THEORY OF COMPUTING, Volume 13 (20), 2017, pp. 1–25
www.theoryofcomputing.org

The Inapproximability of Maximum Single-Sink Unsplittable, Priority and Confluent Flow Problems

F. Bruce Shepherd*   Adrian R. Vetta†

Received September 18, 2015; Revised May 17, 2017; Published December 24, 2017

Abstract: While the maximum single-sink unsplittable and confluent flow problems have been studied extensively, algorithmic work has been primarily restricted to the case where one imposes the *no-bottleneck assumption (NBA)* (that the maximum demand $d_{\max}$ is at most the minimum capacity $u_{\min}$). For instance, under the NBA there is a factor-4.43 approximation algorithm due to Dinitz et al. (1999) for the unsplittable flow problem. Under the even stronger assumption of uniform capacities, there is a factor-3 approximation algorithm due to Chen et al. (2007) for the confluent flow problem. We show, however, that unlike the unsplittable flow problem, a constant-factor approximation algorithm cannot be obtained for the single-sink confluent flow problem even *with* the no-bottleneck assumption. Specifically, we prove that it is NP-hard to approximate single-sink confluent flow to within $O(\log^{1-\varepsilon}(n))$, for any $\varepsilon > 0$.

The remainder of our results focus upon the setting *without* the no-bottleneck assumption. Using exponential-size demands, Azar and Regev prove a $\Omega(m^{1-\varepsilon})$ inapproximability result for maximum cardinality single-sink unsplittable flow in directed graphs. We prove that this

*Supported by an NSERC Discovery Research Grant.
†Supported by an NSERC Discovery Research Grant.

ACM Classification: G.1.6, G.2.2

AMS Classification: 68Q17, 68W25, 68Q25

Key words and phrases: algorithms, complexity theory, multiflow, network flow, unsplittable flow, confluent flow, priority flow

 DOI: 10.4086/toc.2017.v013a020

lower bound applies to undirected graphs, including planar networks (and for confluent flow). This is the first super-constant hardness known for undirected single-sink unsplittable flow. Furthermore, we show $\Omega(m^{1/2-\varepsilon})$-hardness even if all demands and capacities lie within an arbitrarily small range $[1,1+\Delta]$, for $\Delta > 0$. This result is sharp in that if $\Delta = 0$, then it becomes a single-sink maximum edge-disjoint paths problem which can be solved exactly via a maximum flow algorithm. This motivates us to study maximum *priority flows* for which we show the same inapproximability bound.

1 Introduction

In this paper we improve known lower bounds (and upper bounds) on the approximability of the maximization versions of the *single-sink unsplittable flow*, *single-sink priority flow* and *single-sink confluent flow* problems. In the single-sink network flow problem, we are given a directed or undirected graph $G = (V,E)$ with n nodes and m edges that has edge capacities $u(e)$ or node capacities $u(v)$. There are a collection of demands that have to be routed to a unique destination *sink* node t. Each demand i is located at a *source* node s_i (multiple demands could share the same source) and requests an amount d_i of flow capacity in order to route. We will primarily focus on the following two well-known versions of the single-sink network flow problem.

- `Unsplittable Flow.` Each demand i must be sent along a unique path P_i from s_i to t_i.
- `Confluent Flow.` Any two demands that meet at a node must then traverse identical paths to the sink. In particular, at most one edge out of each node v is allowed to carry flow. Consequently, the support of the flow is a tree in undirected graphs, and an arborescence rooted at t in directed graphs.

Unsplittable flows were introduced in [16] to study optical (SONET) ring networks. The single-sink version was introduced in [23, 22] and has led to a wealth of algorithmic developments. Confluent flows were introduced to study the effects of next-hop routing [12]. In that application, routers are capacitated and, consequently, nodes in the confluent flow problem are assumed to have capacities but not edges. In contrast, in the unsplittable flow problem it is the edges that are assumed to be capacitated. We follow these conventions in this paper. In addition, we also examine a third network flow problem called `Priority Flow` (defined in Section 1.2). In the literature, subject to network capacities, there are two standard maximization objectives.

- `Cardinality.` Maximise the total number of demands routed.
- `Throughput.` Maximise satisfied demand, that is, the total flow carried by the routed demands.

These objectives can be viewed as special cases of the *profit-maximisation* flow problem. There each demand i has a profit π_i in addition to its demand d_i. The goal is to route a subset of the demands of maximum total profit. The cardinality model then corresponds to the unit-profit case, $w_i = 1$ for every demand i; the throughput model is the case $\pi_i = d_i$. Clearly the lower bounds we will present also apply to the more general profit-maximisation problem.

1.1 Previous work

The unsplittable flow problem has been extensively studied since its introduction by Cosares and Saniee [16] and Kleinberg [23]. However, most positive results have relied upon the *no-bottleneck assumption* (NBA) where the maximum demand is at most the minimum capacity, that is, $d_{\max} \le u_{\min}$. Given the NBA, the best known result is a factor-4.43 approximation algorithm due to Dinitz, Garg and Goemans [17] for the maximum throughput objective.

The confluent flow problem was first examined by Chen, Rajaraman and Sundaram [12]. There, and in variants of the problem [11, 18, 31], the focus was on uncapacitated graphs.[1] The current best result for maximum confluent flow is a factor-3 approximation algorithm for maximum throughput in uncapacitated networks [11].

Observe that uncapacitated networks (i. e., graphs with uniform capacities) trivially also satisfy the NBA. Much less is known about networks where the NBA does **not** hold. This is reflected by the dearth of progress for the case of multiflows (that is, multiple sink) without the NBA. It is known that a constant-factor approximation algorithm exists for the case in which G is a path [5], and that a poly-logarithmic approximation algorithm exists for the case in which G is a tree [7]. The extreme difficulty of the unsplittable flow problem is suggested by the following result of Azar and Regev [3].

Theorem 1.1 (Azar–Regev)**.** *If* $\mathsf{P} \neq \mathsf{NP}$ *then, for any* $\varepsilon > 0$*, there is no* $O(m^{1-\varepsilon})$*-approximation algorithm for the cardinality objective of the single-sink unsplittable flow problem in directed graphs.*

This is the first (and only) super-constant lower bound for the maximum single-sink unsplittable flow problem.

1.2 Our results

The main focus of this paper is on single-sink flow problems where the no-bottleneck assumption (NBA) does not hold. It turns out that the hardness of approximation bounds are quite severe even in the (often more tractable) single-sink setting. In some cases they match the worst case bounds for PIPs (general packing integer programs). In particular, we strengthen Theorem 1.1 in several ways. First, as noted by Azar and Regev, the proof of their result relies critically on having directed graphs. We prove that it holds for undirected graphs and even *planar* undirected graphs. Second, we show that the result also applies to the confluent flow problem.

Theorem 1.2. *If* $\mathsf{P} \neq \mathsf{NP}$ *then, for any* $\varepsilon > 0$*, there is no* $O(m^{1-\varepsilon})$*-approximation algorithm for the cardinality objective of the maximum single-sink unsplittable and confluent flow problems in undirected graphs. Moreover for unsplittable flows, the lower bound holds even when we restrict to planar inputs.*

Third, Theorems 1.1 and 1.2 rely upon the use of exponentially large demands—we call this the *large-demand regime*. A second demand scenario that has received attention in the literature is the *polynomial-demand regime*. This is the regime considered in [21]. We show that strong hardness results apply in the polynomial-demand regime; in fact, they apply to the *small-demand regime* where the *demand spread* $d_{\max}/d_{\min} = 1+\Delta$, for some arbitrarily "small" constant $\Delta > 0$. (Note that $d_{\min} \le u_{\min}$ and so

[1] An exception concerns the analysis of graphs with constant treewidth [19].

the demand spread of an instance is at least the *bottleneck value* $d_{\max}/u_{\min}$.) In this regime we obtain similar hardness results for the throughput objective for the single-sink unsplittable and confluent flow problems. Formally, we show the following $m^{1/2-\varepsilon}$-inapproximability result. We note however that the hard instances have a linear number of edges so one may prefer to call this an $n^{1/2-\varepsilon}$-inapproximability result.

Theorem 1.3. *If* $\mathsf{P} \neq \mathsf{NP}$*, then the single-sink versions of maximum unsplittable and confluent flow problems cannot be approximated to within a factor of* $O(m^{1/2-\varepsilon})$*, for any* $\varepsilon > 0$*. This holds in either the cardinality or throughput objectives and for undirected and directed graphs even when instances are restricted to have demand spread* $d_{\max}/d_{\min} = 1+\Delta$*, where* $\Delta > 0$ *is an arbitrarily small constant.*

Again for the unsplittable flow problem this hardness result applies even in planar graphs. Theorems 1.2 and 1.3 are the first super-constant hardness for any undirected version of the single-sink unsplittable flow problem, and any directed version with small demands.

To clarify what is happening in our hardness results for small $\Delta > 0$, we introduce and examine an intermediary problem, the *maximum priority flow problem*. Here, we have a graph $G = (V,E)$ with a sink node t, and demands from nodes s_i to t. These demands are unit demands, and thus $\Delta = 0$. However, a demand may not traverse every edge. Specifically, we have a partition of E into k priority classes E_i. Each demand also has a *priority*, and a demand of priority i may only use edges of priority i or better (i. e., edges in $E_1 \cup E_2 \cup \dots E_i$). The goal is to find a maximum routable subset of the demands. Observe that, for this unit-demand problem, the throughput and cardinality objectives are identical. We view this as an intermediate problem since we can realize it as an instance of the small demand unsplittable flow problem. We first select numbers $1 > \delta_1 > \delta_2 > \dots > \delta_k > 0$ and then set demand i to have value $d_i = 1+\delta_i$. We then assign capacity $1+\delta_i$ to the edges of E_i.

Whilst various priority network design problems have been considered in the literature (cf. [6, 14]), we are not aware of existing results on maximum priority flow. Our results immediately imply the following.

Corollary 1.4. *The single-sink maximum priority flow problem cannot be approximated to within a factor of* $m^{1/2-\varepsilon}$*, for any* $\varepsilon > 0$*, in planar directed or undirected graphs.*

It was noted [21] that "*...the hardness of undirected edge-disjoint paths remains an interesting open question. Indeed, even the hardness of edge-capacitated unsplittable flow remains open.*"[2] A polylogarithmic lower bound later appeared in [2] for the maximum edge-disjoint paths (MEDP) problem (this was subsequently extended to the regime where edge congestion is allowed [1]). For the capacitated version, our results show that unsplittable flow suffers polynomial hardness (even for single-sink instances). Moreover, a polynomial lower bound for MEDP seems less likely given the recent $O(1)$-congestion polylog-approximation algorithms [13, 15]. In this light, our hardness results for single-sink unsplittable flow again highlight the sharp threshold involved with the NBA (or the special case of MEDP here). That is, if we allow some slight variation in demands and capacities within a tight range $[1, 1+\Delta]$ we immediately jump from (plausible) polylogarithmic approximations for MEDP to (known) polynomial hardness of the corresponding maximum unsplittable flow instances. A similar phenomenon holds for planar graphs

[2] In [21], they do however establish an inapproximability bound of $n^{1/2-\varepsilon}$, for any $\varepsilon > 0$, on *node-capacitated* USF in undirected graphs.

where MEDP has a constant-factor approximation with congestion 2 [10, 30] and constant-factor (with no congestion) for high-diameter planar graphs [24]. In contrast, our results imply polynomial hardness when some slight demand variation is allowed.

We next note that Theorems 1.1 and 1.2 are stronger than Theorem 1.3 in the sense that they have exponents of $1-\varepsilon$ rather than $1/2-\varepsilon$. Again, this extra boost is due to their use of exponential demand sizes. One can obtain a more refined picture as to how the hardness of cardinality single-sink unsplittable/confluent flow varies with the demand sizes, or more precisely how it varies on the bottleneck value $d_{\max}/u_{\min}$.[3]

Theorem 1.5. *Consider any fixed $\varepsilon > 0$ and $d_{\max}/u_{\min} > 1$. It is* NP*-hard to approximate cardinality single-sink unsplittable/confluent flow to within a factor of $O(m^{1/2-\varepsilon} \cdot \sqrt{\log(d_{\max}/u_{\min})})$ in undirected or directed graphs. For unsplittable flow, this remains true for planar graphs.*

Once again we see the message that there is a sharp cutoff for $d_{\max}/u_{\min} > 1$ even in the large-demand regime. This is because if the bottleneck value is at most 1, then the no-bottleneck assumption holds and, consequently, the single-sink unsplittable flow problem admits a constant-factor approximation (not $\sqrt{m}$ hardness). We mention that a similar hardness bound cannot hold for the maximum throughput objective, since one can always reduce to the case where $d_{\max}/u_{\min}$ is small with a polylogarithmic loss, and hence the lower bound becomes at worst $O(m^{1/2-\varepsilon} \cdot \log m)$. We feel the preceding hardness bound is all the more interesting since known greedy techniques yield almost-matching upper bounds, even for general multiflows.

Theorem 1.6. *There is an $O(\sqrt{m}\log(d_{\max}/u_{\min}))$ approximation algorithm for cardinality unsplittable flow and an $O(\sqrt{m}\log n)$ approximation algorithm for throughput unsplittable flow, in both directed and undirected graphs.*

We next focus on confluent flows assuming the NBA. Recall that for the maximum single-sink unsplittable flow problem there is a constant-factor approximation algorithm given the NBA. This is not the case for the single-sink confluent flow problem as we provide a super-constant lower bound. Its proof is more complicated but builds on the techniques used for our previous results.

Theorem 1.7. *Given the* NBA*, the single-sink confluent flow problem cannot be approximated to within a factor $O(\log^{1-\varepsilon} n)$, for any $\varepsilon > 0$, unless* P $=$ NP*. This holds for both the maximum cardinality and maximum throughput objectives in undirected and directed graphs.*

An almost-matching polylogarithmic approximation has subsequently been found for maximum confluent flow with the NBA [32].

Finally, we include a hardness result for the congestion minimization problem for confluent flows. That is, the problem of finding the minimum value $\alpha \geq 1$ such that all demands can be routed confluently if all node capacities are multiplied by α. This problem has two plausible variants.

[3] This seems likely connected to a footnote in [3] that a lower bound of the form

$$O\left(m^{1/2-\varepsilon} \cdot \sqrt{\log(d_{\max}/u_{\min})}\right)$$

exists for maximum unsplittable flow in directed graphs. Its proof was omitted however.

A *strong congestion* algorithm is one where the resulting flow must route on a tree T such that for any demand v the nodes on its path in T must have capacity at least $d(v)$. A *weak congestion* algorithm does not require this extra constraint on the tree capacities. Both variants are of possible interest. If the motive for congestion is to route all demands in some limited number α of rounds of admission, then each round should be feasible on T—hence strong congestion is necessary. On the other hand, if the objective is to simply augment network capacity so that all demands can be routed, weak congestion is the right notion. In Section 3.1.3 we show that it is hard to approximate strong congestion to within polynomial factors.

Theorem 1.8. *It is* NP*-hard to approximate the minimum (strong) congestion problem for single-sink confluent flow instances (with polynomial-size demands) to factors of at most $m^{1/2-\varepsilon}$ for any $\varepsilon > 0$.*

1.3 Overview of paper

At the heart of our reductions are gadgets based upon the *capacitated* 2-disjoint paths problem. We discuss this problem in Section 2. In Section 3, we establish a $\sqrt{m}$ inapproximability result for maximum single-sink unsplittable/confluent flow in the small-demand regime (Theorem 1.3); we give a similar hardness for single-sink priority flow (Corollary 1.4). Using a similar basic construction, we then establish, in Section 4, a polylogarithmic inapproximability result for maximum single-sink confluent flow even given the NBA (Theorem 1.7). In Section 5, we give lower bounds on the approximability of the cardinality objective for general-demand regimes (Theorems 1.2 and 1.5). Finally, in Section 6, we present an almost matching upper bound for unsplittable flow (Theorem 1.6) and priority flow.

The following two tables (Figures 1 and 2) consolidate the known inapproximability results when the NBA is not enforced.

	Undirected	Directed	Planar (Directed or Undirected)
MAX UFP	$\Omega(m^{1-\varepsilon})$ this paper	$\Omega(m^{1-\varepsilon})$ Azar-Regev	$\Omega(m^{1-\varepsilon})$ this paper
MAX CONFLUENT	$\Omega(m^{1-\varepsilon})$ this paper	$\Omega(m^{1-\varepsilon})$ this paper	NP-Hard by subset sum

Figure 1: Inapproximability results without NBA and with exponential demands.

	Directed or Undirected	Planar (Directed or Undirected)
MAX UFP/ Max Priority	$\Omega(m^{1/2-\varepsilon})$ this paper	$\Omega(m^{1/2-\varepsilon})$: this paper
MAX CONFLUENT	$\Omega(m^{1/2-\varepsilon})$ this paper	NP-hard by subset sum

Figure 2: Inapproximability results without NBA and with polynomial demands.

2 The Two-Disjoint Paths problem

Our hardness reductions require gadgets based upon the *capacitated* 2-disjoint paths problem. Before describing this problem, recall the classical 2-disjoint paths problem:

`2-Disjoint Paths (Uncapacitated)`: Given a graph G and node pairs $\{x_1, y_1\}$ and $\{x_2, y_2\}$. Does G contain paths P_1 from x_1 to y_1 and P_2 from x_2 to y_2 such that P_1 and P_2 are disjoint?

Observe that this formulation incorporates four distinct problems because the graph G may be directed or undirected and the desired paths may be edge-disjoint or node-disjoint. In undirected graphs the 2-disjoint paths problem, for both edge-disjoint and node disjoint paths, can be solved in polynomial time—see Robertson and Seymour [29]. In directed graphs, perhaps surprisingly, the problem is NP-hard. This is the case for both edge-disjoint and node disjoint paths, as shown by Fortune, Hopcroft and Wyllie [20].

In general, the unsplittable and confluent flow problems concern capacitated graphs. Therefore, our focus is on the capacitated version of the 2-disjoint paths problem.

`2-Disjoint Paths (Capacitated)`: We are given a graph G and two parameters $\beta \geq \alpha$. The edges of G are assigned capacities $\in \{\alpha, \beta\}$. Given node pairs $\{x_1, y_1\}$ and $\{x_2, y_2\}$, does G contain paths P_1 from x_1 to y_1 and P_2 from x_2 to y_2 such that:

(i) P_1 and P_2 are disjoint.

(ii) P_2 may only use edges of capacity β. (P_1 may use both capacity α and capacity β edges.)

For directed graphs, the result of Fortune et al. [20] immediately implies that the capacitated version is hard—simply assume every edge has capacity β. In undirected graphs, the case of node-disjoint paths was proven to be hard by Guruswami et al. [21]. The case of edge-disjoint paths was recently proven to be hard by Naves, Sonnerat and Vetta [27], even in planar graphs where terminals lie on the outside face (in an interleaved order, which will be important for us). These results are summarised in Table 1.

	Directed	Undirected
Node-Disjoint	NP-hard [20]	NP-hard [21]
Edge-Disjoint	NP-hard [20]	NP-hard [27]

Table 1: Hardness of the Capacitated 2-Disjoint Paths Problem.

Recall that the unsplittable flow problem has capacities on edges, whereas the confluent flow problem has capacities on nodes. Consequently, our hardness reductions for unsplittable flows require gadgets based upon the hardness for edge-disjoint paths [27]; for confluent flows we require gadgets based upon the hardness for node-disjoint paths [21].

3 Polynomial hardness of single-sink unsplittable, confluent and priority flow

In this section, we establish that the single-sink maximum unsplittable and confluent flow problems are hard to approximate within polynomial factors for both the cardinality and throughput objectives. We will then show how these hardness results extend to the single-sink maximum priority flow problem. We begin with the small-demand regime by proving Theorem 1.3. Its proof introduces some core ideas that are used in later sections in the proofs of Theorems 1.5 and 1.7.

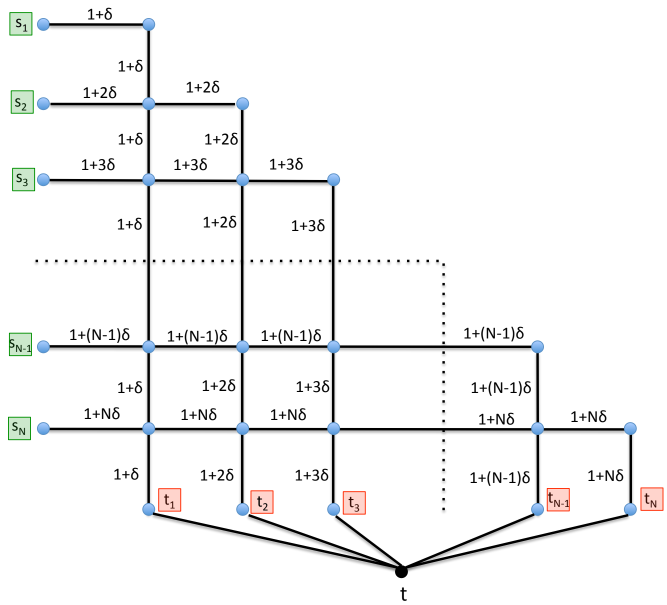

Figure 3: A Half-Grid G_N.

3.1 $\sqrt{n}$-hardness in the small-demand regime

Our approach uses a grid routing structure much as in the hardness proofs of Guruswami et al. [21]. Specifically:

(1) We introduce a graph G_N that has the following properties. There are demands with values $1+i\delta$ for $i=1,\dots,N$ and so $d_{\min}=1+\delta \le d_{\max}=1+N\delta$. There is a set of pairwise crossing paths that can route demands of total value $\sum_{i=1}^{N}(1+i\delta)=N+\delta(N/2)(N+1)$. On the other hand, any collection of pairwise non-crossing paths can route at most $d_{\max}=1+N\delta$ units of the total demand. For a given $\Delta\in(0,1)$ we choose δ to be small enough so that $d_{\max}\le 1+\Delta<2$.

(2) We then build a new network $\mathcal{G}$ by replacing each node of G_N by an instance of the capacitated 2-disjoint paths problem. This routing problem is chosen because it induces the following properties. If it is a YES-instance, then a maximum unsplittable (or confluent) flow on $\mathcal{G}$ corresponds to routing demands in G_N using pairwise-crossing paths. In contrast, if it is a NO-instance, then a maximum unsplittable or confluent flow on $\mathcal{G}$ corresponds to routing demands in G_N using pairwise non-crossing paths.

Since G_N contains $n=O(N^2)$ nodes, it follows that an approximation algorithm with guarantee better than $\Theta(\sqrt{n})$ allows us to distinguish between YES- and NO-instances of our routing problem, giving an inapproximability lower bound of $\Omega(\sqrt{n})$. Furthermore, at all stages we show how this reduction can be applied using only undirected graphs. This will prove Theorem 1.3.

3.1.1 A half-grid graph G_N

Let's begin by defining the graph G_N. There are N rows (numbered from top to bottom) and N columns (numbered from left to right). We call the leftmost node in the i^{th} row s_i, and the bottom node in the j^{th} column t_j. There is a demand of size $c_i := 1 + i\delta$ located at s_i. Recall, that δ is chosen so that all demands and capacities lie within a tight range $[1, 1+\Delta]$ for fixed Δ small. All the edges in the i^{th} row and all the edges in the i^{th} column have capacity c_i. The i^{th} row extends as far as the i^{th} column and vice versa; thus, we obtain a "half-grid" that is a weighted version of the network considered by Guruswami et al. [21]. Finally we add a sink t. There is an edge of capacity c_j between t_j to t. The complete construction is shown in Figure 3.

For the unsplittable flow problem we have edge capacities. We explain later how the node capacities are incorporated for the confluent flow problem. We also argue about the undirected and directed reductions together. For directed instances we always enforce edge directions to be downwards and to the right.

Note that there is a unique $s_i - t$ path P_i^* consisting only of edges of capacity c_i, that is, the hooked path that goes from s_i along the i^{th} row and then down the i^{th} column to t. We call this the *canonical* path for demand i.

Claim 3.1. *Take two feasible paths Q_i and Q_j for demands i and j. If $i < j$, then the paths must cross on row j, between columns i and $j-1$.*

Proof. Consider demand i originating at s_i. This demand cannot use any edge in columns 1 to $i-1$ as it is too large. Consequently, any feasible path Q_i for demand i must include all of row i. Similarly, Q_j must contain all of row j. Row j cuts off s_i from the sink t, so Q_i must meet Q_j on row j. Demand i cannot use an edge in row j as demand j is already using up all the capacity along that row. Thus Q_i crosses Q_j at the point they meet. As above, this meeting cannot occur in columns 1 to $i-1$. Thus the crossing point must occur on some column between i and $j-1$ (by construction of the half-grid, column j only goes as high as row j so the crossing cannot be there). □

By Claim 3.1, if we are forced to route using pairwise non-crossing paths, then only one demand can route. Thus we can route at most a total of $c_N = 1 + \delta N < 2$ units of demand.

3.1.2 The instance $\mathcal{G}$

We build a new instance $\mathcal{G}$ by replacing each degree 4 node in G_N with an instance of the 2-disjoint paths problem. For the unsplittable flow problem in undirected graphs we use gadgets H corresponding to the capacitated edge-disjoint paths problem. Observe that a node at the intersection of column i and row j (with $j > i$) in G_N is incident to two edges of capacity c_i and to two edges of weight c_j. We construct $\mathcal{G}$ by replacing each such node of degree four with the routing graph H. We do this in such a way that the capacity c_i edges of G_N are incident to x_1 and y_1, and the c_j edges are incident to x_2 and y_2. We also let $\alpha = c_i$ and $\beta = c_j$.

For the confluent flow problem in undirected graphs we now have node capacities. Hence we use gadgets H corresponding to the node-capacitated 2-paths problem discussed above.

For directed graphs, the mechanism is simpler as the gadgets may now come from the uncapacitated disjoint paths problem. Thus the hardness comes from the directedness and not from the capacities.

Specifically, we may set the edge capacities to be $C = \max\{c_i, c_j\}$. Moreover, for unsplittable flow we may perform the standard operation of splitting each node in H into two, with the new induced arc having capacity of C. It follows that if there are two flow paths through H, each carrying at least $c_i \geq c_j$ flow, then they must be from x_1 to y_1 and x_2 to y_2. These provide a solution to the node-disjoint directed paths problem in H.

The hardness result will follow once we see how this construction relates to crossing and non-crossing collections of paths.

Lemma 3.2. *If H is a YES-instance, then the maximum unsplittable/confluent flow in $\mathcal{G}$ has value at least N. For a NO-instance the maximum unsplittable/confluent flow has value at most $1+\Delta < 2$.*

Proof. If H is a YES-instance, then we can use its paths to produce paths in $\mathcal{G}$, whose images in G_N, are free to cross at any node. Hence we can produce paths in $\mathcal{G}$ whose images are the canonical paths P_i^*, $1 \leq i \leq N$ in G_N. This results in a flow of value greater than N. Note that in the confluent case, these paths yield a confluent flow as they only meet at the root t.

Now suppose H is a NO-instance. Take any flow and consider two paths $\hat{Q}_i$ and $\hat{Q}_j$ in $\mathcal{G}$ for demands i and j, where $i < j$. These paths also induce two feasible paths Q_i and Q_j in the half-grid G_N. By Claim 3.1, these paths cross on row j of the half-grid (between columns i and $j-1$). In the directed case (for unsplittable or confluent flow) if they cross at a grid-node v, then the paths they induce in the copy of H at v must be node-disjoint. This is not possible in the directed case since such paths do not exist for (x_1, y_1) and (x_2, y_2).

In the undirected confluent case, we must also have node-disjoint paths through this copy of H. As we are in row j and a column between column i and $j-1$, we have $\beta = c_j$ and $c_i \leq \alpha \leq c_{j-1}$. Thus, demand j can only use the β-edges of H. This contradicts the fact that H is a NO-instance. For the undirected case of unsplittable flow the two paths through H need to be edge-disjoint, but now we obtain a contradiction as our gadget was derived from the capacitated edge-disjoint paths problem.

It follows that no such pair $\hat{Q}_i$ and $\hat{Q}_j$ can exist and, therefore, the confluent/unsplittable flow routes at most one demand and, hence, routes a total demand of at most $1+\Delta$. □

We then obtain our hardness result.

Theorem 1.3. *Neither cardinality nor throughput can be approximated to within a factor of $O(m^{1/2-\varepsilon})$, for any $\varepsilon > 0$, in the single-sink unsplittable and confluent flow problems. This holds for undirected and directed graphs even when instances are restricted to have bottleneck value $d_{\max}/u_{\min} = 1+\Delta$ where $\Delta > 0$ is arbitrarily small.*

Proof. It follows that if we could approximate the maximum (unsplittable) confluent flow problem in $\mathcal{G}$ to a factor better than N, we could determine whether the optimal solution is at least N or at most $1+\Delta$. This in turn would allow us to determine whether H is a YES- or a NO-instance.

Note that $\mathcal{G}$ has $n = \Theta(pN^2)$ edges, where $p = |V(H)|$. If we take $N = \Theta(p^{(1/2)(1/\varepsilon-1)})$, where $\varepsilon > 0$ is an (arbitrarily) small constant, then $n = p^{1/\varepsilon}$ and so $N = \Theta(n^{(1/2)(1-\varepsilon)})$. A precise lower bound of $n^{1/2-\varepsilon'}$ is obtained for $\varepsilon' > \varepsilon$ sufficiently small, when n is sufficiently large. □

3.1.3 Priority flows and congestion

We now establish the hardness of priority flows. To do this, we use the same half-grid construction, except we must replace the capacities by priorities. This is achieved in a straight-forward manner. Priorities are defined by the magnitude of the original demands/capacities. The larger the demand or capacity in the original instance, the higher its priority in the new instance. (Given the priority ordering we may then assume all demands and capacities are set to 1.) In this setting, observe that Claim 3.1 also applies for priority flows.

Claim 3.3. *Consider two feasible paths Q_i and Q_j for demands i and j in the priority flow problem. If $i < j$, then the paths must cross on row j, between columns i and $j-1$.*

Proof. Consider demand i originating at s_i. This demand cannot use any edge in columns 1 to $i-1$ as they do not have high enough priority. Consequently, any feasible path Q_i for demand i must include all unit capacity edges of row i. Similarly, Q_j must contain all of row j. Row j cuts off s_i from the sink t, so Q_i must meet Q_j on row j. Demand i cannot use an edge in row j as demand j is already using up all the capacity along that row. Thus Q_i crosses Q_j at the point they meet. As above, this meeting cannot occur in columns 1 to $i-1$. Thus the crossing point must occur on some column between i and $j-1$. □

Repeating our previous arguments, we obtain the following hardness result for priority flows. (Again, it applies to both throughput and cardinality objectives as they coincide for priority flows.)

Corollary 1.4. *The maximum single-sink priority flow problem cannot be approximated to within a factor of $m^{1/2-\varepsilon}$, for any $\varepsilon > 0$, in planar directed or undirected graphs.*

We close the section by establishing Theorem 1.8. Consider a grid instance built from a YES instance of the 2 disjoint path problem. As before we may find a routing of all demands with congestion at most 1. Otherwise, suppose that the grid graph is built from a NO instance and consider a tree T returned by a strong congestion algorithm. As it is a strong algorithm, the demand in row i must follow its canonical path horizontally to the right as far as it can. As it is a confluent flow, all demands from rows $> i$ must accumulate at this rightmost node in row i. Inductively this implies that the total load at the rightmost node in row 1 has load $> N$. As before, for any $\varepsilon > 0$ we may choose N sufficiently large so that $N \geq n^{1/2-\varepsilon}$. Hence we have a YES instance of 2 disjoint paths if and only if the output from a $n^{1/2-\varepsilon}$-approximate strong congestion algorithm returns a solution with congestion $\leq N$.

4 Logarithmic hardness of single-sink confluent flow *with* the no-bottleneck assumption

We now prove the logarithmic hardness of the confluent flow problem given the no-bottleneck assumption. A similar two-step plan is used as for Theorem 1.3 but the analysis is more involved.

Step (1). We introduce a planar graph G_N which has the same structure as our previous half-grid, except its capacities are changed. As before, we have demands associated with the s_i's, but we assume these demands are tiny—this guarantees that the no-bottleneck assumption holds. We thus refer to the demands located at an s_i as the *packets* from s_i. We define G_N to ensure that there is a collection of pairwise crossing paths that can route packets of total value equal to the harmonic number $\eta_N \approx \log N$.

Step (2). We then build a new network $\mathcal{G}$ by replacing each node of G_N by an instance of the 2-disjoint paths problem. Again, this routing problem is chosen because it induces the following properties. If it is a YES-instance, then we can find a routing that corresponds to pairwise crossing paths with the correct capacities. Hence, we are able to route η_N demand. In contrast, if it is a NO-instance, the capacity constraints within the gadgets become restrictive and force the maximum confluent flow value to be at most 2.

It follows that an approximation algorithm with guarantee better than logarithmic would allow us to distinguish between YES- and NO-instances of our routing problem, giving a lower bound of $\Omega(\log N)$. We will see that this bound is equal to $\Theta(\log^{1-\varepsilon} n)$.

4.1 An updated half-grid graph

Again we take the graph G_N with N rows (now numbered from bottom to top) and N columns (now numbered from right to left). As we are considering the confluent flow problem, we sub-divide the edges in order to place capacities on the nodes. The nodes in the i^{th} row and the nodes in the i^{th} column have capacity $1/i$. The i^{th} row extends as far as the i^{th} column and vice versa; thus, we obtain a half-grid similar to our earlier construction but with updated capacities. Then we add a sink t. There is an edge to t from the bottom node (called t_i) in column i; the capacity of t_i is $1/i$. Finally, at the leftmost node (called s_i) in row i there is a collection of packets ("subdemands") whose total weight is $1/i$. These packets are very small. In particular, they are much smaller than $1/n$, so they satisfy the no-bottleneck assumption. The construction is shown in Figure 4. In the directed setting, edges are oriented to the right and downwards.

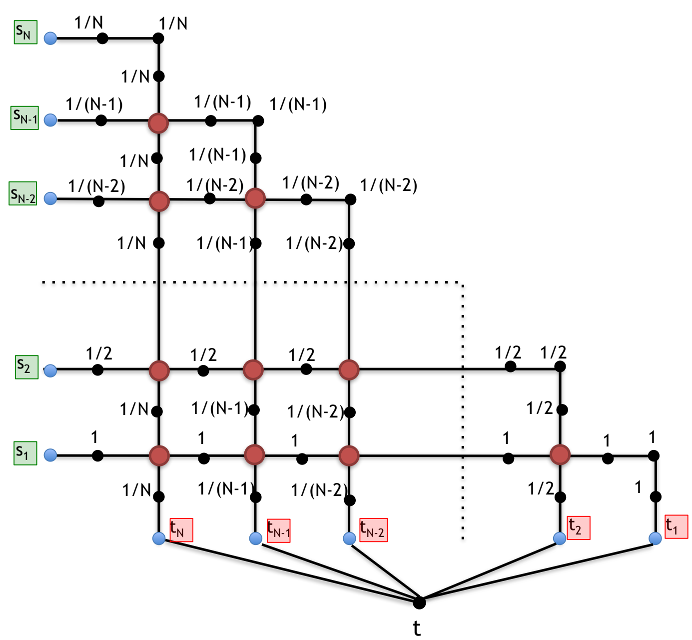

Figure 4: An Updated NBA Half-Grid G_N.

Again, there is a unique s_i-t path P_i^* consisting only of nodes of weight $1/i$, that is, the hooked path that goes from s_i along the ith row and then down the i^{th} column to t. Moreover, for $i \neq j$, the path P_i^* intersects P_j^* precisely once. If we route packets along the paths $\mathcal{P}^* = \{P_1^*, P_2^*, \dots, P_N^*\}$, then we obtain a flow of total value $\eta_N = 1 + (1/2) + \cdots + (1/N)$. Since every node incident to t is used in $\mathcal{P}^*$ with its maximum capacity, this solution is a maximum single-sink flow.

For the hardness construction, we again build a new instance $\mathcal{G}$ by replacing each degree 4 node in G_N with an instance of the 2-disjoint paths problem. For this confluent flow problem, our gadgets correspond to the capacitated node-disjoint paths problem.[4] We replace the node at the intersection of row i and column j (with $i < j$) in G_N by a copy of the routing graph H—see Section 2. For clarity, we denote the gadget H located at this intersection by $H_{i,j}$. The nodes x_1 and y_1 are placed on row i and have (high) capacity $c_i = 1/i$. The nodes x_2 and y_2 are placed on column j and have (low) capacity $c_j = 1/j$. We also let $\alpha = c_i$ and $\beta = c_j$.

If H is a YES instance, then we can find a high capacity path, Q_1, from x_1 to y_1 and a low capacity path (that is, a path that may use both high and low capacity edges), Q_2, from x_2 to y_2 that is node disjoint from Q_1. Clearly, combining the hooked paths $\mathcal{P}^* = \{P_1^*, P_2^*, \dots, P_N^*\}$ with the paths Q_1 and Q_2 within each gadget gives a confluent flow of value η_N.

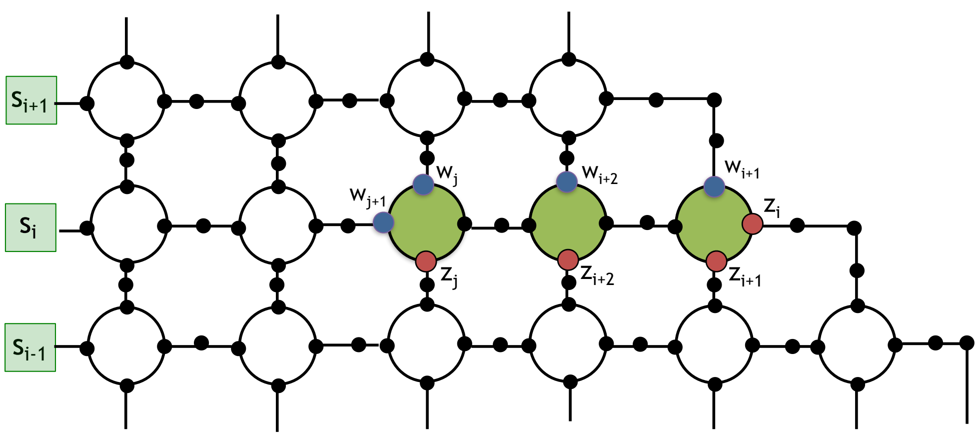

Figure 5: The Interval $\mathcal{I}[i,j]$ of Gadgets.

Now, let's examine what happens if H is a NO instance. We require the following terminology. Let $\mathcal{I}[i,j] = \{H_{i,j}, H_{i,j-1}, \dots, H_{i,i+2}, H_{i,i+1}\}$ be an *interval of gadgets* in row i. We say that $\{z_j, z_{j-1}, \dots, z_{i+1}, z_i\}$ are the *exit* nodes for this interval of gadgets, where z_i is the "y_1 node" in gadget $H_{i,i+1}$ and z_k, for $j \geq k \geq i+1$ is the y_2 node in gadget $H_{i,k}$. We say that $\{w_{j+1}, w_j, \dots, w_{i+1}\}$ are the *entry* nodes for this interval of gadgets, where w_{j+1} is the x_1 node in gadget $H_{i,j}$ and w_k, for $j \geq k \geq i+1$, is the x_2 node in gadget $H_{i,k}$. (We remark that, in the undirected case, flow may actually enter a gadget

[4]In the directed case, we could simply adopt the uncapacitated version but its easier to keep the capacitated version in mind for both cases.

through an “exit" node and leave via an “entry" node.) This is illustrated in Figure 5 where the entry nodes to $\mathcal{I}[i,j]$ are shown in blue and the exit nodes in red.

The next lemma bounds the amount of confluent flow which can be sent from $\{s_i, s_{i+1}, \dots, s_N\}$ through $\mathcal{I}[i,j]$ (via its entry nodes) and on to its exit nodes. Formally, we refer to an $\mathcal{I}[i,j]$ *confluent flow* as any confluent flow where

(i) the source of any flow path is in $\{s_i, s_{i+1}, \dots, s_N\}$,

(ii) the terminal node of any flow path is an exit node of $\mathcal{I}[i,j]$ and each flow path contains precisely one such exit node,[5] and

(iii) the edge entering the exit node of any flow path is inside one of the gadgets from $\mathcal{I}[i,j]$.

Lemma 4.1. *Let H be a NO instance and let $\mathcal{I}[i,j] = \{H_{i,j}, H_{i,j-1}, \dots, H_{i,i+1}\}$ be an interval of gadgets for a pair i,j, where $i < j \le N$. Then the value of any $\mathcal{I}[i,j]$ confluent flow is at most $2 \cdot (j-i+1)/j$.*

Before proceeding to the proof, note that any confluent flow in the whole instance corresponds to an $\mathcal{I}[1,n]$ confluent flow. This is because any flow from $\{s_1, s_2, \dots, s_N\}$ to t must reach the exit nodes of $\mathcal{I}[1,n]$ from within $\mathcal{I}[1,n]$ and not from below. Hence the following key property is immediate.

Corollary 4.2. *If H is a NO instance, then the maximum value of a confluent flow in G_N is at most 2.*

Proof. We now prove Lemma 4.1 by induction on decreasing values of i, and then increasing values of j. For the base case consider $i = N-1$. Now j can take only the value N. The interval $\mathcal{I}[N-1,N]$ consists of a single gadget, namely $H_{n-1,n}$ and so all flow from s_N and s_{N-1} must pass through this gadget. We now show that the maximum amount of flow that can be sent confluently from s_N and s_{N-1} to the exits $\{z_{N-1}, z_N\}$ is $(j-i+1/j) = (N-(N-1)+1)(N) = (2/N)$. Clearly this is less than $2 \cdot (j-i+1/j) = (4/N)$. We have four cases, as illustrated in Figure 6.

(i) The flows through x_1 and x_2 meet and merge and exit through $y_1 = z_{N-1}$. As y_1 has capacity $1/(N-1)$, the total flow is at most $1/(N-1) < (2/N)$.

(ii) The flows through x_1 and x_2 meet and merge and then exit through $y_2 = z_N$. As y_2 has capacity $1/N$, the total flow is at most $1/N < 2/N$.

(iii) Using two node-disjoint paths, the flow through x_1 exits at $y_2 = z_N$ and the flow through x_2 exits at $y_1 = z_{N-1}$. Since the capacities into x_2 and out of y_2 are $1/N$, the total flow is at most $2/N$.

(iv) Using two node-disjoint paths, the flow through x_1 exits at $y_1 = z_{N-1}$ and the flow through x_2 exits at $y_2 = z_N$. But, H is a NO instance. Therefore, the path used from x_1 to y_1 must use at least one low capacity node. Thus, again, the flow value is at most $2/N$, as required.

We proceed by induction. Take a pair i,j with $i < j \le N$. If $j = i+1$ then $\mathcal{I}[i,j]$ consists of a single gadget and the argument used in the base case can be successfully applied. So assume that $j > i+1$. Now take any confluent flow from $\{s_i, s_{i+1}, \dots, s_N\}$ through $\mathcal{I}[i,j]$ (via its entry nodes) to its exit nodes.

[5] Note however that a flow path may actually contain multiple entry nodes of $\mathcal{I}[i,j]$.

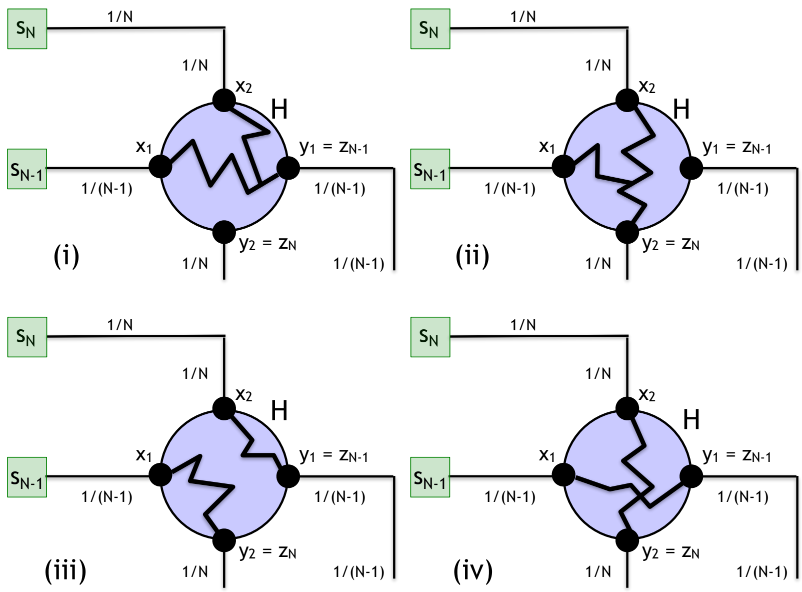

Figure 6: Base Case.

Observe that this flow can reach the entry nodes of $\mathcal{I}[i,j]$ in two ways. Either it reaches the entry node w_{j+1} via the horizontal edge e in row i or it reaches the entry nodes $\{w_j, w_{j-1}, \ldots, w_{i+1}\}$ from above via the entry (and exit) nodes of the interval $\mathcal{I}[i+1,j]$. See Figure 7. (Recall we are not interested in flow that reaches the exit nodes of $\mathcal{I}[i,j]$ from below, as this flow does not pass through $\mathcal{I}[i,j]$.)

Now suppose no flow is sent to exit node z_j. Since, $j > i+1$, by induction on the case $\{i, j-1\}$, the total flow sent is at most

$$2\cdot((j-1)-i+1)/(j-1) = 2\cdot(j-i)/(j-1) \le 2\cdot(j-i+1)/j$$

as required. So we now assume that some flow is sent to exit node z_j.

We consider two types of flow: (Type 1) flow that does not enter at w_{j+1} and (Type 2) flow that does enter at w_{j+1}. Let us first consider the Type 1 flow. Some of this flow may originate from $\{s_{i+1}, s_{i+2}, \ldots, s_N\}$ and hence forms an $\mathcal{I}[i+1,j]$ confluent flow. By induction, the value of this type of flow is at most $2\cdot(j-(i+1)+1)/j = 2\cdot(j-i)/j$. The rest of the Type 1 flow originates at s_i. Because the flow is confluent these packets route along a single path. Since this flow path does not enter w_{j+1}, it must go via an entry node of $\mathcal{I}[i+1,j]$. But to reach $\mathcal{I}[i+1,j]$, without touching any exit nodes of $\mathcal{I}[i,j]$, this path must traverse up some column ℓ where $N \ge \ell > j$. The capacity of the nodes in this column are $1/\ell < 1/j$. Consequently this class of Type 1 flow contributes flow value at most $1/j$.

So let us now consider Type 2 flow. This flow originates either at $\{s_{i+1}, s_{i+2}, \ldots, s_N\}$ or $\{s_i\}$ and travels through the horizontal edge e in row i into gadget $H_{i,j}$ at node w_{j+1}. Suppose first that this flow either goes down through exit z_j or up through entry w_j in gadget $H_{i,j}$. Then this flow contributes value at most $1/j$, because both these nodes have capacity $1/j$. Otherwise this flow exits gadget $H_{i,j}$ on the right via its y_1 node. Recall, by assumption, some other flow does reach z_j. By confluence, this flow cannot

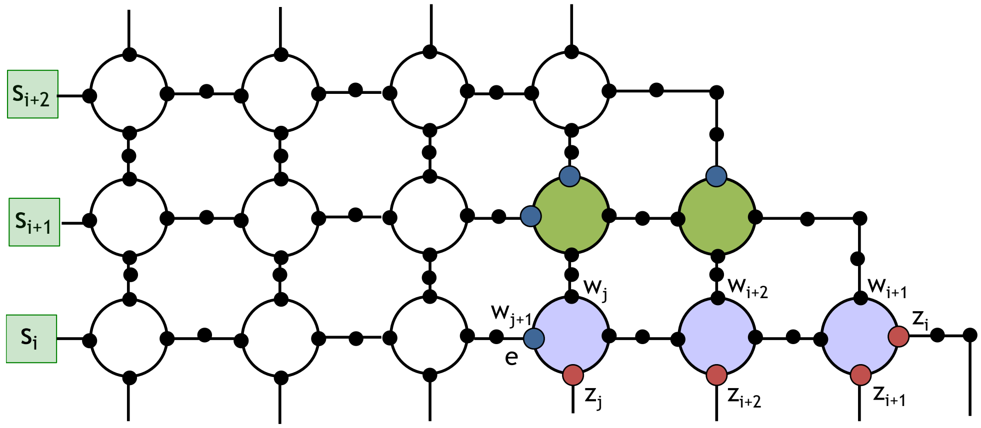

Figure 7: Induction Step.

have entered the gadget via w_{j+1} or the y_1 node. Hence it must come via the $x_2 = w_j$ node. Thus we have disjoint paths from x_1 to y_1 and from x_2 to y_2. Because H is a NO instance, the path from x_1 to y_1, used by the Type 2 flow, must contain an edge of low capacity. Thus the Type 2 flow path again contributes at most value $1/j$. So total flow is at most

$$2\cdot(j-i)/j+(1/j)+(1/j)=2\cdot(j-i+1)/j$$

as desired. The result follows by induction. $\square$

We can now establish the claimed inapproximability bound.

Theorem 1.7. *Given the no-bottleneck assumption, the single-sink confluent flow problem cannot be approximated to within a factor $O(\log^{1-\varepsilon} n)$, for any $\varepsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$. This holds for both the maximum cardinality and maximum throughput objectives in undirected and directed graphs.*

Proof. It follows that if we could approximate the maximum confluent flow problem in $\mathcal{G}$ to a factor better than $\eta_N/2$, we could determine whether the optimal solution is 2 or η_N. This in turn would allow us to determine whether H is a YES- or a NO-instance.

Note that $\mathcal{G}$ has $n = \Theta(pN^2)$ edges, where $p = |V(H)|$. If we take $N = \Theta(p^{(1/2)((1/\varepsilon)-1)})$, where $\varepsilon > 0$ is a small constant, then $\eta_N = \Theta((1/2)((1/\varepsilon)-1)\log p)$. For p sufficiently large, this is $\Omega((\log n)^{1-\varepsilon}) = ((1/\varepsilon)\log p)^{1-\varepsilon}$. This gives a lower bound of $\Omega((\log n)^{1-\varepsilon})$. $\square$

5 Stronger lower bounds for cardinality single-sink unsplittable flow with arbitrary demands

In the large-demand regime even stronger lower bounds can be achieved for the cardinality objective. To see this, we explain the technique of Azar and Regev [3] (used to prove Theorem 1.1) in Section 5.1

and show how to extend it to undirected graphs and to confluent flows. Then in Section 5.2, we combine their construction with the half-grid graph to obtain lower bounds in terms of the bottleneck value (Theorem 1.5).

5.1 $m^{1-\varepsilon}$ hardness in the large-demand regime

Theorem 1.2. *If* $\mathsf{P} \neq \mathsf{NP}$ *then, for any* $\varepsilon > 0$*, there is no* $O(m^{1-\varepsilon})$*-approximation algorithm for the cardinality objective of the single-sink unsplittable/confluent flow problem in undirected graphs.*

Proof. We begin by describing the construction of Azar and Regev for directed graphs. They embed instances of the uncapacitated 2-disjoint paths problem into a directed path. Formally, we start with a directed path $z^1, z^2, \dots, z^\ell$ where $t = z^\ell$ forms our sink destination for all demands. In addition, for each $i < \ell$, there are two parallel edges from z^{i-1} to z^i. One of these has capacity 2^i and the other has a smaller capacity of $2^i - 1$. There is a demand s_i from each z^i, $i < \ell$ to z^ℓ of size 2^{i+1}. Note that this unsplittable flow instance is feasible as follows. For each demand s_j, we may follow the high capacity edge from z^j to z^{j+1} (using up all of its capacity) and then use low capacity edges on the path $z^{j+1}, z^{j+2}, \dots, z^\ell$. Call these the *canonical paths* for the demands. The total demand on the low capacity edge from z^j is then $\sum_{i \le j} 2^i = 2^{j+1} - 1$, as desired.

Now replace each node z^j, $1 \le j < \ell$, by an instance H^j of the uncapacitated directed 2-disjoint paths problem. Each edge in H^j is given capacity 2^{j+1}. Furthermore we have the following.

(i) The tail of the high capacity edge out of z^j is identified with the node y_2.

(ii) The tail of the low capacity edge out of z^j is identified with y_1.

(iii) The heads of both edges into z^j (if they exist) are identified with x_1.

(iv) The node x_2 becomes the starting point of the demand s_j from z^j.

This construction is shown in Figure 8.

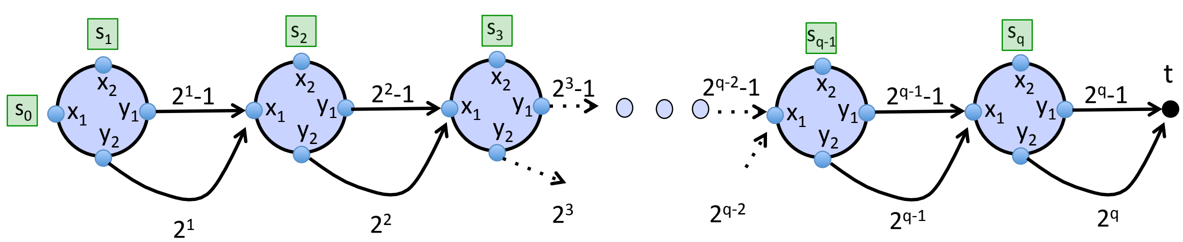

Figure 8: An Azar-Regev Path.

Now if we have a YES-instance of the 2-disjoint paths problem, we may then simulate the canonical paths in the standard way. The demand in H^j uses the directed path from x_2 to y_2 in H^j; it then follows the high capacity edge from y_2 to the x_1-node in the next instance H^{j+1}. All the total demand arriving from upstream H^i's entered H^j at its node x_1 and follows the directed path from x_1 to y_1. This total demand is at most $\sum_{i\le j} 2^i$ and thus fits into the low capacity edge from H^j into H^{j+1}. Observe that this routing is also confluent in our modified instance because the paths in the H^j's are node-disjoint. Hence, if we have a YES-instance of the 2-disjoint paths problem, both the unsplittable and confluent flow problems have a solution routing all of the demands.

Now suppose that we have a NO-instance, and consider a solution to the unsplittable (or confluent) flow problem. Take the highest value i such that the demand from H^i is routed. By construction, this demand must use a path P_2 from x_2 to y_2. But this saturates the high capacity edge from y_2. Hence any demand from H^j, $j < i$ must pass from y_1 to x_1 while avoiding the edges of P_2. This is impossible, and so we route at most one demand.

This gives a gap of ℓ for the cardinality objective. Azar-Regev then choose $\ell = |V(H)|^{\lceil 1/\varepsilon \rceil}$ to obtain a hardness of $\Omega(n^{1-\varepsilon'})$.

Now consider undirected graphs. Here we use an undirected instance of the capacitated 2-disjoint paths problem. We plug this instance into each H^j, and use the two capacity values of $\beta = 2^{j+1}$ and $\alpha = 2^{j+1} - 1$. A similar routing argument then gives the lower bound. □

We remark that it is easy to see why this approach does not succeed for the throughput objective. The use of exponentially growing demands implies that high throughput is achieved simply by routing the largest demand.

5.2 Lower bounds for arbitrary demands

By combining paths and half-grids we are able to refine the lower bounds in terms of the bottleneck value (or demand spread).

Theorem 1.5. *Consider any fixed $\varepsilon > 0$ and $d_{\max}/u_{\min} > 1$. It is* NP*-hard to approximate cardinality single-sink unsplittable/confluent flow to within a factor of $O(\sqrt{\log(d_{\max}/u_{\min})} \cdot m^{1/2-\varepsilon})$ in undirected or directed graphs. For unsplittable flow, this remains true for planar graphs.*

Proof. We start with two parameters p and q. We create p copies of the Azar-Regev path and attach them to a $p \times p$ half-grid, as shown in Figure 9.

Now take the i^{th} Azar-Regev path, for each $i = 1, 2 \ldots p$. The path contains q supply nodes with demands of sizes $2^{(i-1)q}, 2^{(i-1)q+1}, \ldots, 2^{iq-1}$. (Supply node s_j has demand 2^{j-1}.) Therefore the total demand on path i is $\tau_i := 2^{(i-1)q}(2^q - 1) < 2^{iq}$. The key point is that *the total demand of path i is less than the smallest demand in path $i+1$*. Note that we have pq demands, and thus demand sizes from 2^0 up to 2^{pq-1}. Consequently the demand spread is 2^{pq-1}. We set $u_{\min} = d_{\min}$ and thus

$$pq - 1 = \log(d_{\max}/d_{\min}) = \log(d_{\max}/u_{\min}) .$$

It remains to prescribe capacities to the edges of the half-grid. To do this every edge in ith canonical hooked path has capacity τ_i (not c_i). These capacity assignments, in turn, induce corresponding capacities in each of the disjoint paths gadgets. It follows that if each gadget on the paths and half-grid correspond to a YES-instance gadget then we may route all pq demands.

Now suppose the gadgets correspond to a NO-instance. It follows that we may route at most one demand along each Azar-Regev path. But, by our choice of demand values, any demand on the ith path

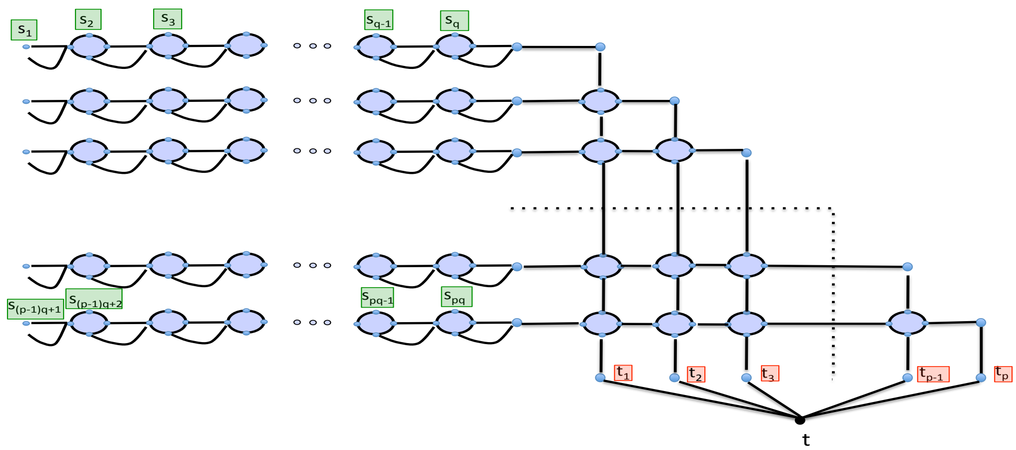

Figure 9:

is too large to fit into any column $j < i$ in the half-grid. Hence we have the same conditions required in Theorem 1.3 to show that at most one demand in total can feasibly route. It follows that we cannot approximate the cardinality objective to better than a factor pq.

Note that the construction contains at most $m = O((qp+p^2)\cdot|E(H)|)$ edges, where H is the size of the 2-disjoint paths instance. Now we select p and q such that $q \geq p$ and $pq \geq |E(H)|^{1/\varepsilon}$. Then, for some constant C, we have

$$\begin{aligned} C\cdot m^{1/2-\varepsilon}\cdot\sqrt{\log(d_{\max}/d_{\min})} &= m^{1/2-\varepsilon}\cdot\sqrt{\log(d_{\max}/d_{\min})} \\ &\leq \sqrt{pq}\cdot\sqrt{pq} \\ &= pq\,. \end{aligned}$$

Therefore, since we cannot approximate to within pq, we cannot approximate the cardinality objective to better than a factor $O(\sqrt{\log(d_{\max}/u_{\min})}\cdot m^{1/2-\varepsilon})$. □

6 Upper bounds for flows with arbitrary demands

In this section we present upper bounds for maximum flow problems with arbitrary demands.

6.1 Unsplittable flow with arbitrary demands

One natural approach for the cardinality unsplittable flow problem is used repeatedly in the literature (even for general multiflows) and we appeal to it in the proof of Theorem 1.6 and defer the details for now. Group the demands into at most $O(\log d_{\max}/d_{\min})$ bins, and then consider each bin separately. This approach can also be applied to the throughput objective or the more general profit-maximisation model. This immediately incurs, however, a loss factor relating to the number of bins, and this feels slightly artificial. In fact, given the NBA regime, there is no need to lose this extra factor; Baveja et al. [4] gave an $O(\sqrt{m})$ approximation for profit-maximisation when $d_{\max} \le u_{\min}$. On the other hand, our lower bound in Theorem 1.5 shows that if $d_{\max} > u_{\min}$ we do need to lose some factor dependent on $d_{\max}$. But how large does this need to be? The current best upper bound is $O(\log(d_{\max}/u_{\min}) \cdot \sqrt{m \log m})$ by Guruswami et al. [21], and this works for the general profit-maximisation model.[6] For the cardinality and throughput objectives, however, we can obtain a better upper bound. The proof combines analyses from [4] and [25] (which focus on the NBA case). We emphasize that the following theorem applies to all multiflow problems not just the single-sink case.

We also point out a related analysis of the greedy algorithm for unsplittable flows (without NBA) given by Kolman [26]. In the cardinality maximization version he gives a $O(\min\{n^{2/3}, \sqrt{m}\})$-approximation which is superior to our bound below. In the directed setting, he obtains a $O(\min\{n^{4/5}, \sqrt{m}\})$-approximation for cardinality maximization (Theorem 6 of [26]).

Theorem 1.6. *There is an $O(\sqrt{m}\log(d_{\max}/u_{\min}))$ approximation algorithm for cardinality unsplittable flow and an $O(\sqrt{m}\log n)$ approximation algorithm for throughput unsplittable flow, in both directed and undirected graphs. There is an $O(\log(d_{\max}/u_{\min})\sqrt{m \log m})$ approximation for general weights.*

Proof. We apply a result from [21] which shows that for cardinality unsplittable flow, with $d_{\max} \le \Delta d_{\min}$, the greedy algorithm yields a $O(\Delta\sqrt{m})$ approximation. Their proof is a technical extension of the greedy analysis of Kolliopoulos and Stein [25]. We first find an approximation for the sub-instance consisting of the demands at most $u_{\min}$. This satisfies the NBA and an $O(\sqrt{m})$-approximation is known for general profits [4]. More precisely, we define *bin i* to be the demands whose value lies in the range $(2^i u_{\min}, 2^{i+1} u_{\min}]$, but for bin 0 we also include demands of value exactly $u_{\min}$. Thus we need at most $\log(d_{\max}/u_{\min}) \le \log(d_{\max}/d_{\min})$ bins. The greedy algorithm above then gives the desired guarantee for the cardinality problem. The same approach applies for the throughput objective, since all demands within the same bin have values within a constant factor of each other. Moreover, we require only $\log n$ bins as demands of at most $d_{\max}/n$ may be discarded as they are not necessary for obtaining high throughput. □

As alluded to earlier, this upper bound is not completely satisfactory as pointed out in [8]. Namely, all of the lower bound instances have a linear number of edges $m = O(n)$. Therefore, it is possible that there exist upper bounds dependent on $\sqrt{n}$. Indeed, for the special case of MEDP in undirected graphs and directed acyclic graphs $O(\sqrt{n})$-approximations have been developed [9, 28].

6.2 Priority flow with arbitrary demands

Next we show that the lower bound for the maximum priority flow problem is tight.

Theorem 6.1. *Consider an instance of the maximum priority flow problem with k priority classes. There is a polytime algorithm that approximates the maximum flow to within a factor of $O(\min\{k, \sqrt{m}\})$.*

[6] Actually, they state the bound as $\log^{3/2} m$ because exponential-size demands are not considered in that paper.

Proof. First suppose that $k \leq \sqrt{m}$. Then for each class i, we may find the optimal priority flow by solving a maximum flow problem in the subgraph induced by all edges of priority i or better. This yields a k-approximation. Next consider the case where $\sqrt{m} < k$. Then we may apply Lemma 6.2, below, which implies that the greedy algorithm yields a $O(\sqrt{m})$-approximation. The theorem follows. □

The following proof for uncapacitated networks follows ideas from the greedy analysis of Kleinberg [23], and Kolliopoulos and Stein [25]. One may also design an $O(\sqrt{m})$-approximation for general edge capacities using more intricate ideas from [21]; we omit the details.

Lemma 6.2. *A greedy algorithm yields a $O(\sqrt{m})$-approximation to the maximum priority flow problem.*

Proof. We now run the greedy algorithm as follows. On each iteration, we find the demand s_i which has a shortest feasible path in the residual graph. Let P_i be the associated path, and delete its edges. Let the greedy solution have cardinality t. Let $\mathcal{O}$ be the optimal maximum priority flow and let $\mathcal{Q}$ be those demands which are satisfied in some optimal solution but not by the greedy algorithm. We aim to give an upper bound on the size of $\mathcal{Q}$.

Let Q be a path used in the optimal solution satisfying some demand in $\mathcal{Q}$. Consider any edge e and the greedy path using it. We say that P_i *blocks* an optimal path Q if i is the least index such that P_i and Q share a common edge e. Clearly such an i exists or else we could still route on Q.

Let l_i denote the length of P_i. Let k_i denote the number of optimal paths (corresponding to demands in $\mathcal{Q}$) that are blocked by P_i. It follows that $k_i \leq l_i$. But, by the definition of the greedy algorithm, we have that each such blocked path must have length at least l_i at the time when P_i was packed. Hence it used up at least $l_i \geq k_i$ units of capacity in the optimal solution. Therefore the total capacity used by the unsatisfied demands from the optimal solution is at least $\sum_{i=1}^{t} k_i^2$. As the total capacity is at most m we obtain

$$\frac{(\sum_{i=1}^{t} k_i)^2}{t} \leq \sum_{i=1}^{t} k_i^2 \leq m \tag{6.1}$$

where the first inequality is by the Chebyshev Sum Inequality or simply convexity. Since $\sum_i k_i = |\mathcal{Q}| = |\mathcal{O}| - t$, we obtain $(|\mathcal{O}| - t)^2/t \leq m$. One may verify that if $t < |\mathcal{O}|/\sqrt{m}$ then this inequality implies $|\mathcal{O}| = O(\sqrt{m})$ and, so, routing a single demand yields the desired approximation. □

7 Conclusion

It would be interesting to improve the upper bound in Theorem 1.6 to be in terms of $\sqrt{n}$ rather than $\sqrt{m}$. This is already open in the case of general directed MEDP where the best upper bound is $\min\{\sqrt{m}, n^{2/3}\}$. Eliminating the $\sqrt{\log(m)}$ term from Theorem 1.6 in the general profit maximization, and resolving the discrepancy with Theorem 1.5 (between $\sqrt{\log(d_{\max}/u_{\min})}$ and $\log(d_{\max}/u_{\min})$) would also round out the picture.

Acknowledgments. We are grateful to the referees for their excellent reports which contained many detailed comments which greatly improved the paper. We also thank Qin Bo, Mordecai Golin, Petr Kolman and Guyslain Naves for their careful reading and helpful comments. In particular, we are indebted to Qin Bo and Mordecai Golin for detecting a flaw in the proof for the undirected case of Theorem 1.7 in an earlier version of this paper. Finally, the authors gratefully acknowledge support from the NSERC Discovery Grant Program.

References

[1] MATTHEW ANDREWS, JULIA CHUZHOY, VENKATESAN GURUSWAMI, SANJEEV KHANNA, KUNAL TALWAR, AND LISA ZHANG: Inapproximability of edge-disjoint paths and low congestion routing on undirected graphs. *Combinatorica*, 30(5):485–520, 2010. [doi:10.1007/s00493-010-2455-9] 4

[2] MATTHEW ANDREWS AND LISA ZHANG: Logarithmic hardness of the undirected edge-disjoint paths problem. *J. ACM*, 53(5):745–761, 2006. Preliminary version in STOC'05. [doi:10.1145/1183907.1183910] 4

[3] YOSSI AZAR AND ODED REGEV: Combinatorial algorithms for the unsplittable flow problem. *Algorithmica*, 44(1):49–66, 2006. Preliminary version in IPCO'01. [doi:10.1007/s00453-005-1172-z] 3, 5, 16

[4] ALOK BAVEJA AND ARAVIND SRINIVASAN: Approximation algorithms for disjoint paths and related routing and packing problems. *Math. Oper. Res.*, 25(2):255–280, 2000. [doi:10.1287/moor.25.2.255.12228] 20

[5] PAUL S. BONSMA, JENS SCHULZ, AND ANDREA WIESE: A constant-factor approximation algorithm for unsplittable flow on paths. *SIAM J. Computing*, 43(2):767–799, 2014. Preliminary version in FOCS'11. [doi:10.1137/120868360, arXiv:1102.3643] 3

[6] MOSES CHARIKAR, JOSEPH S. NAOR, AND BARUCH SCHIEBER: Resource optimization in QoS multicast routing of real-time multimedia. *IEEE/ACM Transactions on Networking*, 12(2):340–348, 2004. Preliminary version in INFOCOM'00. [doi:10.1109/TNET.2004.826288] 4

[7] CHANDRA CHEKURI, ALINA ENE, AND NITISH KORULA: Unsplittable flow in paths and trees and column-restricted packing integer programs. In *Proc. 12th Internat. Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'09)*, pp. 42–55. Springer, 2009. [doi:10.1007/978-3-642-03685-9_4] 3

[8] CHANDRA CHEKURI AND SANJEEV KHANNA: Edge-disjoint paths revisited. *ACM Trans. Algorithms*, 3(4), 2007. Preliminary version in SODA'03. [doi:10.1145/1290672.1290683] 20

[9] CHANDRA CHEKURI, SANJEEV KHANNA, AND F. BRUCE SHEPHERD: An $O(\sqrt{n})$ approximation and integrality gap for disjoint paths and unsplittable flow. *Theory of Computing*, 2(7):137–146, 2006. [doi:10.4086/toc.2006.v002a007] 20

[10] CHANDRA CHEKURI, SANJEEV KHANNA, AND F. BRUCE SHEPHERD: Edge-disjoint paths in planar graphs with constant congestion. *SIAM J. Computing*, 39(1):281–301, 2009. Preliminary version in STOC'06. [doi:10.1137/060674442] 5

[11] JIANGZHUO CHEN, ROBERT D. KLEINBERG, LÁSZLÓ LOVÁSZ, RAJMOHAN RAJARAMAN, RAVI SUNDARAM, AND ADRIAN R. VETTA: (Almost) Tight bounds and existence theorems for single-commodity confluent flows. *J. ACM*, 54(4):16, 2007. Preliminary version in STOC'04. [doi:10.1145/1255443.1255444] 3

[12] JIANGZHUO CHEN, RAJMOHAN RAJARAMAN, AND RAVI SUNDARAM: Meet and merge: Approximation algorithms for confluent flows. *J. Comput. System Sci.*, 72(3):468–489, 2006. Preliminary version in STOC'03. [doi:10.1016/j.jcss.2005.09.009] 2, 3

[13] JULIA CHUZHOY: Routing in undirected graphs with constant congestion. *SIAM J. Computing*, 45(4):1490–1532, 2016. Preliminary version in STOC'12. [doi:10.1137/130910464, arXiv:1107.2554] 4

[14] JULIA CHUZHOY, ANUPAM GUPTA, JOSEPH S. NAOR, AND AMITABH SINHA: On the approximability of some network design problems. *ACM Trans. Algorithms*, 4(2):23:1–17, 2008. Preliminary version in SODA'05. [doi:10.1145/1361192.1361200] 4

[15] JULIA CHUZHOY AND SHI LI: A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2. *J. ACM*, 63(5):45:1–51, 2016. Preliminary version in FOCS'12. [doi:10.1145/2893472, arXiv:1208.1272] 4

[16] STEVEN COSARES AND IRAJ SANIEE: An optimization problem related to balancing loads on SONET rings. *Telecommunication Systems*, 3(2):165–181, 1994. [doi:10.1007/BF02110141] 2, 3

[17] YEFIM DINITZ, NAVEEN GARG, AND MICHEL X. GOEMANS: On the single-source unsplittable flow problem. *Combinatorica*, 19(1):17–41, 1999. Preliminary version in FOCS'98. [doi:10.1007/s004930050043] 3

[18] PATRICK DONOVAN, F. BRUCE SHEPHERD, ADRIAN R. VETTA, AND GORDON WILFONG: Degree-constrained network flows. In *Proc. 39th STOC*, pp. 681–688. ACM Press, 2007. [doi:10.1145/1250790.1250889] 3

[19] DANIEL DRESSLER AND MARTIN STREHLER: Capacitated confluent flows: complexity and algorithms. In *Proc. 7th Internat. Conf. on Algorithms and Complexity (CIAC'10)*, pp. 347–358. Springer, 2010. [doi:10.1007/978-3-642-13073-1_31] 3

[20] STEVEN FORTUNE, JOHN HOPCROFT, AND JAMES WYLLIE: The directed subgraph homeomorphism problem. *Theor. Comput. Sci.*, 10(2):111–121, 1980. [doi:10.1016/0304-3975(80)90009-2] 7

[21] VENKATESAN GURUSWAMI, SANJEEV KHANNA, RAJMOHAN RAJARAMAN, F. BRUCE SHEPHERD, AND MIHALIS YANNAKAKIS: Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. *J. Comput. System Sci.*, 67(3):473–496, 2003. Preliminary version in STOC'99. [doi:10.1016/S0022-0000(03)00066-7] 3, 4, 7, 8, 9, 20, 21

[22] JON M. KLEINBERG: *Approximation Algorithms for Disjoint Paths Problems*. Ph. D. thesis, MIT, 1996. Available from DSpace@MIT. 2

[23] JON M. KLEINBERG: Single-source unsplittable flow. In *Proc. 37th FOCS*, pp. 68–77. IEEE Comp. Soc. Press, 1996. [doi:10.1109/SFCS.1996.548465] 2, 3, 21

[24] JON M. KLEINBERG AND ÉVA TARDOS: Approximations for the disjoint paths problem in high-diameter planar networks. *J. Comput. System Sci.*, 57(1):61–73, 1998. Preliminary version in STOC'95. [doi:10.1006/jcss.1998.1579] 5

[25] STAVROS G. KOLLIOPOULOS AND CLIFFORD STEIN: Approximating disjoint-path problems using packing integer programs. *Math. Program.*, 99(1):63–87, 2004. Preliminary version in IPCO'98. [doi:10.1007/s10107-002-0370-6] 20, 21

[26] PETR KOLMAN: A note on the greedy algorithm for the unsplittable flow problem. *Inform. Process. Lett.*, 88(3):101–105, 2003. [doi:10.1016/S0020-0190(03)00351-X] 20

[27] GUYSLAIN NAVES, NICOLAS SONNERAT, AND ADRIAN R. VETTA: Maximum flows on disjoint paths. In *Proc. 13th Internat. Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'10)*, pp. 326–337. Springer, 2010. [doi:10.1007/978-3-642-15369-3_25] 7

[28] THÀNH NGUYEN: On the disjoint paths problem. *Oper. Res. Lett.*, 35(1):10–16, 2007. [doi:10.1016/j.orl.2006.02.001] 20

[29] NEIL ROBERTSON AND PAUL D. SEYMOUR: Graph minors. XIII. The disjoint paths problem. *J. Comb. Theory, Ser. B*, 63(1):65–110, 1995. [doi:10.1006/jctb.1995.1006] 7

[30] LOÏC SEGUIN-CHARBONNEAU AND F. BRUCE SHEPHERD: Maximum edge-disjoint paths in planar graphs with congestion 2. In *Proc. 52nd FOCS*, pp. 200–209. IEEE Comp. Soc. Press, 2011. [doi:10.1109/FOCS.2011.30] 5

[31] F. BRUCE SHEPHERD: Single-sink multicommodity flow with side constraints. In *Research Trends in Combinatorial Optimization*, pp. 429–450. Springer, 2009. [doi:10.1007/978-3-540-76796-1_20] 3

[32] F. BRUCE SHEPHERD, ADRIAN R. VETTA, AND GORDON WILFONG: Polylogarithmic approximations for the capacitated single-sink confluent flow problem. In *Proc. 56th FOCS*, pp. 748–758. IEEE Comp. Soc. Press, 2015. [doi:10.1109/FOCS.2015.51] 5

AUTHORS

F. Bruce Shepherd
Professor
McGill University, Montreal, Canada
bruce.shepherd@mcgill.ca
http://www.mcgill.ca/~bshepherd

Adrian R. Vetta
Associate professor
McGill University, Montreal, Canada
vetta@math.mcgill.ca
http://www.mcgill.ca/~vetta

ABOUT THE AUTHORS

BRUCE SHEPHERD graduated from C&O at the University of Waterloo in 1990. His advisor was Bill Pulleyblank. His thesis focused on packing problems and polyhedral combinatorics. Apart from his passion for the theory of algorithms and optimization, he has also endeavoured to apply the gospel in practice. He has worked as an operations research consultant in London and worked for 10 years at Bell Labs, Lucent Technologies.

ADRIAN VETTA is an Associate Professor in the Department of Mathematics & Statistics and the School of Computer Science at McGill University. He holds a B. Sc. and M. Sc. from the London School of Economics and a doctorate from MIT, written under the supervision of Santosh Vempala.