges, paths p_4 and p_5 are mapped to distinct image edges. (The figure shows fewer than the actual number of canonical paths in P and fewer than the actual number of edges in T .)

P , $D_c := YR/Q_c$ demands have canonical paths that go through q_c . Since two canonical paths of the same demand cannot share a common answer edge in q_c , $I_c := D_c$ canonical paths go through any answer edge a_c in q_c . Consider all the canonical paths that go through a_c in P . We map each of these paths to a random image edge in $b_{q_c,z}$ subject to the constraint that no two of these paths are mapped to the same image edge. That is, there is a random matching between the canonical paths going through a_c and the image edges they are mapped to in $b_{q_c,z}$. This is always possible due to the choice of I_c .

We now perform a similar construction for each variable query q_v . Note that for a fixed clause and a fixed variable, the number of satisfying assignments of the clause for which the variable is set to 1 is either 3 or 4 and the number of satisfying assignments for which the variable is set to 0 is also either 3 or 4. This implies that for any v-blob in P , at most 4^ℓ canonical paths of the same demand can share a common answer edge. Therefore, at most $I_v := 4^\ell \cdot YR/Q_v$ canonical paths can go through any answer edge a_v in q_v . These canonical paths are randomly mapped to *distinct* image edges in $b_{q_v,z}$ at every level z . (See Figure 2.) We emphasize that the random mapping for blob $b_{q,z}$ is independent of the random mapping for blob $b_{q,z'}$ for any $1 \leq z < z' \leq Z$.

Consider a canonical path p for demand $d_{r,y}$ that goes through query blobs q_c and q_v in P . For notational convenience, if p passes through the image edge $f_{q_c,z,j}$ (resp. $f_{q_v,z,j}$) we sometimes write $f_{c,p,z} = f_{q_c,z,j}$ (resp. $f_{v,p,z} = f_{q_v,z,j}$); If p is for demand $d_{r,y}$ we sometimes write $d_p = d_{r,y}$, $s_p = s_{r,y}$ and $t_p = t_{r,y}$. The above discussion describes which image edges path p passes through in T . Under the new notation path p has the following form which gives the ordering of the image edges on the path:

$$s_p \rightarrow \dots \rightarrow f_{c,p,1} \rightarrow \dots \rightarrow f_{v,p,1} \rightarrow \dots \rightarrow f_{c,p,z} \rightarrow \dots \rightarrow f_{v,p,z} \rightarrow \dots \rightarrow f_{c,p,Z} \rightarrow \dots \rightarrow f_{v,p,Z} \rightarrow \dots \rightarrow t_p$$

We add edges to T to make sure that these paths are realizable. In particular we add edges between s_p and $f_{c,p,1}$, between $f_{c,p,z}$ and $f_{v,p,z}$ for all z , between $f_{v,p,z}$ and $f_{c,p,z+1}$ for all z and between $f_{v,p,Z}$ and t_p . We have now shown how to map a canonical path in P into a canonical path in T . As before, we may add parallel edges so that no two canonical paths share a common edge other than inside a blob. This completes the description of the graph T and a MINCONGESTION instance on T . (See Figure 3.)

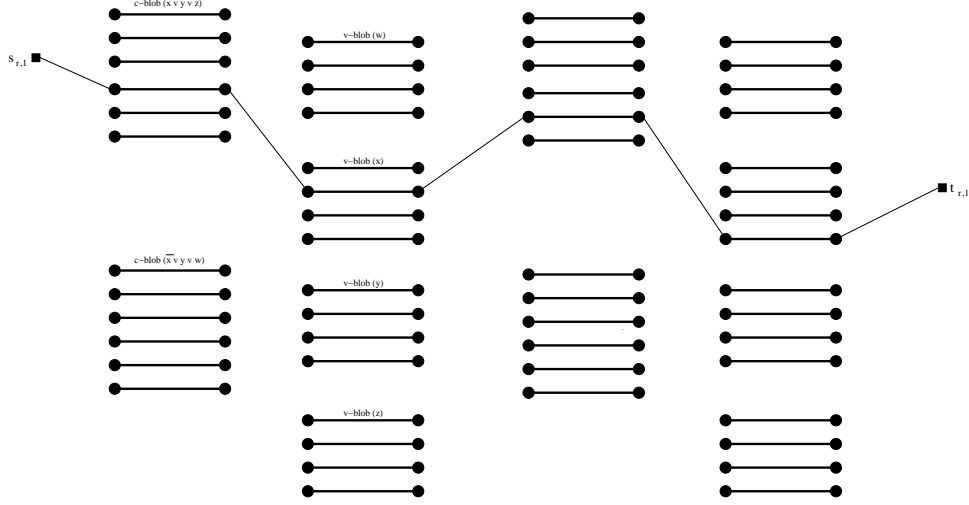


Figure 3: The transformed graph T from graph P shown in Figure 1, $Z = 2$. The figure shows 1 out of 7 canonical paths for demand $d_{r,1}$. Note also that the figure shows fewer than the actual number of edges per v-blob.

2.2 Parameters

Given a $3SAT(5)$ formula ϕ with n variables, we choose the Raz verifier repetition parameter ℓ to be

$$\ell = \beta \log \log \log n \quad \text{for a sufficiently large constant } \beta. \quad (1)$$

We use the following parameters in constructing graphs P and T .

$$Q_c = (5n/3)^\ell \quad \text{no. of c-blobs in } P \text{ and per level in } T; \text{ no. of c-queries} \quad (2)$$

$$Q_v = n^\ell \quad \text{no. of v-blobs in } P \text{ and per level in } T; \text{ no. of v-queries} \quad (3)$$

$$A_c = 7^\ell \quad \text{no. of answer edges per c-blob in } P; \text{ no. of answers to a c-query} \quad (4)$$

$$A_v = 2^\ell \quad \text{no. of answer edges per v-blob in } P; \text{ no. of answers to a v-query} \quad (5)$$

$$I_c = Y \cdot 3^\ell \quad \text{no. of image edges per c-blob in } T \quad (6)$$

$$I_v = Y \cdot 20^\ell \quad \text{no. of image edges per v-blob in } T \quad (7)$$

$$R = (5n)^\ell \quad \text{no. of c-v query pairs} \quad (8)$$

$$YR \quad \text{no. of demands}$$

$$D_c = Y \cdot 3^\ell \quad \text{no. of demands per c-blob in } P \text{ and } T \quad (9)$$

$$D_v = Y \cdot 5^\ell \quad \text{no. of demands per v-blob in } P \text{ and } T \quad (10)$$

In order to define Y , the number of demands per c-v query, and Z , the number of levels in T , we introduce two new parameters, k and h . The parameter k is related to the length of non-canonical paths in T and h is used to define the concept of *heavy blobs* in P .

$$h = \ell^{-1} \cdot \log \log n \quad \text{definition of heavy blob in } P \quad (11)$$

$$Z = (40 \cdot 8^\ell)^{h+1} \quad \text{no. of levels in } T \quad (12)$$

$$k = Z\sqrt{Z} \quad \text{non-canonical path length in } T \quad (13)$$

$$Y = (6RQ_cA_cQ_vA_vZ)^{2k+1} \quad \text{no. of demands per c-v query pair} \quad (14)$$

Let M be the number of edges in T . We have YRZ image edges from c-blobs and $YRZ \cdot 4^\ell$ image edges from v-blobs. Canonical paths do not share common edges outside blobs in T and there are $7^\ell(2Z + 1)YR$ such edges. Therefore,

$$M = YRZ + YRZ \cdot 4^\ell + 7^\ell(2Z + 1)YR. \quad (15)$$

To get a sense of how big Z , Y and M are, we note that $\ell h = \log \log n$ and therefore $Z = \text{polylog}(n)$, $Y = e^{\text{poly } Z}$ which is $e^{\text{polylog}(n)}$. Finally, since every term in M is $e^{\text{polylog}(n)}$, M is $e^{\text{polylog}(n)}$ as well. We discuss how the values of these parameters are chosen in Section 3.3, after we present the proof.

2.3 Overview of Proof

At a high level our proof proceeds as follows. We use a solution to the MINCONGESTION instance on T to determine whether the original 3SAT(5) instance ϕ (that defined the proof system) is a yes-instance or a no-instance. The most important feature of our construction is that for all the canonical paths that pass through the same answer edge in query blob q in P , the corresponding canonical paths in T all pass through different image edges in $b_{q,z}$ in T . Therefore, if all demands are routed on canonical paths then for any solution with high congestion in P , the corresponding solution in T typically has low congestion. Conversely, for any solution with low congestion in P the corresponding solution in T typically has high congestion.

If ϕ is a yes-instance it is straightforward to argue that demands can be routed in T so that the congestion is 1. The two provers can always answer any query pair using one satisfiable assignment of ϕ and the verifier always accepts. Therefore, these accepting interactions define one canonical path per demand and only one answer edge in each query blob of P is used. By construction, when these canonical paths are mapped from P to T these paths are mapped to distinct image edges in every blob in T . Therefore, the corresponding solution in T has congestion 1.

We concentrate on the case that ϕ is a no-instance. If all demands are routed on canonical paths only in T (resp. in P), we say this solution is canonical in T (resp. in P). For a canonical solution to T , we show in Section 3.1 that the corresponding canonical solution to P must have demands routed through a large number of answer edges in *some* blob. Otherwise, a small number of answer edges would be used in *every* blob and we could use them to define a pair of provers that would violate the error probability of the proof system. (In the extreme example of a satisfiable ϕ , only one answer edge is used in every query blob in P and they define the two provers.) We then show that for the blob in P in which many answer edges are used, one of the Z corresponding blobs in T necessarily has high congestion of $\Omega(h)$ with high probability.

In Section 3.2 we consider the case in which non-canonical paths may be used. We say that a canonical path p for demand d_p is *regular* at level z in T if d_p is routed through both $f_{c,p,z}$ and $f_{v,p,z}$. We say a demand is *regular* at level z if it has a canonical path regular at level z ; a demand is regular at ζ levels if it has a canonical path that is regular at ζ levels. We show that for most demands they are either routed on a long path in T or else they are regular at many levels. The proof of this fact uses girth properties of T and is similar to an analysis of the Buy-at-Bulk network design problem presented in [1].

If many demands are routed on long paths it is easy to show that the congestion must be high on some edge. If many demands are regular at many levels we generalize the argument for canonical solutions in Section 3.1 (i.e. demands regular at all levels) to show that the solution in T must have high congestion of $\Omega(h)$. Therefore, we have shown a gap of $\Omega(h)$ in congestion depending on whether or not ϕ is a yes-instance or a no-instance.

3 No-instances

For the case that ϕ is a no-instance we now carry out the detailed analysis outlined previously.

3.1 Special Case: Canonical Paths Only

Let us begin with some notation. Let \mathcal{M} be the particular random mapping between the canonical paths in P and in T . If \mathcal{T} is a canonical solution to the MINCONGESTION instance on T , then \mathcal{T} and \mathcal{M} define a canonical solution to P which we denote $\mathcal{P}(\mathcal{T}, \mathcal{M})$. Conversely, if \mathcal{P} is a canonical solution to P , \mathcal{P} and \mathcal{M} define a canonical solution to T which we denote $\mathcal{T}(\mathcal{P}, \mathcal{M})$.

Under a canonical solution \mathcal{P} , we say that an answer edge in a blob in P is *heavy* if at least $D/(10A)$ demands go through the edge. (As before whenever there is no need to specify if a blob is a c-blob or v-blob we omit the subscripts c and v . For example, A denotes the number of edges in the blob of interest and D denotes the number of demands that go through the blob.) In addition, we say a query blob is heavy if it has at least h heavy answer edges; we say a solution \mathcal{P} is heavy if it has at least one heavy query blob. We refer to anything as *light* if it is not heavy. For a canonical solution \mathcal{T} to T , we first show that $\mathcal{P}(\mathcal{T}, \mathcal{M})$ has to be a heavy solution under any mapping \mathcal{M} . Otherwise, we choose *one* edge from the heavy answer edges in each blob and these chosen answer edges define two provers. We then show that since every blob is light, many demands are routed along the chosen answer edges only. Furthermore, since canonical paths correspond to accepting interactions of the proof system, it is a contradiction for a no-instance ϕ if there are more such paths than the error probability of the proof system would allow. We then show that with high probability, every heavy canonical solution corresponds to a canonical solution in T that has high congestion. This probability is taken over all the random mappings \mathcal{M} .

Lemma 2 *If \mathcal{T} is a canonical solution, P has a heavy blob under $\mathcal{P}(\mathcal{T}, \mathcal{M})$.*

Proof: For the purpose of contradiction let us assume $\mathcal{P}(\mathcal{T}, \mathcal{M})$ is a light solution. Then every blob in P has fewer than h heavy answer edges. For a particular query blob q fewer than $A \cdot D/(10A)$ demands can go through light edges in q . Summing over all the v-blobs and c-blobs we have at most $YR/5$ demands whose routes contain light answer edges. Hence at least $4/5$ of all demands route through heavy answer edges only under solution $\mathcal{P}(\mathcal{T}, \mathcal{M})$.

We now choose one heavy edge uniformly at random in each query blob in P . At least $(4YR/5) \cdot h^{-2}$ demands are expected to go through these randomly chosen heavy edges only. Hence there exists a choice of heavy edges such that at least $(4YR/5) \cdot h^{-2}$ demands go through these heavy edges. We refer to these demands as *accepting demands* and the rest as *unaccepting*. Recall that we have defined Y demands per c-v query pair. Under solution $\mathcal{P}(\mathcal{T}, \mathcal{M})$, some of the Y demands that correspond to the same random string may be accepting whereas the rest may be unaccepting. If this is the case for any random string, we reroute the unaccepting demands along the path of the accepting ones. Therefore, under this new solution either all Y demands of a random string are accepting or all Y demands are unaccepting. Moreover, a fraction of at least $4/(5h^2)$ of all demands are accepting.

Recall that canonical paths are defined by accepting interactions. Since accepting demands are routed along canonical paths, we have shown that at least a $4/(5h^2)$ fraction of random strings generate queries that have accepting answers to the verifier. By the choice of ℓ in (1) and h in (11), $4/(5h^2)$ is higher than the error probability $\alpha^{-\ell}$ defined in Theorem 1. This is a contradiction for a no-instance ϕ . Note that the choice of ℓ is the smallest possible to ensure the contradiction here. \square

We study the congestion in T that occurs when $\mathcal{P}(\mathcal{T}, \mathcal{M})$ has a heavy blob. The mapping of canonical paths between P and T corresponds to the following balls-and-bins game. We are given I bins and we throw n_i balls into n_i distinct bins during the i th round. For $D = n_1 + \dots + n_A$ and $I \geq D$ what is the maximum number of balls in a bin at the end of A rounds?

To see the connection to our problem, let us consider how demands are routed in one query blob q in P under a solution $\mathcal{P}(\mathcal{T}, \mathcal{M})$. The n_i 's represent the number of demands that are routed along the i th answer

edge in q ; D represents the total number of demands routed through q ; A represents the number of edges in q and I represents the number of edges in a blob in T that corresponds to q . By construction these n_i paths are mapped to distinct edges in every corresponding blob in T . This is the same as throwing n_i balls into I bins and each ball lands in a distinct bin. Therefore, the maximum number of balls per bin is the maximum edge congestion in one corresponding blob in T .

Without loss of generality, let us assume $n_1 \geq n_2 \dots \geq n_A$. Let x be such that $n_x \geq D/(10A) > n_{x+1}$. If $n_A > D/(10A)$ then $x = A$. Such an x always exists.

Lemma 3 *The probability that every bin has fewer than x balls is at most $e^{-(20Aa)^{-x-1}D}$ where $a = I/D$.*

Proof: We compute the probability that some bin has x balls after the first x rounds and ignore the balls thrown after x rounds. It is easy to see that for a particular bin to have a ball in round i equals n_i/I since n_i distinct bins out of I are chosen at random. Therefore, this probability is at least $1/(10Aa)$ for bins $i \leq x$ since $n_i \geq D/(10A)$ and $a = I/D$. Since each round is independent, the probability that one bin has x balls or more is at least $(10Aa)^{-x}$. Equivalently, the probability that one bin has fewer than x balls is at most $1 - (10Aa)^{-x}$.

In order to bound the probability that every bin has fewer than x balls, we deal with the dependence among the bins in the following way. Consider bin j where $j \leq D/(20A)$.

$$\begin{aligned}
& \Pr [\text{Bin } j \text{ has at least } x \text{ balls} \mid \text{Any configuration of bin } 1, \dots, j-1] \\
& \geq \Pr [\text{Bin } j \text{ has at least } x \text{ balls} \mid \text{Bin } 1, \dots, j-1 \text{ each has } x \text{ balls}] \\
& = \prod_{1 \leq i \leq x} \frac{n_i - j}{I - j} \\
& \geq \prod_{1 \leq i \leq x} \frac{n_i - D/(20A)}{I} \\
& \geq (20Aa)^{-x}
\end{aligned}$$

Therefore,

$$\Pr [\text{Bin } j \text{ has fewer than } x \text{ balls} \mid \text{Any configuration of bin } 1, \dots, j-1] \leq 1 - (20Aa)^{-x}.$$

Using conditional probability, we have

$$\begin{aligned}
\Pr [\text{Every bin has fewer than } x \text{ balls}] & \leq \Pr [\text{Bins } 1, \dots, D/(20A) \text{ each has fewer than } x \text{ balls}] \\
& \leq (1 - (20Aa)^{-x})^{D/(20A)} \leq e^{-(20Aa)^{-x-1}D}.
\end{aligned}$$

□

We now define a set of bad events. Let q be a query blob in P , let $H = \{a_1, \dots, a_h\}$ be a set of h answer edges and let E_{a_i} , for $1 \leq i \leq h$, be a set of $D/10A$ canonical paths in P that pass through the answer edge a_i . Let $B(q, E_{a_1}, \dots, E_{a_h})$ be the bad event that under \mathcal{M} , the images of the paths in E_{a_1}, \dots, E_{a_h} in T create congestion less than h .

Recall in the construction of P at most 4^ℓ canonical paths of the same demand can share the same answer edge in a v-blob in P and no two canonical paths of the same demand can share the same answer edge in a c-blob in P . Therefore, the value of a in Lemma 3 is 4^ℓ for a v-blob and $a = 1$ for a c-blob. Recall also that for every blob in P we create Z consecutive blobs in T , and the mappings of the canonical paths from P to these Z blobs are independent from one another. Finally, imagine that for each E_{a_i} a demand is routed along each path in E_{a_i} . In this case the value of x (defined above) satisfies $x \geq h$. Therefore, Lemma 3 and the choice of Z immediately imply,

Corollary 4 For fixed $q, E_{a_1}, \dots, E_{a_h}$, the probability that $B(q, E_{a_1}, \dots, E_{a_h})$ occurs is at most

$$\begin{aligned} e^{-(20A_c)^{-h-1}D_c Z} &\leq e^{-D_c} && \text{if } q \text{ is a c-blob,} \\ e^{-(20A_v 4^\ell)^{-h-1}D_v Z} &\leq e^{-D_v} && \text{if } q \text{ is a v-blob.} \end{aligned}$$

This probability is with respect to the random mapping \mathcal{M} .

We now count the number of bad events. If q is a v-blob then at most $D_v 4^\ell$ canonical paths pass through any answer edge in q . Therefore, an upper bound on the total number of events of the form $B(q, E_{a_1}, \dots, E_{a_h})$ for a v-blob q is

$$\begin{aligned} Q_v \binom{A_v}{h} \left(\frac{D_v 4^\ell}{D_v / (10A_v)} \right)^h &\leq Q_v \cdot (A_v)^h \cdot (10A_v 4^\ell e)^{hD_v / (10A_v)} \\ &= e^{\log Q_v} \cdot e^{h \log A_v} \cdot e^{(1 + \log 10 + \log A_v + \log 4^\ell) h D_v / (10A_v)} \\ &= e^{o(D_v)}. \end{aligned}$$

The last equality holds since the exponent in each of the three terms is $o(D_v)$. If q is a c-blob at most D_c canonical paths pass through any answer edge in q . A similar (but simpler) calculation shows that an upper bound on the total number of events of the form $B(q, E_{a_1}, \dots, E_{a_h})$ for a c-blob q is

$$Q_c \binom{A_c}{h} \left(\frac{D_c}{D_c / (10A_c)} \right)^h = e^{o(D_c)}.$$

By a union bound, the probability that *some* event $B(q, E_{a_1}, \dots, E_{a_h})$ happens is at most,

$$e^{-D_c} \cdot Q_c \binom{A_c}{h} \left(\frac{D_c}{D_c / (10A_c)} \right)^h + e^{-D_v} \cdot Q_v \binom{A_v}{h} \left(\frac{D_v 4^\ell}{D_v / (10A_v)} \right)^h \leq e^{-D_v} \cdot e^{o(D_v)} + e^{-D_c} \cdot e^{o(D_c)},$$

which is at most $1/\text{poly}(n)$.

Now suppose that under mapping \mathcal{M} no such bad event occurs. For any heavy solution \mathcal{P} in P , by definition we can find a query blob q with h answer edges such that for each such edge $D/10A$ demands are routed on a canonical path that passes through the edge. Since no bad event occurs, the images of these canonical paths in T create congestion at least h . In other words, the canonical solution $\mathcal{T}(\mathcal{P}, \mathcal{M})$ in T has congestion at least h . We have therefore shown,

Lemma 5 With probability $1 - 1/\text{poly}(n)$ every heavy solution \mathcal{P} corresponds to a solution $\mathcal{T}(\mathcal{P}, \mathcal{M})$ in which the congestion is at least h in T .

Lemma 2 states that if \mathcal{T} is a canonical solution then $\mathcal{P}(\mathcal{T}, \mathcal{M})$ must be a heavy solution in P . Therefore, by Lemma 5, with high probability over the choice of \mathcal{M} , \mathcal{T} must have congestion at least h since $\mathcal{T} = \mathcal{T}(\mathcal{P}(\mathcal{T}, \mathcal{M}), \mathcal{M})$. In summary,

Theorem 6 With probability $1 - 1/\text{poly}(n)$, every canonical solution \mathcal{T} has congestion at least h .

3.2 General Case

We now consider the general problem in which a demand does not follow a canonical path p but takes a detour. Whenever this happens the canonical path p and the detour form a cycle in T . We would like to bound the number and the lengths of these cycles in T and thereby quantify the detours. However, a direct

analysis on T seems hard mainly due to the cycles formed by two complete canonical paths of the same demand. To facilitate the analysis, we create an *incidence graph* G for which we are able to show the number of small cycles is likely to be small. This allows us to show that for most demands, either they are routed along long paths in T or else they are regular at many levels. We show the congestion in T is high in both cases.

Recall a canonical path p is regular at level z if the demand is routed through both $f_{c,p,z}$ and $f_{v,p,z}$, i.e. it does not take a detour from the two image edges along p at level z ; a demand is regular at ζ levels if it has a canonical path regular at ζ levels.

3.2.1 Incidence Graph

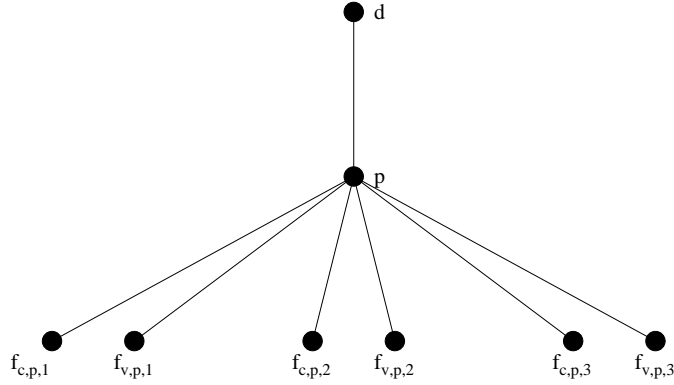


Figure 4: A portion of the incidence graph G , showing one demand d and one of d 's canonical paths p .

We begin with a description of the incidence graph. The incidence graph G has a node for each demand d , each canonical path p and each image edge $f_{q,z,j}$. If path p is a canonical path for demand d , (i.e. $d_p = d$) then there is an edge between d and p in G . If path p passes through $f_{q,z,j}$, (i.e. $f_{c,p,z} = f_{q,z,j}$ or $f_{v,p,z} = f_{q,z,j}$) then there is an edge between p and $f_{q,z,j}$. Figure 4 shows a portion of G that corresponds to one canonical path of a demand. (Recall we have two ways of denoting an image edge in T . The notation $f_{c,p,z}$ refers to the image edge in the c-blob at level z along path p ; $f_{q,z,j}$ refers to the j th image edge in the z th blob that corresponds to blob q of P .)

We now show how each edge in T maps into a path in G . Each image edge in T maps to a single node in G which we call an *image node*. Each edge that connects two image edges (or, an image edge and a source/destination node) along a canonical path p in T maps to the two-edge path that connects the two image nodes (or, the image node and the demand node d_p) via the path node p . We refer to these paths in G as *route components*. More formally, the route components are defined as follows.

- The image edge $f_{i,p,y}$ in T maps into the single image node $f_{i,p,y}$ in G .
- The edge between $f_{c,p,z}$ and $f_{v,p,z}$ maps into the path $f_{c,p,z} \rightarrow p \rightarrow f_{v,p,z}$.²
- The edge between $f_{v,p,z}$ and $f_{c,p,z+1}$ maps into the path $f_{v,p,z} \rightarrow p \rightarrow f_{c,p,z+1}$.
- The edge between s_p and $f_{c,p,1}$ maps into the path $d \rightarrow p \rightarrow f_{c,p,1}$.

²If the route in T goes from $f_{c,p,z}$ to $f_{v,p,z}$ then the path in G goes from node $f_{c,p,z}$ to node p to node $f_{v,p,z}$. If the route in T goes from $f_{v,p,z}$ to $f_{c,p,z}$ then the path in G goes in the opposite direction. A similar statement regarding the orientation of the path in G holds for each of the other mappings.

- The edge between $f_{v,p,Z}$ and t_p maps into the path $f_{v,p,Z} \rightarrow p \rightarrow d$.

The above definition of route components implies the following relationship between the path lengths in T and in G .

Lemma 7 *For any route in T of length x , the corresponding route in G has length at most $2x$.*

We can assume without loss of generality that the route in T is *simple* (i.e. it never visits the same node twice). This implies that although the route in G is not necessarily simple, it can traverse each of the above route components at most once.

Before presenting the details, we first discuss how T and G are related at a high level. Consider a demand d and let π be the path that d is routed along in T . The first edge in π , i.e. the one that leaves the source node s_d , belongs to a canonical path, say p . Suppose that p and π are not identical. This implies that the union of p and π form a cycle in T . There are two scenarios to consider, depending on whether or not $\pi(G)$ contains a cycle in G , where $\pi(G)$ is the mapping of path π in graph G .

- **Case 1: $\pi(G)$ contains a cycle in G .**

In this case we can show that either $\pi(G)$ contains a small cycle and the demand node d is close to this small cycle; or $\pi(G)$ contains a small cycle but d far away from any such small cycle; or $\pi(G)$ contains large cycles only. If the first situation holds then we say the demand is of type (1); if the latter two situations hold the demand is of type (2). We show in Lemma 11 that G is likely to only have a small number of small cycles due to the randomness in our construction. Consequently, the number of type-(1) demands is probably small. We also have that long paths in G correspond to long paths in T , hence if d is a type-(2) demand then π is a long path in T . Using this fact we show in Lemma 12 that if there are many type-(2) demands the congestion in T has to be high.

- **Case 2: $\pi(G)$ does not contain a cycle in G .**

We show in Lemma 8 that the only way this scenario can happen is if an image edge in p is detoured along a cycle formed by two complete canonical paths of a different demand d' . (See Figure 5.) (We remark that we allow “nested” detours in that some of the image edges on the detour may themselves be detoured.) In other words, if a non-image edge in p is detoured or if an image edge in p is detoured in a different manner, then $\pi(G)$ has a cycle in G .

If π detours from p at more than \sqrt{Z} levels, we say d is of type (3); otherwise d is of type (4). Since each detour of an image edge in this scenario has length longer than Z , type-(3) demands are routed along long paths of length longer than $Z\sqrt{Z}$. If there are many type-(3) demands the congestion in T has to be high, as shown in Lemma 12. A type-(4) demand is routed along a path π that resembles the canonical path p in most places. In particular, the path π passes through the same image edges as p at more than $Z - \sqrt{Z}$ levels, i.e. the path p is regular at more than $Z - \sqrt{Z}$ levels. In Section 3.2.5 we focus on the $Z - \sqrt{Z}$ regular levels and carry out an analysis similar to the special case of canonical paths only.

3.2.2 Demands of 4 Types

We now formally show that demands have four types. The proof is similar to an analysis presented in [1].

Lemma 8 *Every demand $d_{r,y}$ has at least one of the four following types, where k is the parameter defined in (13).*

1. $\pi(G)$ contains a cycle of length at most k in G and the demand node $d_{r,y}$ is at most distance k from this cycle.

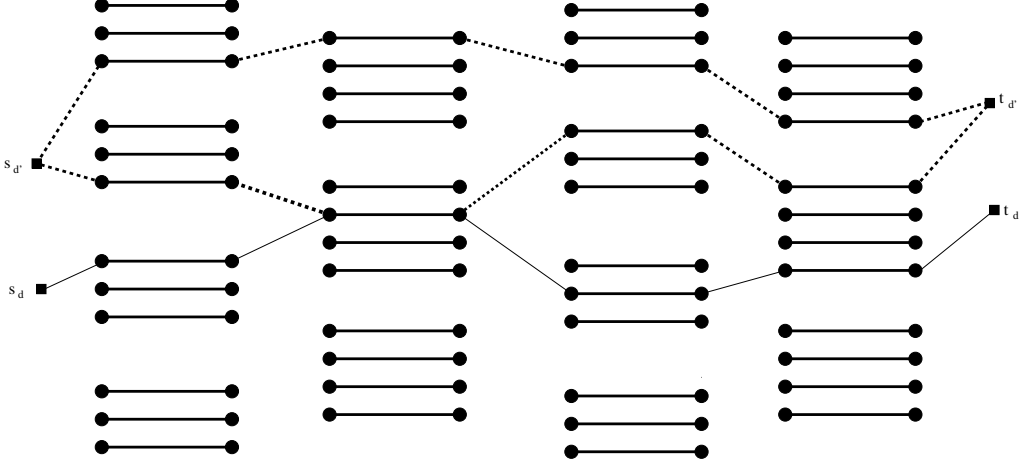


Figure 5: An image edge, $f_{v,p,1}$, along canonical path p for demand d is detoured along a cycle formed by two complete canonical paths of demand d' . The cycle is shown in dotted lines. Image edges on the dotted cycle can be detoured along other such cycles.

2. $\pi(G)$ contains a cycle in G but demand $d_{r,y}$ is not of type-(1). This implies that the length of $\pi(G)$ is more than k in G .
3. Demand $d_{r,y}$ is regular at at most $Z - \sqrt{Z}$ levels in T . This implies the length of π is more than $Z\sqrt{Z}$ in T .
4. Demand $d_{r,y}$ is regular at more than $Z - \sqrt{Z}$ levels in T .

Proof: If the demand $d_{r,y}$ is type (2), then either $\pi(G)$ contains a cycle longer than k , or $\pi(G)$ contains cycles of lengths at most k but $d_{r,y}$ is more than distance k from any of these cycles. In both cases the length of $\pi(G)$ is more than k . If the demand $d_{r,y}$ is not of types (1) or (2), then $\pi(G)$ does not have a cycle in G . Under this situation, we use the following fact repeatedly to show that if an edge along π deviates from p then it must be an image edge and it detours along a cycle formed by two complete canonical paths of a different demand. Such detours can be nested. (See Figure 5.) This implies the demand must be type (3) or (4).

Fact 9 Consider a route in a graph such that the subgraph induced by the route does not contain a cycle. Then, if the route leaves node v along edge e , it cannot return to node v except along edge e . More generally, if the route traverses edge e at any time after visiting node v , it cannot return to node v without traversing edge e in the opposite direction.

Consider now the route for demand $d_{r,y}$ in T . The route in T must begin at the source node for $d_{r,y}$ and then cross one of the edges connecting the source node to an answer edge at level 1. Recall that the route in G must follow the route components and it can traverse each route component at most once. Hence the route in G must begin $d_{r,y} \rightarrow p \rightarrow f_{c,p,1}$ for some path p that is owned by $d_{r,y}$. The route in G must return to node $d_{r,y}$ since the route in T must eventually reach the destination node for $d_{r,y}$. By Fact 9 this return to $d_{r,y}$ must happen via node p . Fact 9 also implies that on this return to node p the previous node must be $f_{c,p,1}$. Since each route component can be used at most once, the route in G must have the form,

$$d_{r,y} \rightarrow p \rightarrow f_{c,p,1} \rightarrow \Gamma_{c,1} \rightarrow f_{c,p,1} \rightarrow p \rightarrow f_{v,p,1} \dots$$

for some subroute $\Gamma_{c,1}$ that does not pass through p . Once again, Fact 9 implies that the route must return to node p in order to get back to node $d_{r,y}$. It must do this through node $f_{v,p,1}$. Therefore, because we can use each of our route components at most once, the route in G must have the form,

$$d_{r,y} \rightarrow p \rightarrow f_{c,p,1} \rightarrow \Gamma_{c,1} \rightarrow f_{c,p,1} \rightarrow p \rightarrow f_{v,p,1} \rightarrow \Gamma_{v,1} \rightarrow f_{v,p,1} \rightarrow p \rightarrow f_{c,p,2} \dots$$

for some subroutes $\Gamma_{c,1}$ and $\Gamma_{v,1}$ that do not pass through p . We can repeat this argument inductively to show that the route in G must have the form,

$$\begin{aligned} d_{r,y} \rightarrow p \rightarrow f_{c,p,1} \rightarrow \Gamma_{c,1} \rightarrow f_{c,p,1} \rightarrow p \rightarrow f_{v,p,1} \rightarrow \Gamma_{v,1} \rightarrow f_{v,p,1} \rightarrow p \rightarrow f_{c,p,2} \dots \\ \vdots \\ f_{c,p,z} \rightarrow \Gamma_{c,z} \rightarrow f_{c,p,z} \rightarrow p \rightarrow f_{v,p,z} \rightarrow \Gamma_{v,z} \rightarrow f_{v,p,z} \rightarrow p \rightarrow f_{v,p,z+1} \dots \\ \vdots \\ f_{c,p,Z} \rightarrow \Gamma_{c,Z} \rightarrow f_{c,p,Z} \rightarrow p \rightarrow f_{v,p,Z} \rightarrow \Gamma_{v,Z} \rightarrow f_{v,p,Z} \rightarrow p \rightarrow d_{r,y} \end{aligned}$$

for some subroutes $\Gamma_{c,1}, \dots, \Gamma_{c,z}, \Gamma_{v,z}, \dots, \Gamma_{v,Z}$ that do not pass through p .

We now consider the subroutes $\Gamma_{c,z}$ and $\Gamma_{v,z}$. There are two cases.

- **A:** Neither $\Gamma_{c,z}$ nor $\Gamma_{v,z}$ passes through path node p' for any $p' \neq p$. This can only happen if the route for demand $d_{r,y}$ passes through both $f_{c,p,z}$ and $f_{v,p,z}$. Hence path p is regular at level z .
- **B:** Either $\Gamma_{c,z}$ or $\Gamma_{v,z}$ pass through some path node $p' \neq p$. Suppose that this is true for $\Gamma_{c,z}$. (The other case is similar.) Then, $\Gamma_{c,z}$ either has the form,

$$\dots f_{c,p',z} \rightarrow p' \rightarrow f_{v,p',z-1} \dots$$

or else has the form

$$\dots f_{c,p',z} \rightarrow p' \rightarrow f_{v,p',z} \dots$$

The two cases can be treated similarly. We focus on the former. By Fact 9 since the subroute $\Gamma_{c,z}$ lies between two visits to p this subroute must return to p' via $f_{v,p',z-1}$. Therefore $\Gamma_{c,z}$ must have the form

$$\begin{aligned} \dots f_{c,p',z} \rightarrow p' \rightarrow f_{v,p',z-1} \dots \\ \dots f_{v,p',z-1} \rightarrow p' \rightarrow f_{c,p',z-1} \dots \end{aligned}$$

By continuing the argument inductively in this manner (as we did for path p), we can show that $\Gamma_{c,z}$ must eventually have the form $\dots p' \rightarrow d_{p'} \dots$. However, Fact 9 implies that the route must return to p' via $d_{p'}$, i.e. the route must have the form $\dots p' \rightarrow d_{p'} \dots d_{p'} \rightarrow p' \dots$. The only way this can happen is if the route for demand $d_{r,y}$ in T passes through source node $s_{p'}$ as well as destination node $d_{p'}$. Any route in T that does this must have length at least Z .

If situation A occurs for more than \sqrt{Z} levels then path p is regular at more than $Z - \sqrt{Z}$ levels, i.e. the demand has type (4). Otherwise situation B occurs for at least \sqrt{Z} levels which implies that π , the route for demand $d_{r,y}$ in T , has length at least $Z\sqrt{Z}$, i.e. the demand has type (3). \square

3.2.3 Demands of Type (1)

Since G is constructed in a random fashion, most nodes in G are far from small cycles of length at most k . More formally, let $B(G)$ be the bad event that there are more than Y type-(1) demands. In this section we show that $B(G)$ does not happen with a constant probability. The analysis resembles that presented in [1]. We begin with a result whose proof is similar to an argument used in the proof of the Erdős-Sachs theorem [4]. (The Erdős-Sachs theorem states that *high-girth* graphs exist. The girth of a graph is the length of its shortest cycle.)

Lemma 10 *Consider a random graph with ν nodes and let $\{e_0, e_1, \dots, e_{\kappa-1}\}$ represent a set of $\kappa < k$ potential edges. If,*

$$Pr[e_0 \text{ exists} | e_1, \dots, e_{\kappa-1} \text{ exist}] \leq \rho,$$

for all such sets of edges and $\nu\rho \geq 2$ then the expected number of cycles of fixed length $k' \leq k$ is at most $(\nu\rho)^{k'}$. This implies that with probability $\frac{2}{3}$, the number of cycles of length at most k is at most $3(\nu\rho)^{k+1}$.

Proof: The total number of potential cycles of length k' is at most $\frac{1}{2k'} \frac{\nu!}{(\nu-k')!}$. Each such cycle occurs with probability at most $\rho^{k'}$. Therefore, the expected number of cycles of length k' is at most,

$$\frac{\nu! \rho^{k'}}{2k'(\nu-k')!} \leq (\nu\rho)^{k'}.$$

This implies that the expected number of cycles of length less than or equal to k is,

$$\sum_{k'=1}^k (\nu\rho)^{k'} \leq (\nu\rho)^{k+1},$$

since $\nu\rho \geq 2$. By Markov's inequality, with probability $\frac{2}{3}$ the number of cycles of length at most k is at most $3(\nu\rho)^{k+1}$. \square

Lemma 11 *$Pr[B(G)] \leq \frac{1}{3}$, i.e. with probability at least $\frac{2}{3}$ the number of type-(1) demands is at most Y .*

Proof: We calculate ν and ρ for the incidence graph G . We begin by counting the number of nodes, ν . The graph G has

- RY demand nodes, $d_{r,y}$;
- at most $A_c A_v RY$ path nodes, p ;
- at most $(Q_c I_c + Q_v I_v)Z$ image nodes.

Therefore, the number of nodes in G is at most $\nu \leq RY + A_c A_v RY + (Q_c I_c + Q_v I_v)Z$ which is crudely upper bounded by $Q_c A_c Q_v A_v RY Z$. It is also easy to verify that each node has degree at most $\max\{A_c A_v, 2Z + 1, A_c, A_v\}$ which is crudely upper bounded by $A_c A_v Z$. We now calculate the probability that a potential edge in G exists. If a fixed set of up to k edges already exist,

- the probability that demand node $d_{r,y}$ is connected to path node p is at most $1/(Y - k)$;
(This is because for any accepting interaction (r, a_c, a_v) there is a random matching between the demands $d_{r,y}$ and the canonical paths corresponding to (r, a_c, a_v) . See Section 2.1.)
- the probability that path node p is connected to any given image node is at most $1/(\min\{I_c, I_v\} - k) \leq 1/(Y - k)$.

Hence for the incidence graph G , $\rho = 1/(Y - k)$.

By the definitions of Y and k in (14) and (13) we ensure that Y is (exponentially) larger than k . Hence $2 \leq \nu\rho \leq 2RQ_cA_cQ_vA_vZ$. Lemma 10 implies that with probability $\frac{2}{3}$, the number of cycles of length at most k is at most $(6RQ_cA_cQ_vA_vZ)^{k+1}$.

There are at most k nodes on each such cycle. By our degree bound each node is within distance k of at most $(A_cA_vZ)^k$ other nodes. Hence with probability $\frac{2}{3}$ the number of nodes in G that are within distance k of a node that is part of a cycle of length at most k is at most,

$$k(6Q_cA_cQ_vA_vRZ)^{k+1}(A_cA_vZ)^k \leq (6Q_cA_cQ_vA_vRZ)^{2k+1} = Y.$$

□

3.2.4 Demands of Types (2) and (3)

Lemma 12 *If the number of type-(2) demands, or the number of type-(3) demands, is at least $RY/4$, then some edge in T has congestion $\Omega(h)$.*

Proof: Each of the type-(2) demands has a route of length at least k in G which by Lemma 7 implies that they have a route of length at least $k/2$ in T . Therefore the total number of edges used by these routes is at least $RYk/8$. By (15) the number of edges in T is at most $RY7^\ell(3Z + 1)$. Therefore some edge has congestion at least,

$$\frac{k}{8 \cdot 7^\ell(3Z + 1)} = \frac{Z((40 \cdot 8^\ell)^{h+1})^{\frac{1}{2}}}{8 \cdot 7^\ell(3Z + 1)} = \Omega(h).$$

Each of the type-(3) demands has a route of length at least $Z\sqrt{Z}$ in T . If the number of type-(3) demands is at least $RY/4$ then some edge has congestion at least,

$$\frac{Z\sqrt{Z}}{4 \cdot 7^\ell(3Z + 1)} = \frac{Z((40 \cdot 8^\ell)^{h+1})^{\frac{1}{2}}}{4 \cdot 7^\ell(3Z + 1)} = \Omega(h).$$

□

3.2.5 Type-(4) Demands

Suppose the bad event $B(G)$ does not occur and the situation in Lemma 12 does not apply, then there are fewer than Y type-(1) demands, fewer than $RY/4$ type-(2) demands and fewer than $RY/4$ type-(3) demands. Since there are RY demands in total, Lemma 8 implies at least $RY/4$ demands are of type (4). The routing of these type-(4) demands resembles the special case analyzed in Section 3.1 as each demand is routed along a path that coincides with a canonical path of the demand at most levels. We use \tilde{T} to denote the ‘‘almost canonical’’ routing of these type-(4) demands in T and \mathcal{T} to denote the canonical solution to T that resembles \tilde{T} . Following the notation of Section 3.1 we use $\mathcal{P}(\mathcal{T}, \mathcal{M})$ and $\mathcal{T}(\mathcal{P}, \mathcal{M})$ to relate the canonical solutions in P and T where \mathcal{M} defines the mapping between the two graphs. Therefore, each demand of type (4) has 3 routes of interest: one under solution \tilde{T} , one under \mathcal{T} and one under \mathcal{P} . We are ready to reapply the analysis of Section 3.1 to show \tilde{T} has high congestion. We first show two lemmas analogous to Lemma 2.

Lemma 13 *If type (4) has more than $RY/4$ demands then P has a heavy blob under $\mathcal{P}(\mathcal{T}, \mathcal{M})$.*

Proof: For the purpose of contradiction let us assume every blob in P is light. At most $A_cD_c/(10A_c)$ demands pass through light edges in any c-blob in P ; at most $A_vD_v/(10A_v)$ demands pass through light

edges in any v-blob in P . Summing over all blobs, at most $RY/5$ demands in total pass through light edges in P . Therefore, at least

$$\frac{RY}{4} - \frac{RY}{5} = \frac{RY}{20}$$

demands pass through heavy edges in P and they do not induce a heavy solution.

We choose one heavy edge uniformly at random in each query blob in P . At least $RY/(20h^2)$ demands are expected to go through these randomly chosen heavy edges only. As argued in the proof of Lemma 2, this implies that at least a fraction of $1/(20h^2)$ query pairs have accepting answers to the verifier. By the choice of h , $1/(20h^2)$ is higher than the error probability $\alpha^{-\ell}$. This contradicts Theorem 1 since ϕ is a no-instance. Therefore, the $RY/4$ canonical paths must form a heavy solution in P . \square

As in Section 3.1 we now define a set of bad events. Let q be a query blob in P , let $H = \{a_1, \dots, a_h\}$ be a set of h answer edges and let E_{a_i} , for $1 \leq i \leq h$, be a set of $D/10A$ canonical paths in P that pass through the answer edge a_i . For each such path p let L_p be a set of $Z - \sqrt{Z}$ levels that we say are *marked* for path p . Let L be the set of all sets L_p . Let $B(q, E_{a_1}, \dots, E_{a_h}, L)$ be the bad event that under \mathcal{M} , the images in T of the paths in E_{a_1}, \dots, E_{a_h} have congestion less than $h/4$, *even if a path only contributes to congestion at a level at which it is marked*. We wish to analyze the probability of these bad events. The analysis is complicated by the fact that different paths may be marked at different levels. However, we make use of the following fact.

Lemma 14 *For any $q, E_{a_1}, \dots, E_{a_h}, L$ we can find a set of $Z/4$ levels such that at each of these levels z there are $h/4$ sets E_{a_i} that have at least $D/20A$ canonical paths that are marked at level z .*

Proof: Consider one particular set E_{a_i} . We first show that there exist $Z/2$ levels each of which has $D/(20A)$ paths in E_{a_i} that are marked at this level. (We stress that these paths need not be the same at every level.) To see this, the number of marked levels per path multiplied by the number of paths in E_{a_i} is at least $(Z - \sqrt{Z})D/(10A)$. Let x_z be the number of marked paths at level z . If $Z/2$ of these x_z 's are smaller than $D/(20A)$, then $\sum_z x_z$ is less than $(Z/2)D/(20A) + (Z/2)D/(10A)$ which is less than $(Z - \sqrt{Z})D/(10A)$. This is a contradiction.

For each level z we now let y_z be the number of sets E_{a_i} that have $D/(20A)$ paths marked at level z . We have just shown that $\sum_z y_z \geq hZ/2$. If $3Z/4$ of the y_z 's are smaller than $h/4$, then $\sum_z y_z < (3Z/4)(h/4) + (Z/4)h < hZ/2$ which is a contradiction. Therefore, we have $Z/4$ levels such that at each level z there are $h/4$ sets E_{a_i} each of which has at least $D/20A$ canonical paths marked at level z . (We stress that these $h/4$ sets may be different at different levels.) \square

We can now analyze the probability that bad event $B(q, E_{a_1}, \dots, E_{a_h}, L)$ occurs by concentrating on the $Z/4$ levels whose existence is guaranteed by Lemma 14. We derive the probability by applying a small variation of Lemma 3. The variation is due to the fact that for a heavy blob we now only consider $Z/4$ levels instead of Z levels in T , we only consider $h/4$ heavy edges instead of h per level and we only consider $D/(20A)$ paths instead of $D/(10A)$ on a heavy edge at each level. We obtain,

Lemma 15 *For fixed $q, E_{a_1}, \dots, E_{a_h}, L$, the probability that $B(q, E_{a_1}, \dots, E_{a_h}, L)$ occurs is at most,*

$$\begin{aligned} e^{-(40A_c)^{-h/4-1} D_c Z/4} &\leq e^{-D_c Z^{2/3}} && \text{if } q \text{ is a } c\text{-blob,} \\ e^{-(40A_v 4^\ell)^{-h/4-1} D_v Z/4} &\leq e^{-D_v Z^{2/3}} && \text{if } q \text{ is a } v\text{-blob.} \end{aligned}$$

For any path, the number of ways to choose $Z - \sqrt{Z}$ marked levels is $\binom{Z}{\sqrt{Z}}$. Therefore, the total number of events of the form $B(q, E_{a_1}, \dots, E_{a_h}, L)$ for a v-blob q is upper bounded by,

$$Q_v \binom{A_v}{h} \binom{D_v 4^\ell}{D_v/(10A_v)}^h \binom{Z}{\sqrt{Z}}^{hD_v/(10A_v)} = e^{o(D_v)} \cdot e^{(\log \sqrt{Z} + 1)\sqrt{Z} D_v h/(10A_v)} = e^{o(Z^{2/3} D_v)}.$$

The first equality holds due to the analysis in Section 3.1. Similarly, the total number of events of the form $B(q, E_{a_1}, \dots, E_{a_h}, L)$ for a c-blob q is upper bounded by,

$$Q_c \binom{A_c}{h} \binom{D_c}{D_c/(10A_c)}^h \left(\frac{Z}{\sqrt{Z}} \right)^{hD_c/(10A_c)} = e^{o(Z^{2/3}D_c)}.$$

Hence by a union bound, the probability that *some* event $B(q, E_{a_1}, \dots, E_{a_h}, L)$ happens is at most $e^{-D_c Z^{2/3}} e^{o(D_c Z^{2/3})} + e^{-D_v Z^{2/3}} e^{o(D_v Z^{2/3})}$, which is $1/\text{poly}(n)$.

Now suppose that under mapping \mathcal{M} no such bad event occurs. Consider the solution \tilde{T} in T and the related canonical solution \mathcal{T} . For any canonical path in \mathcal{T} we mark $Z - \sqrt{Z}$ levels at which the path is regular in the solution \tilde{T} . (Since all demands routed in \tilde{T} are type-(4) demands this is always possible.) An important observation is that the congestion in \tilde{T} is at least the congestion in T *under the assumption that a path only contributes to congestion at a level at which it is marked*. This is because a path in \tilde{T} and the related canonical path in \mathcal{T} pass through the same image edges at all levels at which the canonical path is marked.

We know from Lemma 13 that the solution $\mathcal{P}(\mathcal{T}, \mathcal{M})$ is heavy. By definition we can find a query blob q with h answer edges such that for each such edge $D/10A$ demands are routed on a canonical path that passes through the edge. Since no bad event occurs for this particular \mathcal{M} , the images of these canonical paths in T have congestion at least $h/4$ even if a path only contributes to congestion at a level at which it is marked. By our earlier remarks this implies that the solution \tilde{T} has congestion at least $h/4$. We now have the equivalent of Theorem 6.

Theorem 16 *If type (4) has more than $RY/4$ demands then with probability $1 - 1/\text{poly}(n)$, \tilde{T} has congestion at least $h/4$.*

Combining Lemmas 8, 11, 12, and Theorem 16, we have

Theorem 17 *If ϕ is a no-instance, T has congestion $\Omega(h)$ with a constant probability.*

3.3 Wrapping Up

Recall that if ϕ is a yes-instance then all demands can be routed with congestion 1. Therefore the “gap” between the congestion in the yes-instance case and the congestion in the no-instance case is $\Omega(h)$. We now describe this gap in terms of the size of T . Recall that M , the number of edges in T , is defined in (15). From the parameter definitions in Section 2.2 it is easy to see that $\log M = O(\log Y)$ and $\log \log M = O(\log Z)$. Since $\log Z = \Theta(\ell h)$, $\ell = \Theta(\log \log \log n)$ and $h = \Theta(\log \log n / \log \log \log n)$ we have,

$$\log \log M / \log \log \log M = O(h).$$

We also compute the complexity of our reduction. As discussed at the end of Section 2.2 we have

$$M = e^{\text{polylog}(n)},$$

i.e. the reduction can be carried out in quasipolynomial time. We have therefore shown that for some constant γ , there is no $\gamma \log \log M / \log \log \log M$ approximation for the undirected congestion minimization problem unless $NP \subseteq \text{coRTIME}(n^{\text{polylog}(n)})$. A standard result states that if $NP \subseteq \text{coRTIME}(n^{\text{polylog}(n)})$ then $NP \subseteq \text{ZPTIME}(n^{\text{polylog}(n)})$, i.e. NP has randomized algorithms that always give the correct answer and have quasi-polynomial running time.

Theorem 18 *There is no $\frac{\gamma \log \log M}{\log \log \log M}$ -approximation for MINCONGESTION unless $NP \subseteq \text{ZPTIME}(n^{\text{polylog}(n)})$, where M is the size of the graph and γ is some positive constant.*

Remarks. Our construction utilized a large number of parameters. We now briefly discuss how the values of these parameters are chosen. Recall ℓ is the repetition parameter of the Raz verifier. We need the error probability $\alpha^{-\ell}$ to be smaller than $4/(5h^2)$ to reach a contradiction in Lemma 2. We also require ℓh to be $O(\log \log n)$ so that the reduction is quasipolynomial. In addition, we have shown that $\Theta(h)$ is the inapproximability gap. Therefore, we choose $h = \Theta(\log \log n / \log \log \log n)$, which is as large as possible. This in turn implies that $\ell = \Theta(\log \log \log n)$. Once ℓ is fixed, the Raz verifier related parameters (2)-(10) are fixed as well.

The value of Z is determined by Corollary 4 and Lemma 15 in the probabilistic analysis. The value of Y upper bounds the number of type-(1) demands, and is determined in Lemma 11. The value of k is essentially $Z \cdot \zeta$ where $Z - \zeta$ is the number of regular levels in the definition of type-(3) and type-(4) demands. We note that ζ cannot be too small since Lemma 12 requires $\zeta = \Omega(h \cdot 7^\ell)$. Meanwhile, ζ cannot be too big since the union bound that counts the number of bad events of the form $B(q, E_{a_1}, \dots, E_{a_h}, L)$ has a factor $\binom{Z}{\zeta}^{hD_v/(10A_v)}$, and the number of bad events needs to be smaller than $e^{D_c Z^{2/3}}$. We choose $\zeta = \sqrt{Z}$. Our analysis works for $\zeta = Z^c$ where c can be any constant in $(0, 2/3)$, though the exact choice of the exponent does not affect the inapproximability gap of $\Theta(h)$.

3.4 Integrality Gap

We conclude by showing that our example also gives a lower bound on the integrality gap for MINCONGESTION. As we mentioned in the Introduction, applying randomized rounding to the linear relaxation of MINCONGESTION yields a logarithmic approximation of the problem [6]. We observe that our construction of the problem instance on T yields a fractional solution of congestion at most 1, even for no-instances. To see this, note that each demand has 7^ℓ canonical paths. Suppose that each canonical path carries a $7^{-\ell}$ fraction of the demand. By construction, if two canonical paths share an image edge in T their corresponding canonical paths in P cannot share an answer edge in P . Since a c-blob in P has 7^ℓ answer edges and a v-blob has 2^ℓ answer edges, at most 7^ℓ canonical paths can share an image edge in T . Therefore, the congestion on an image edge in T is at most 1. The congestion on other edges is $7^{-\ell}$ since canonical paths can only share image edges in T .

As outlined above, for no-instances all integral solutions have congestion $\Omega(h)$ with high probability. Therefore, there exists an instance for which there is a fractional solution with 1 but for all integral solutions the congestion is $O(h) = O(\log \log M / \log \log \log M)$. Hence,

Corollary 19 *The linear relaxation of MINCONGESTION has an integrality gap of $\frac{\gamma \log \log M}{\log \log \log M}$ for some positive constant γ .*

3.5 Conclusions

In this paper we have shown that for some constant γ there is no $\gamma \log \log M / \log \log \log M$ approximation for the undirected congestion minimization problem unless $NP \subseteq ZPTIME(n^{\text{polylog}(n)})$. Our techniques also apply to directed graphs. However, the resulting gap of $\gamma \log \log M / \log \log \log M$ is weaker than that of [3].

A number of open problems remain. First, there is still an exponential gap between the best-known $O(\log M / \log \log M)$ -approximation algorithm for MINCONGESTION and our $\gamma \log \log M / \log \log \log M$ hardness factor. Second, our reduction is quasipolynomial and randomized. It would be interesting to obtain a deterministic polynomial-time reduction. Although we believe that it would be possible to use deterministic constructions of high-girth graphs to create a deterministic instance of MINCONGESTION in which the bad event $B(G)$ does not hold, we know of no way to do this in such a way that the bad events $B(q, E_{a_1}, \dots, E_{a_h}, L)$ do not hold.

Acknowledgements

The authors wish to thank Phil Whiting and Carl Nuzman for their help in analyzing the balls-and-bins game in Section 3.1.

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] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.
- [3] J. Chuzhoy and J. Naor. New hardness results for congestion minimization and machine scheduling. In *Proceedings of the 36th Annual ACM Symposium of Theory of Computing*, pages 28 – 34, Chicago, IL, 2004.
- [4] P. Erdős and H. Sachs. Reguläre graphen gegebener Tailenweite mit minimaler Knotenzahl. *Wiss. Z. Uni. Halle-Wittenburg (Math. Nat.)*, 12:251–257, 1963.
- [5] R.M. Karp. Reducibility among combinatorial problems. In *R.E. Miller and J.W. Thatcher, editors, Complexity of Computer Computations*, pages 85 – 103, 1972.
- [6] P. Raghavan and C.D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365 – 374, 1991.
- [7] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.