Improved Throughput for All-or-Nothing Multicommodity Flows with Arbitrary Demands

ANYA CHATURVEDI, Computer Science and Engineering, CIDSE, Arizona State University, US
CHANDRA CHEKURI, Dept. of Computer Science, University of Illinois at Urbana-Champaign, US
ANDRÉA W. RICHA, Computer Science and Engineering, CIDSE, Arizona State University, US
MATTHIAS ROST, Faculty of Electrical Engineering and Computer Science, TU Berlin, Germany
STEFAN SCHMID*, Faculty of Computer Science, University of Vienna, Austria
JAMISON WEBER, Computer Science and Engineering, CIDSE, Arizona State University, US

Throughput is a main performance objective in communication networks. This paper considers a fundamental maximum throughput routing problem — the *all-or-nothing multicommodity flow (ANF)* problem — in arbitrary *directed* graphs and in the practically relevant but challenging setting where *demands can be (much) larger than the edge capacities*. Hence, in addition to assigning requests to valid flows for each routed commodity, an admission control mechanism is required which prevents overloading the network when routing commodities.

Formally, the input for the ANF problem is an edge-capacitated directed graph $G = (V, E)$ and k source-destination pairs (s_i, t_i) of demand $d_i > 0$ and weight $w_i > 0$. The goal is to route a maximum weight subset of the given pairs (i.e., the weighted *throughput*), respecting the edge capacities; a pair (s_i, t_i) is routed if all of its demand d_i is routed from s_i to t_i (this is the all-or-nothing aspect); splitting (fractional) flows is allowed.

We make several contributions. On the theoretical side we obtain substantially improved bi-criteria approximation algorithms for this NP-hard problem. We present two non-trivial linear programming relaxations and show how to convert their fractional solutions into integer solutions via randomized rounding. One is an exponential-size formulation (solvable in polynomial time using a separation oracle) that considers a "packing" view and allows a more flexible approach, while the other is a generalization of the compact LP formulation of Liu et al. (INFOCOM'19) that allows for easy solving via standard LP solvers. We prove the "equivalence" of the two relaxations and highlight the advantages of each of the two approaches. Via these, we obtain a polynomial-time randomized algorithm that yields an arbitrarily good approximation on the weighted throughput while violating the edge capacity constraints by at most an $O(\min\{k, \log n/\log\log n\})$ multiplicative factor. This improves on the best-known previous result by Liu et al., which achieved a 1/3 throughput approximation and an edge capacity violation ratio of $O(\sqrt{k \log n})$. We also describe a deterministic rounding algorithm by derandomization, using the method of pessimistic estimators.

We complement our theoretical results with a proof of concept empirical evaluation, considering a variety of network scenarios. We study two different ways to solve the LP efficiently in terms of time and space: (a) by solving the compact ANF formulation directly using an off-the-shelf solver, and (b) by approximately solving the packing LP relaxation via a well-known multiplicative weight update (MWU) approach (based on Lagrangean relaxation) or via a faster MWU-based heuristic called permutation routing. We highlight the benefits of the ANF packing LP formulation by presenting some more general scenarios of interest to networking applications (such as routing along short paths or a small number of paths) that this formulation allows.

*[Research supported by the ERC Consolidator project AdjustNet, grant agreement No. 864228.]

Authors' addresses: Anya Chaturvedi, Computer Science and Engineering, CIDSE, Arizona State University, US, anya.chaturvedi@asu.edu; Chandra Chekuri, Dept. of Computer Science, University of Illinois at Urbana-Champaign, US, chekuri@illinois.edu; Andréa W. Richa, Computer Science and Engineering, CIDSE, Arizona State University, US, aricha@asu.edu; Matthias Rost, Faculty of Electrical Engineering and Computer Science, TU Berlin, Germany, mrost@inet.tu-berlin.de; Stefan Schmid, Faculty of Computer Science, University of Vienna, Austria, stefan_schmid@univie.ac.at; Jamison Weber, Computer Science and Engineering, CIDSE, Arizona State University, US, jwweber@asu.edu.

1 INTRODUCTION

The study of routing and multicommodity flow problems is motivated by many real-world applications, e.g., related to the optimization of communication and traffic networks, as well as by the crucial role flows and cuts play in combinatorial optimization [7]. In this paper, we are interested in throughput optimization in the context of communication networks serving multiple commodities. Throughput is a most fundamental performance metric in many networks [23], and we are particularly interested in the practically relevant scenario where flows have certain minimal performance or quality-of-service requirements, in the sense that they need to be served in an *all-or-nothing* manner eith respect to their respective demands.

Our problem belongs to the family of *all-or-nothing (splittable) multicommodity flow* problems. In contrast to most existing literature, we consider a more realistic model in the following respects:

- The underlying communication graph can be *directed*. This is motivated by the fact that in most practical communication networks (e.g., optical networks or wireless networks), the available capacities in the different link directions can differ.
- A single commodity demand can be larger than the capacity of a single link or path. Consider for example a bulk transfer, or the fact that traffic patterns are often highly skewed, with a small number of elephant flows consuming a significant amount of bandwidth resources [30]. Only *splittable* flows can serve such demands.
- The total demand can be larger than the network capacity. To make efficient use of the given network resources, we hence need a clever *admission control* mechanism, in addition to a routing algorithm.

We define the *All-or-Nothing (Splittable) Multicommodity Flow (ANF)* problem as follows: It takes as input a flow network modeled as a capacitated directed graph $G(V, E)$, where V is the set of nodes, E is the set of edges, and each edge e has a capacity $c_e > 0$. Let $n = |V|$ and $m = |E|$. We are given a set of source-destination pairs (s_i, t_i), where $s_i, t_i \in V$, $i \in [k]$[1], each with a given (non-uniform) demand $d_i > 0$ and weight $w_i > 0$. The edge capacities c_e, the demands d_i and the weights w_i can be arbitrary positive functions on n and k, for any $e \in E$ or $i \in [k]$. A valid set of flows for commodities $1, \ldots, k$ in G (i.e., a valid *multicommodity flow*), must satisfy standard flow conservation constraints for each commodity i, which imply that the amount of flow for commodity i entering a node v has to be equal to the flow for commodity i leaving v, if $v \neq s_i, t_i$. The *load* of an edge e, given by the sum of the flows for all commodities on e, must not exceed the edge's capacity c_e. Commodity i is *satisfied* if d_i units of flow of this commodity can be successfully routed from s_i to t_i in the network. (See also our mixed integer program formulation later).

We aim to maximize the total profit of a subset of commodities that can be concurrently satisfied in a valid multicommodity flow. Specifically, the goal is to find a subset $K' \subseteq [k]$ of commodities to be concurrently satisfied such that the (weighted) *throughput*, given by $\sum_{i \in K'} w_i$, is maximized over all possible K'. The flow can be *split* arbitrarily along many branching routes (subject to flow conservation and edge capacity constraints) and does not have to be integral.

The ANF problem was introduced in [9] as a relaxation of the classical Maximum Edge-Disjoint Paths problem (MEDP) and is known to be NP-Hard and APX-hard even in the restricted setting of unit demands and when the underlying graph is a tree [9, 16]. In directed graphs, even with unit demands, the problem is hard to approximate to within an $n^{\Omega(1/c)}$ factor even when edge capacities are allowed to be violated by a factor c [13]. When demands can exceed the minimum capacity strong lower bounds exist even in very restricted settings [31]. Hence, the literature has followed a bi-criteria optimization approach where edge capacities can be violated slightly. Namely, in this

[1] Let $[x]$ denotes the set $\{1, \ldots, x\}$, for any positive integer x.

paper we seek an (α, β)-*approximation algorithm*: For parameters $\alpha \in (0, 1]$ and $\beta \geq 1$, we seek a polynomial time algorithm that outputs a solution to the ANF problem whose throughput is at least an *α fraction of the maximum throughput* and whose *load on any edge e is at most β times the edge capacity c_e*, with high probability (With probability at least $1 - 1/n^c$, where $c > 0$ is a constant). The parameter β hence provides an upper bound on the *edge capacity violation ratio* (or *congestion*) incurred by the algorithm.

1.1 Our Contributions

This paper revisits a fundamental maximum throughput routing problem, the all-or-nothing multicommodity flow (ANF) problem, considering a more general and practical setting where the network topology can be an *arbitrary directed graph*, with *arbitrary commodity demands* that can be much larger than the edge capacities, in contrast to most of the existing work in the literature. This model is challenging as it not only requires a clever algorithm to efficiently route the splittable commodities across the directed and capacitated network, but also an admission control policy.

We make several contributions. On the theoretical side, we obtain substantially improved bicriteria approximation algorithms for this NP-hard problem. More specifically,

- We present *two non-trivial linear programming relaxations*: One is an exponential-size formulation (solvable in polynomial time using a separation oracle) that considers a *"packing"* view and allows a more flexible approach, while the other is a generalization of the *compact edge-flow* LP formulation of Liu et al. [21] that allows for *arbitrary non-uniform demands and weights* and that also allows for easy solving via standard LP solvers. We prove the *"equivalence"* of the two relaxations and highlight the advantages of each of the two approaches.
- Via these relaxations, we obtain a polynomial-time *randomized rounding* algorithm that yields an $(1 - \epsilon)$ *throughput approximation*, for any $1/m < \epsilon < 1$, with an *edge capacity violation ratio* of $O(\min\{k, \log n / \log \log n\})$, with high probability.
- We also present a *deterministic rounding algorithm by derandomization*, using the method of pessimistic estimators. Contrary to most algorithms obtained this way, our derandomized algorithm is simple enough to be also of relevance in practice.

In addition, our packing framework for ANF has interesting networking applications, beyond the specific model considered in this paper. We discuss different examples, related to *unsplittable flows*, flows that are *split into a small number of paths*, *routing along disjoint paths* for fault-tolerance, using *few edges for the flow*, or routing flow along and *short paths*.

As a proof of concept, we show how to *engineer our algorithms* for practical scenarios. To this end, we couple three algorithms that allow one to compute the relaxed LP solutions efficiently, in terms of time and space, with both our randomized and derandomized algorithms. The first algorithm *directly solves the compact ANF formulation* using an off-the-shelf solver, in our case CPLEX; the second algorithm approximately solves the packing LP relaxation via a well-known *multiplicative weight update (MWU)* approach, based on Lagrangean relaxation; the last and third algorithm is a faster MWU-based heuristic called *permutation routing*. We provide general guidelines about the relative efficacy of these algorithms in specific real-world networks. As a contribution to the research community, to ensure reproducibility and facilitate follow-up work, we will release our implementation (source code) and experimental artefacts with this paper.

1.2 Novelty and Related Work

Liu et al. [21] presented a $(1/3, O(\sqrt{k \log n}))$-approximation algorithm for the ANF problem for the case of *uniform* demands and weights in directed graphs, where k is the number of commodities (note that while they consider the case of uniform demands, they also do not assume any restriction on how large these demands can be when compared to the edge capacities). Our current work significantly improves and generalizes the randomized rounding framework outlined in [21], in several ways: (a) We are able to achieve an arbitrarily good throughput approximation bound; (b) our bound on the edge capacity violation does not depend on the number of commodities k[2], and significantly improves on the bound of $O(\sqrt{k \log n})$in [21]; and (c) we were also able to accommodate arbitrary non-uniform demands and commodity weights. In addition, we provide a derandomized algorithm for the ANF problem and a more flexible packing MIP formulation for the ANF problem that leads to several interesting extensions of practical interest.

Other work on bi-criteria (α, β)-approximation schemes for the ANF problem that are closely related to ours aims at keeping β constant, while letting α be a function of n. The work of Chekuri et al. [9, 10, 12] is the most relevant and was also the first to formalize the ANF problem. Their work implies an approximation algorithm for the general (weighted, non-uniform demands) ANF problem in *undirected graphs* with $\alpha = \Omega(1/\log^3 k)$ and $\beta = 1$. A requirement of their algorithm is that $\max_i d_i \le \min_e c_e$. This is a strong assumption, since it eliminates all (undirected) networks G where the above assumption fails, such as for example complete graphs with unit edge capacities and demands $2 \le d_i \le n-1$, for all i. Hence, besides the fact that our approximation guarantees differ from those of [9] (we have constant α and logarithmic β, while they achieve constant β at the expense of a polylogarithmic $1/\alpha$), our results also apply to *any directed graph* G, without any assumptions on how d_i compares to individual edge capacities. We note that even in undirected graphs and unit demand the ANF problem does not admit a constant factor approximation if constant congestion is allowed [3]. Thus, obtaining a good throughput approximation even in restricted settings requires congestion violation.

The ANF problem gets considerably more challenging in directed graphs. Chuzhoy et al. [13] show that, even if restricted to unit demands, the problem is hard to approximate to within polynomial factors in directed graphs when constant congestion is allowed. In [7], Chekuri and Ene consider a variation of the ANF problem — the *Symmetric All or Nothing Flow (SymANF)* problem — in *directed* graphs with *symmetric unit demand pairs and unit edge capacities*, also aiming at constant β and polylogarithmic $1/\alpha$. In SymANF, the input pairs are unordered and a pair $s_i t_i$ is routed if and only if both the ordered pairs (s_i, t_i) and (t_i, s_i) are routed; the goal is to find a maximum subset of the given demand pairs that can be routed. The authors provide a poly-logarithmic approximation with constant congestion for SymANF, by extending the well-linked decomposition framework of [11] to the directed graph setting with symmetric demand pairs. However, their approach, like the one for undirected graphs is limited to the setting where $\max_i d_i \le \min_e c_e$. As explained above, our work considers a more general network setting where demand pairs need not be symmetric and demands values can exceed the capacities. Further, our goal is to obtain an arbitrarily good approximation of the throughput while relaxing the capacity violation which is different regime.

The *Maximum Edge-Disjoint Paths (MEDP)* [15] problem considers a set of pairs of nodes to be routable if they can be connected using edge-disjoint paths and aims at finding the largest number of routable pairs. The *Unsplittable Flow Problem* (UFP) is a generalization of MEDP to non-uniform demands while requiring that all flow for a pair is routed along a single path. MEDP and UFP are classical routing problems and have been extensively studied in VLSI routing where the constraint of using a single path for connecting pairs is particularly important. MEDP and UFP tend to be harder to approximate than ANF. For instance, even for unit demands and undirected graphs MEDP is hard to approximate to almost any polynomial factor [14], and in directed graphs the problem is hard to approximate to within an $\Omega(m^{1/2-\epsilon})$-factor [18]. MEDP and UFP have been mostly considered under the no-bottleneck assumption, that is, when $\max_i d_i \le \min_e c_e$. Without this

[2]Unless k is very small $o(\log n / \log\log n)$, in which case we get an approximation bound of k.

assumption UFP becomes hard to approximate to within an $m^{1/2-\epsilon}$ factor even for very restricted settings [31].

Finally, our work leverages randomized rounding techniques presented by Rost et al. [28, 29] in the different context of virtual network embedding problems (i.e., in their context, flow endpoints are subject to optimization).

1.3 Organization

The remainder of the paper is organized as follows. We introduce our packing framework in Section 2 and the compact edge-flow formulation in Section 3. The multiplicative-weight-update (MWU) algorithm is described in Section 4, our randomized rounding algorithm in Section 5, and our derandomized algorithm in Section 6. We discuss more general applications of our packing framework in Section 7. We report on simulation results in Section 8, and conclude in Section 9.

2 A PACKING FRAMEWORK FOR ANF

We develop two non-trivial mixed integer programming (MIP) formulations for the ANF problem, presented in this section and in Section 3. In our approach, we compute their linear programming (LP) relaxation solutions in polynomial time and then convert these solutions into integer solutions via appropriate randomized rounding. In this section we present the first such MIP formulation, that takes a "packing" view of the ANF problem and allows for a more flexible approach, as we discuss below and in Section 7. In this formulation, we will be packing an entire flow assignment for each commodity i, selected from the set of all possible valid flows between s_i and t_i. Since the number of possible flows will be exponential, this formulation has exponential size, but we show that its LP relaxation can still be solved in polynomial time via a separation oracle. This is akin to use the path formulation for flows rather than the edge-based flow formulation. This perspective allows one to easily see why the randomized rounding framework for rounding paths easily generalizes to rounding "flows".

Recall that the input is a directed graph $G = (V, E)$, with $n = |V|$ and $m = |E|$, and with edge capacities $c : E \to \mathbb{Z}_+$ and k demand pairs $(s_1, t_1), \ldots, (s_k, t_k)$. Each demand pair i has an associated non-negative weight w_i and a non-negative integer demand d_i. We say that $f : E \to \mathbb{R}_+$ is a *valid* flow for pair i if f routes d_i units from s_i to t_i in G and respects the edge capacities. Note that if pair i cannot be routed in isolation in G then we may as well discard it (since there are no valid flows for i). Let $\mathcal{F}_i$ denote the set of all valid flows for pair i. Each $\mathcal{F}_i$ is not necessarily a finite set. However, we can restrict attention to a finite set of flows by considering the polyhedron of all feasible s_i-t_i flows in G and considering only the finitely many vertices of that polyhedron; any valid flow can be expressed as a convex combination of the flows defined by the polyhedron's vertices.

We now describe a mixed integer programming formulation that captures the ANF problem. This formulation is very large: In general it can be exponential in n, m and k. For each i, we have a binary variable x_i to indicate whether commodity i is routed or not. For each i and each valid flow $f \in \mathcal{F}_i$, we have a variable $y(f)$ to indicate the fraction of x_i that is routed using the flow f. For a flow f we let $f(e)$ denote the amount of flow on e used by f; note that $f(e)$ is fixed, for each f and e, and hence is not a variable.

The following lemma is easy to see.

Lemma 2.1. *The formulation shown in Fig 1a(a) is an exact formulation for the ANF problem.*

We will now focus on solving and rounding the LP relaxation of the preceding formulation; we simplify it by eliminating the variables x_i. See Fig 1b(b).

$$\max \sum_{i=1}^{k} w_i x_i$$
$$\sum_{f \in \mathcal{F}_i} y(f) = x_i \qquad 1 \leq i \leq k$$
$$\sum_{i=1}^{k} \sum_{f \in \mathcal{F}_i} f(e) y(f) \leq c(e) \qquad e \in E$$
$$x_i \in \{0, 1\} \qquad 1 \leq i \leq k.$$

(a) Mixed integer programming formulation for ANF based on "flow" variables.

$$\max \sum_{i=1}^{k} w_i \sum_{f \in \mathcal{F}_i} y(f)$$
$$\sum_{f \in \mathcal{F}_i} y(f) \leq 1 \qquad 1 \leq i \leq k$$
$$\sum_{i=1}^{k} \sum_{f \in \mathcal{F}_i} f(e) y(f) \leq c(e) \qquad e \in E$$
$$y(f) \geq 0 \qquad f \in \mathcal{F}_i, 1 \leq i \leq k.$$

(b) LP Relaxation.

2.1 Solving the Packing LP Relaxation

It is not at first obvious that the LP relaxation of the ANF MIP can be solved in polynomial time. There are two ways to see why this is indeed possible. One is to show via the Ellipsoid method that the dual has an *efficient separation oracle for the dual LP* and the other is to describe an *equivalent compact (polynomial-size) formulation* to the ANF LP. In this section, we will present the former approach, which gives us a more flexible formulation that leads to interesting extensions and that also leads to simpler proofs. In Section 3, we will present the compact formulation, of size polynomial in n and k, and show that its relaxation is equivalent to the relaxation of the formulation in Figure 1b(b). The benefits of the compact formulation are that it directly leads to simple randomized and derandomized algorithms, that can be efficiently implemented, as we show in Section 8.

In Figure 2, we present the dual LP to the formulation in Figure 1b(b). There are two types of variables: First, for each of the capacity constraints, we associate a dual variable ℓ_e and for each constraint limiting the total flow to 1 we associate a dual variable z_i. (Recall that the value $f(e)$ is a constant and not a variable.)

The following lemma shows that one can use a polynomial-time separation oracle for solving the dual LP.

Lemma 2.2. *There is a polynomial-time separation oracle for the dual LP.*

Proof. The dual LP is easily seen to reduce to s-t minimum-cost flow. Given non-negative values for the variables $\ell_e, e \in E$ and $z_i, 1 \leq i \leq k$ we compute the minimum-cost flow for each pair (s_i, t_i) of d_i units with edge costs given by $\ell_e, e \in E$. Let this cost be q_i. The values are feasible for the dual

$$\min \sum_{e \in E} c(e)\ell_e + \sum_{i=1}^{k} z_i$$
$$z_i + \sum_{e \in E} f(e)\ell_e \geq w_i \qquad 1 \leq i \leq k, f \in \mathcal{F}_i$$
$$\ell_e \geq 0 \qquad e \in E$$
$$z_i \geq 0 \qquad 1 \leq i \leq k$$

Fig. 2. Dual of the LP relaxation for ANF.

iff $z_i + q_i \geq w_i$ for $1 \leq i \leq k$. If there is an i for which $z_i + q_i < w_i$ the corresponding minimum cost flow f for pair i defines the violated constraint. Minimum-cost flow is poly-time solvable and hence there is a poly-time separation oracle for the dual LP. □

Standard techniques allow one to solve the primal LP from an optimum solution to the dual LP. However, since the Ellipsoid method is impractical, in Sections 3 and 4, we present two efficient ways of solving the ANF packing LP in practice, which we will use in our implementations. Andrea2

2.2 Rounding the Packing LP Relaxation

In this section, we show how to round a (fractional) solution to the primal ANF MIP formulation. We will need the following standard Chernoff bound (see [24]):

Theorem 2.3. *Let $X_1, \ldots, X_n$ be n independent random variables (not necessarily distributed identically), with each variable X_i taking a value of 0 or v_i for some value $0 < v_i \leq 1$. Let $X = \sum_{i=1}^{n} X_i$ be their sum. Then the following hold:*

- *For $\mu \geq E[X]$ and $\delta > 0$, $\Pr[X \geq (1+\delta)\mu] < \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^\mu$.*
- *For $0 \leq \mu \leq E[X]$ and $\delta \in (0,1)$, $\Pr[X \leq (1-\delta)\mu] < e^{-\delta^2\mu/2}$.*

Randomly rounding a feasible solution to the LP relaxation is straightforward, and is very similar to the standard rounding via the path formulation for the Maximum Edge Disjoint Problem (MEDP) pioneered in the work of Raghavan and Thompson [27]. Once the LP relaxation is solved, we consider the support of the solution. For each pair i, the LP relaxation identifies some h_i flows $f_{i_1}, f_{i_2}, \ldots, f_{i_{h_i}} \in \mathcal{F}_i$ along with non-negative values $y(f_{i_1}), \ldots, y(f_{i_{h_i}})$ such that their sum is at most 1. The randomized algorithm simply picks for each i independently, at most one of the flows in its support where the probability of picking f_{i_j} is exactly $y(f_{i_j})$. Note that the probability that one chooses to route pair i is exactly $\sum_{f \in \mathcal{F}_i} y(f) \leq 1$.

We will analyze the algorithm with respect to the weight of the LP solution $\sum_{i=1}^{k} w_i \sum_{j=1}^{h_j} y(f_{i,j})$. We refer to this quantity as W_{LP}. We refer to the value of an optimum LP solution as OPT_{LP} and the value of an optimum integer solution as OPT_{IP}. We observe that $\text{OPT}_{\text{LP}} \geq \text{OPT}_{\text{IP}}$ and $\text{OPT}_{\text{LP}} \geq W_{\text{LP}}$. Note that when solving the formulation in Figure 1b(b) or the compact formulation presented in Section 3, the LP solution obtained will be optimal and hence $W_{\text{LP}} = \text{OPT}_{\text{LP}}$; however, the solution obtained via the multiplicative-weight update algorithm of Section 4 may only approximate OPT_{LP} and hence one could indeed have $\text{OPT}_{\text{LP}} > W_{\text{LP}}$. We will also assume that $\text{OPT}_{\text{LP}} \geq w_{\max}$, since we can discard from consideration any commodity i that cannot be routed alone in the network, as it will never be part of a feasible solution of the MIP formulation, and hence $w_{\max} \leq \text{OPT}_{\text{IP}} \leq \text{OPT}_{\text{LP}}$.

Lemma 2.4. *Let Z be the (random) weight of the pairs chosen to be routed by the algorithm. Then $E[Z] = W_{LP}$ and $\Pr[Z < (1-\delta)W_{LP}] < e^{-\frac{\delta^2}{2}\frac{W_{LP}}{w_{\max}}}$. In particular, $\Pr[Z < (1-\delta)W_{LP}] < e^{-\delta^2/2}$.*

Proof. Let Y_i be the indicator for pair i being chosen to be routed. We have $Z = \sum_{i=1}^k w_i Y_i$. The rounding algorithm implies that $\Pr[Y_i = 1] = \sum_{f \in \mathcal{F}_i} y(f)$. Hence, by linearity of expectation, $E[Z] = \sum_i w_i E[Y_i] = \sum_i w_i \sum_{f\in\mathcal{F}_i} y(f) = W_{\text{LP}}$. Let $Z_i = \frac{w_i}{w_{\max}} Y_i$; note that $Z_i \le 1$ and $\sum_i Z_i = \frac{1}{w_{\max}} Z$. Let $Z' = \sum_i Z_i$. Since Z' is a sum of independent random variable each of which is in $[0,1]$ we can apply the lower-tail Chernoff bound for Z', and obtain a lower-tail bound for Z.

$$\Pr[Z < (1-\delta)W_{\text{LP}}] = \Pr[Z' < (1-\delta)W_{\text{LP}}/w_{\max}] = \Pr[Z' < (1-\delta)E[Z']] \le e^{-\frac{\delta^2}{2}(W_{\text{LP}}/w_{\max})}.$$

□

Lemma 2.5. *For $m \ge 9$ and $b > 1$ the probability that the total flow on an edge e is more than $(3b\ln m/\ln\ln m)c(e)$ is at most $e^{-1.5b\ln m - 3b\ln b\ln m/\ln\ln m - 1}$. Via the union bound, the probability that the total flow on* any *edge e is more than $(3b\ln m/\ln\ln m)c(e)$ is at most $e^{-(1.5b-1)\ln m - 3b\ln b\ln m/\ln\ln m - 1}$.*

Proof. Let X_e be the random variable indicating the total flow on edge e. Let $X_{e,i}$ be the flow on e from the flow chosen for pair i. We have $X_e = \sum_{i=1}^k X_{e,i}$ and moreover the variables $X_{e,i}$, $1 \le i \le k$ are independent by the algorithm. Note that $0 \le X_{e,i} \le c(e)$ since each flow in $\mathcal{F}_i$ is a valid flow by definition. Further

$$E[X_e] = \sum_i E[X_{e,i}] = \sum_{i=1}^k \sum_{f\in\mathcal{F}_i} f(e)y(f) \le c(e).$$

We now apply the Chernoff bound to see that $\Pr[X_e > (3b\ln m/\ln\ln m)c(e)] \le \frac{e^\delta}{(1+\delta)^{(1+\delta)}}$ where $(1+\delta) = 3b\ln m/\ln\ln m$; we note that the standard bound has all variables bounded in $[0,1]$ while all our variables are in $[0, c(e)]$ but we can simply scale all variables by $c(e)$. We have $\frac{e^\delta}{(1+\delta)^{(1+\delta)}} = e^{\delta - (1+\delta)\ln(1+\delta)}$. We consider the expression $\delta - (1+\delta)\ln(1+\delta)$ $(1+\delta) = 3b\ln m/\ln\ln m$.

$$\begin{aligned}
\delta - (1+\delta)\ln(1+\delta) &= (1+\delta)(1-\ln(1+\delta)) - 1 \\
&= (3b\ln m/\ln\ln m)(1 - \ln(3b\ln m/\ln\ln m)) - 1 \\
&= (3b\ln m/\ln\ln m)(1 - \ln 3 - \ln b - \ln\ln m + \ln\ln\ln m) - 1 \\
&\le (3b\ln m/\ln\ln m)(-\ln b - \frac{1}{2}\ln\ln m) - 1 \\
&\le -1.5b\ln m - 3b\ln b\ln m/\ln\ln m - 1.
\end{aligned}$$

In the above we used the fact that $\ln\ln m - \ln\ln\ln m \ge \frac{1}{2}\ln m$ for $m \ge 9$.

The second part follows easily via the union bound over all the m edges. □

We can now put together the preceding lemmas to derive our bicriteria approximation. We will henceforth assume that $m \ge 9$. Let S be the random set of pairs routed by the algorithm. Let $\mathcal{E}_1$ be the event that $w(S) < (1-\delta)W_{\text{LP}}$. From Lemma 2.4, $\Pr[\mathcal{E}_1] \le e^{-\delta^2/2}$. Let $\mathcal{E}_2$ be the event that there is some edge e such that the flow on e is more than $(3b\ln m/\ln\ln m)c(e)$. From Lemma 2.5 $\Pr[\mathcal{E}_2] \le e^{-1.5b\ln m - 3b\ln b\ln m/\ln\ln m - 1}$. For $b = 1$ and $m \ge 9$ we see that $\Pr[\mathcal{E}_2] \le e^{-12}$. Chossing $\delta = 1/2$, $\Pr[\mathcal{E}_1] \le 0.8825$. Thus $\Pr[\mathcal{E}_1 \text{ or } \mathcal{E}_2] \le 0.9$. This implies that with probability ≥ 0.1, the set S of routed pairs satisfies the property that $w(S) \ge 0.5W_{\text{LP}}$ *and* the congestion of every edges is at most $3\ln m/\ln\ln m$. In other words, if $W_{\text{LP}} = \text{OPT}_{\text{LP}} \le \text{OPT}_{\text{IP}}$, we obtain a $(1/2, 3\ln m/\ln\ln m)$ bicriteria approximation with probability at least 0.1. One can boost the success probability by repetition. If the rounding is repeated $10c\ln m$ times, then with probability at least

$1-(1-0.1)^{10c\ln m} \geq 1-m^{-c}$ (in other words with high probability), one of the rounded solutions is a $(1/2, 3\ln m/\ln\ln m)$ bicriteria approximation

We now refine the preceding argument to show that the quality of the rounded solution can get arbitrarily close to W_{LP} but with lower probability, and examine the tradeoff required in the congestion and number of repetitions required. Suppose we want $w(S) \geq (1-\epsilon)W_{\text{LP}}$ for some small $0 < \epsilon < 1/2$. Let $\mathcal{E}_1$ be the event that this does not happen. Then from Lemma 2.4 $\Pr[\mathcal{E}_1] \leq e^{-\epsilon^2/2}$. Let $\mathcal{E}_2$ be the event that some edge congestion exceeds $3b\ln m/\ln\ln m$. Lemma 2.5 allows us to upper bound this probability. Suppose we choose b such that $\Pr[\mathcal{E}_2] \leq \epsilon^2/6$. Then

$$\Pr[\bar{\mathcal{E}_1} \cap \bar{\mathcal{E}_2}] = 1 - \Pr[\mathcal{E}_1 \cup \mathcal{E}_2] \geq (1 - \Pr[\mathcal{E}1] - \Pr[\mathcal{E}_2]) \geq 1 - e^{-\epsilon^2/2} - \epsilon^2/6 \geq \epsilon^2/6.$$

This would yield a $(1-\epsilon, 3b\ln m/\ln\ln m)$ bicriteria approximation with probability at least $\epsilon^2/6$ and one can boost this via repeating $O(\frac{1}{\epsilon^2}\ln m)$ times to get the approximation with high probability. Thus it remains to estimate b such that $\Pr[\mathcal{E}_2] \leq \epsilon^2/6$. From Lemma 2.5, it suffices to choose b such that

$$(1.5b-1)\ln m + 3b\ln b\ln m/\ln m\ln m + 1 \geq \ln(6/\epsilon^2).$$

In particular it suffices to have $b \geq c\frac{\ln(1/\epsilon)}{\ln m}$ for some fixed constant c. Thus for all $\epsilon \geq 1/m$ a fixed constant b (e.g., $b = 1.85$) suffices! Note however that the number of repetitions grows as $\Omega(1/\epsilon^2)$ to guarantee a good solution with high probability.

Theorem 2.6. *For $m \geq 9$ and any fixed $\epsilon > 0$ there is a polynomial-time randomized algorithm that yields a $(1-\epsilon, O(\ln m/\ln\ln m + \ln(1/\epsilon)/\ln m))$-approximation with high probability. Moreover, by setting $\epsilon = 1/m$, we guarantee a $O(1-1/m, O(\ln m/\ln\ln m))$-approximation with high probability.*

Noting that it is trivial to get a $(1, k)$-approximation by simply routing all the commodities at full demand, we get the following corollary, stating our full approximation guarantees:

Corollary 2.7. *For $m \geq 9$ and any $\epsilon \geq 1/m$ (or any fixed $0 < \epsilon < 1$) there is a polynomial-time randomized algorithm that yields a $(1-\epsilon, \min\{k, O(\ln m/\ln\ln m)\})$-approximation with high probability.*

Due to space limitations, we describe a different rounding approach in Appendix A, using an *alteration approach*, that may also be of interest in certain settings and gives a better tradeoff in terms of repetitions.

3 COMPACT EDGE-FLOW FORMULATION

As we saw, one can solve the ANF packing LP via the Ellipsoid method. While this leads to a polynomial-time algorithm for solving the LP, implementing such algorithm would not be trivial nor be very efficient in practice. In this section, we present an alternative polynomial-size compact edge-flow formulation for the ANF problem, which can be solved more efficiently in practice that the packing LP. In Section 4, we present another approach for solving the packing LP more efficiently, albeit only approximately. Both approaches were evaluated in simulations in Section 8.

We generalize the compact edge-flow based LP formulation for the ANF problem presented by Liu et al. [21] to accommodate arbitrary demands and commodity weights in Figure 3. We use an indicator variable $f_i \in \{0,1\}$ to indicate whether a commodity i is successfully routed through G. Next, we denote $f_{i,e} \in [0,1]$ as the fraction of flow for commodity i allocated to a particular edge $e \in E$. The total flow assigned to a fixed edge e is given by $\sum_i d_i \cdot f_{i,e}$ and the total weighted throughput is given by $\sum_i w_i f_i$. Constraint (2) defines the value of the total flow for each commodity i, (3) enforces flow conservation for each i, and (4) stipulate that no edge capacity is violated by the flow assignments. Constraint (5) ensures that for a fixed commodity i, the ratio of flow assigned to an edge e to the total flow of that commodity does not exceed the capacity of e: These constraints are actually redundant for the MIP formulation, but will strengthen the LP relaxation of Figure 3, obtained by allowing each f_i to assume any real value in $[0, 1]$. In fact, (5) is crucial to establish the "equivalence" between the LP relaxation of the ANF packing formulation (Figure 1b(b)) and the LP relaxation of the compact edge-flow MIP.

This formulation has size polynomial in n and k and hence can be solved in polynomial time (e.g., using the Ellipsoid method). Moreover, given the compact nature of the LP, one can use a standard LP solver in practice.

$$\max \sum_{i=1}^{k} w_i f_i \tag{1}$$

$$\sum_{(s_i,v)\in E} f_{i,(s_i,v)} = f_i \qquad \forall i \in [k] \tag{2}$$

$$\sum_{(u,v)\in E} f_{i,(u,v)} = \sum_{(v,u)\in E} f_{i,(v,u)} \qquad \forall i \in [k], \forall v \in V - \{s_i, t_i\} \tag{3}$$

$$\sum_{i=1}^{k} f_{i,(u,v)} \cdot d_i \leq c_{(u,v)} \qquad \forall (u,v) \in E \tag{4}$$

$$f_{i,(u,v)} \cdot d_i \leq f_i \cdot c_{(u,v)} \qquad \forall i \in [k], \forall (u,v) \in E \tag{5}$$

$$f_{i,(u,v)} \geq 0 \qquad \forall i \in [k], \forall (u,v) \in E \tag{6}$$

$$f_i \in \{0, 1\} \qquad \forall i \in [k] \tag{7}$$

Fig. 3. Compact Edge-Flow ANF Formulation

Equivalence with Packing Formulation. Here we prove that the packing formulation in Figure 1b(b) is "equivalent" to the compact formulation given in Figure 3. When we say equivalent we mean the following: Given a feasible solution to one LP we can obtain a feasible solution to the other LP of the same value. We prove both directions below.

First, consider a feasible solution to the compact formulation. Let $f_i \in [0, 1]$ be the amount to which commodity i is routed and let $f_{i,e} \in [0, 1]$ be value on edge $e \in E$. We first construct a flow $g_i : E \longrightarrow \mathbb{R}_+$ of d_i units from s_i to t_i: we set $g_i(e) = d_i f_{i,e}/f_i$. It is easy to verify that g_i is a flow of d_i units from s_i to t_i. Moreover by the strengthening constraint (5) in Figure 3, we see that $g_i(e) \leq c(e)$ for all e and hence g_i is a feasible flow in the capacities. Putting together these facts, $g_i \in \mathcal{F}_i$. We obtain a feasible solution to the packing formulation as follows. For each i we set $x_i = f_i$ and we set $y(f) = x_i$ for $f = g_i$ and $y(f) = 0$ for every other $f \in \mathcal{F}_i$. In other words we are using only one flow for each commodity i. The only non-trivial fact to check is that this solution is feasible. For this we need to verify that $\sum_i y(g_i)g_i(e) \leq c(e)$ but this easily follows from our definition of g_i's and constraint (4) in Figure 3. Since $x_i = f_i$ for all i, we see that the two solutions have the same value.

Second, consider a feasible solution x, y to the packing formulation in Figure 1b(b). Let x_i be the amount routed for commodity i and for each flow $f \in \mathcal{F}_i$, $y(f)$ is the amount routed on f with $\sum_{f\in\mathcal{F}_i} = x_i$. We construct a feasible solution to the compact LP as follows. For each commodity i we set $f_i = x_i$. For each $e \in E$ and each $i \in [k]$, we set $f_{i,e} = \frac{1}{d_i} \sum_{f\in\mathcal{F}_i} f(e)y(f)$. Note that f_i is simply scaling by d_i the total flow on e from all $f \in \mathcal{F}_i$. Since each $f \in \mathcal{F}_i$ is a flow of d_i units from s_i to t_i and $\sum_{f\in\mathcal{F}_i} y(f) = x_i$ we see that $f_{i,e}, e \in E$ corresponds to sending a total of x_i units of flow from s_i to t_i. We focus on constraints (4) and (5) in Figure 3. We observe that

$\sum_i d_i f_{i,e} = \sum_i d_i \sum_{f \in \mathcal{F}_i} \frac{1}{d_i} \sum_{f \in \mathcal{F}_i} f(e)y(f) = \sum_i \sum_{f \in \mathcal{F}_i} y(f)f(e)$ and the last term is at most $c(e)$ from the feasiblity of given solution for the packing formulation. This proves that (4) in Figure 3 is satisfied for the solution we constructed. We observe that for each $f \in \mathcal{F}_i$ and each $e \in E$ we have $f(e) \le c(e)$ since f is a feasible flow in the capacities. Thus $f(e)/d_i \le c(e)/d_i$ and since $y(f) \ge 0$ for each $f \in \mathcal{F}_i$ we have $\sum_{f \in \mathcal{F}_i} y(f)f(e)/d_i \le c(e) \sum_{f \in \mathcal{F}_i} y(f)$ which implies that $f_{i,e} d_i \le f_i c(e)$. Thus the solution also satisfies (5) in Figure 3. This finishes the proof of the equivalence.

Hence, the results in Section 2 that lead to Theorem 2.6 and Corollary 2.7 also apply to a randomized rounding approach based on the compact formulation, as we explain in Section 5.

4 MWU ALGORITHM

While the compact edge-flow formulation can always be solved in polynomial time, one may run into space issues when attempting to solve it in practice: The disadvantage of using a standard LP solver to solve the compact edge-flow LP relaxation is that the number of variables is km which is quadratic in the input size, and the number of constraints is m. Standard LP solvers often require space proportional to km^2 which can be prohibitive even for moderate instances (since it is almost cubic in input size). One advantage of the packing LP formulation, although it is equivalent, to the compact formulation is that one can use well-known multiplicative weight update (MWU) based Lagrangean relaxation approaches to obtain a $(1-\gamma)$-approximation, for any $0 < \gamma < 1$. Although the convergence time can be slow depending on the accuracy required, the space requirement is $O(k+m)$ which is linear in the input size. In addition, there are several optimization heuristics based on the MWU algorithm that can result in very efficient implementations in practice. Since the MWU framework is standard we only describe and explain the algorithm here and state the known guarantees on the number of iterations and time complexity, referring the reader to standard treatments in the literature [4] for a formal analysis on the correctness guarantees.

$$(a) \max \sum_{i=1}^{k} w_i \sum_{f \in \mathcal{F}_i} y(f)$$
$$\sum_{i=1}^{k} \sum_{f \in \mathcal{F}_i} f(e)y(f) \le c(e) \qquad e \in E$$
$$y(f) \ge 0 \qquad f \in \mathcal{F}_i, 1 \le i \le k.$$

$$(b) \min \sum_{e \in E} c(e)\ell_e$$
$$\sum_{e \in E} f(e)\ell_e \ge w_i \qquad 1 \le i \le k, f \in \mathcal{F}_i$$
$$\ell(e) \ge 0 \qquad e \in E$$

Fig. 4. (a) LP Relaxation with no constraint on total amount routed per commodity; (b) its dual.

Algorithm description: MWU based algorithms are iterative and provide a way to obtain arbitrarily good relative approximation algorithms for a large class of linear programs such as packing, covering and mixed packing and covering LPs. In particular we can apply it to the packing LP in Fig 1b(b). The LP has two types of packing constraints, one involving the capacities, and the other involving the total amount of flow routed for each commodity. It is useful to simplify the LP further in order to apply a clean packing framework. For this purpose we alter the given graph $G = (V, E)$ as follows. For each given demand pair (s_i, t_i) we add a dummy source s'_i and connect it to s_i with an edge (s'_i, s_i) of capacity equal to d_i. We replace the pair (s_i, t_i) with the pair (s'_i, t_i), which ensures that the total amount of flow for the pair is at most d_i, and further allows us to eliminate the first set of constraints in Fig. 1b(b). In the modified instance we hence only have edge capacity constraints and the problem becomes a pure maximum throughput problem that

Algorithm 1: MWU for Multi-Commodity ANF Problem

Inputs: Directed graph G(V,E), $c : E \longrightarrow \mathbb{R}^+$, a set S of k pairs of commodities (s_i, t_i) each with demand d_i and $\gamma \in \mathbb{R}^+$

Change G by adding dummy terminal s_i' and edge(s_i', s_i) with capacity d_i. This ensures that we don't route more than d_i units for pair i. We will assume this has been done and simply use (s_i, t_i) instead of (s_i', t_i)

Output: Total flow f_e on each e. $f(s_i', s_i)/d_i$ gives the fraction of commodity i that is routed

Define a length/cost function $\ell : E \longrightarrow \mathbb{R}^+$ and initialize $\ell_e \longleftarrow 1, \forall e \in E$

Define a function $f : E \longrightarrow \mathbb{R}_{\geq 0}$ and initialize $f_e \longleftarrow 0, \forall e \in E$

Define $\eta \longleftarrow \frac{\ln |E|}{\gamma}$

repeat

  for each commodity $i \in S$ **do**

    Compute min-cost flow of d_i units from s_i to t_i with capacities $c(e)$ and cost given by ℓ. (If no feasible flow then pair i can be dropped.) Let this flow be defined by $g_i(e), e \in E$ and let cost of this flow be $\rho(i) = \sum_e \ell(e) g_i(e)$

  Set $i^* \longleftarrow \text{argmin}_{i \in S} \frac{\rho(i)}{w_i}$

  Compute $\delta \longleftarrow \min_e \frac{\gamma}{\eta} \cdot \frac{g_{i^*}(e)}{c(e)}$

  for each e **do**

    Set $f_e \longleftarrow f_e + \delta g_{i^*}(e)$

    if $f_e > c_e$ **then**

      Output f and halt

    else

      Update $\ell_e \longleftarrow \exp(\eta f_e / c_e)$

until termination

allows for a commodity to be routed more than one total unit. The dual LP also simplifies in a corresponding fashion. These are shown in Fig 4.

The MWU Algorithm 1 solves the primal LP in Fig 4 in an iterative fashion as follows. It takes as input an error parameter $\gamma \in (0, 1)$ and its goal is to output a feasible solution of value at least $(1 - \gamma)$ times the optimum LP solution value. Note that the primal LP has an exponential number of variables but only m non-trivial constraints corresponding to the edges, so it maintains only an implicit representation of the primal variables. The MWU algorithm can be viewed as primal-dual algorithm as well and as such it maintains "weights" (hence the name mutiplicative weights udate) for each edge e which correspond to the dual variables $\ell(e)$. To avoid confusion with the weights of commodities we use the term lengths. The algorithm maintains lengths $\ell(e), e \in E$ which are initialized to 1. The algorithm roughly maintains the invariant that $\ell(e)$ is exponential in the current total flow $g(e)$ on edge e; more formally, for a parameter $\eta = \ln m/\gamma$ the algorithm maintains the invariant that $\ell(e) \simeq \exp(\eta f(e)/c(e))$ where $f(e)$ is the total flow on e. In each iteration the goal is to find a good commodity/pair to route. To this end the algorithm computes for each commodity (s_i, t_i) a minimum-cost s_i-t_i flow of d_i units where the cost on e is equal to $\ell(e)$. Let this cost be $\rho(i)$. It then chooses the commodity i^* that has the smallest $\rho(i)/w_i$ ratio among all pairs, as the currently best commodity to route. The algorithm then routes a small amount for i^* along the minimum cost flow computed in that iteration. This corresponds to the step size δ which is chosen to be sufficiently small but not too small to ensure the correctness of the algorithm. After routing the flow for i^* the lengths on the edges are updated to reflect the increase in flow on the edges. The algorithm proceeds in this fashion for several iterations until termination. One can terminate using several different criteria while ensuring correctness. Here we stop the algorithm when we try to route a commodity with the given step size and realize that it violates some edge capacity.

Analysis of iterations, run-time and space: The algorithm's running time is dependent on the time to compute minimum-cost flow and on the total number of iterations. It is known that the MWU algorithm, as suggested above, terminates in $O(m \log m/\gamma^2)$ iterations. Each iteration requires computing k minimum-cost flows. Many algorithms are known for minimum-cost flow ranging from strongly polynomial-time algorithms to polynomial-time scaling algorithms as well as practically fast algorithms based on network-simplex. Instead of listing these we can upper bound the run-time by $O(\text{MCF}(n, m) km \log m/\gamma^2)$ where $\text{MCF}(n, m)$ is min-cost flow running time on a graph with n nodes and m edges. In terms of space we observe that the algorithm only maintains the total flow on each edge and for each commodity the total flow it has routed as well as the lengths on the edges. This is $O(k + m)$. The algorithm also needs space to compute minimum-cost flow and that depends on the algorithm used for it. Most algorithms for minimum-cost flow use space near-linear in the input graph.

The algorithm as described above is a plain "vanilla" implementation of the general MWU algorithm. As such the running time is rather high and computing k minimum-cost flows in each iteration is expensive. Several optimization can be done from both a theoretical and a practical point of view. We do not discuss these issues in detail since this is not the main focus of this paper. We develop a simple heuristic – the *permutation routing* heuristic – based on these ideas that has also theoretical justification, which will be discussed and used for the simulations in Section 8.

5 RANDOMIZED ROUNDING ALGORITHM

Algorithm 2 describes the randomized rounding algorithm that we will use in our simulations. This algorithm performs randomized rounding on the total flow variables of the compact LP and is therefore a special case of the randomized rounding algorithm outlined in Section 2.2 (since we have proven that the set of feasible solutions to the compact LP can be viewed as a subset of the feasible solutions to the packing LP). Algorithm 2 is also a generalization of the algorithm presented in [21] that accommodates arbitrary commodity demands and weights. This algorithm leads to a simpler, more streamlined implementation (also because the randomized rounding approach will be based on a number of variables that is linear in the number of commodities) than if we were using the approach based on the rounding of the variables of the packing LP directly. We assume, as we did in Section 2, that we discard any commodity i that cannot be routed by itself in G.

Algorithm 2: Randomized Rounding Algorithm

Input: Directed graph $G(V, E)$ with edge capacities $c_e > 0, \forall e \in E$; set of k pairs of commodities (s_i, t_i), each with demand $d_i \geq 0$ and weight $w_i \geq 0$; $\epsilon \in (0, 1]$

Output: The final values of f_i and $f_{i,e}$ and $\sum w_i f_i$

1 Let $\tilde{f}_i, \tilde{f}_{i,e}, \forall i \in [k], \forall e \in E$, be a feasible solution to compact LP.

2 For each $i \in [k]$, independently, set $f_i = 1$ with probability $\tilde{f}_i$, otherwise set $f_i = 0$.

3 Rescale the fractional flow $\tilde{f}_{i,e}$ from the LP solution on edge e for commodity i by $\frac{1}{\tilde{f}_i}$: I.e., $f_{i,e} = \frac{\tilde{f}_{i,e}}{\tilde{f}_i} \cdot f_i$ and the flow for commodity i on e is given by $f_{i,e} d_i$.

4 If $\sum_i w_i f_i \geq (1 - \epsilon) \sum w_i \tilde{f}_i$ and $\sum_i f_{i,e} d_i \leq (3b \ln m/\ln \ln m) c(e)$ for all $e \in E$, return the corresponding flow assignments given by f_i and $f_{i,e}, \forall i \in [k]$ and $e \in E$. Otherwise, repeat steps 2 and 3, $O((\ln m)/\epsilon^2)$ times.

We use randomized rounding to round the total fraction $\tilde{f}_i$ of d_i that the compact LP routes for commodity i to $f_i = 1$, with probability $\tilde{f}_i$, and to 0 otherwise. If we set f_i to 1, then in order to satisfy flow conservation constraints (e.g., constraint (3) of Figure 3), we need to re-scale all the $\tilde{f}_{i,e}$ values by $1/\tilde{f}_i$, obtaining the flows $f_{i,e}$ (if $f_i = 0$ then $f_{i,e} = 0$, for all $e \in E$). We repeat Steps 2-3 of Algorithm 2 $\Theta((\ln m)/\epsilon^2)$ times or until we obtain the desired $((1-\epsilon), 3b \ln m/\ln \ln m)$-approximation bounds, amplifying the probability of getting a desired outcome.

Given the equivalence that we showed between the packing and the compact LP, which implied among other things that the two LPS have optimal solutions of the same value and that Algorithm 2 corresponds to the packing randomized rounding approach described in Section 2 when restricted to the subset of solutions to the compact LP, we get the following corollary to Theorem 2.6:

Corollary 5.1. *Algorithm 2, when run on an optimum solution to the LP, achieves a* $((1 - \epsilon), 3b \ln m/\ln \ln m)$-approximation *for the ANF problem on* arbitrary networks *with high probability, for a suitable constant* $b > 1/m$, *e.g.* $b = 1.85$, *and any* $1/m < \epsilon < 1$.

In our implementations, we will also run Algorithm 2 using the solution output by the MWU algorithm, which only guarantees a $(1-\gamma)$ approximation on the throughput for $\gamma \in (0, 1)$.

In that case, we let $\tilde{f}_i = f(s_i', s_i)/d_i$, where $f(s_i', s_i)$ is as defined in Algorithm 1, and the values of $\tilde{f}_{i,e}$ are defined according to the flows chosen for each commodity i. Note that the throughput approximation guarantee for Algorithm 2 in this case will be $(1-\epsilon)(1-\gamma)$.

Another advantage of Algorithm 2 is that it leads to a surprisingly simple derandomized algorithm, as we will see in Section 6, that was also implemented for our simulations.

6 AN EFFICIENT DETERMINISTIC ALGORITHM

In this section, we give a derandomization of Algorithm 2. One of the main attractions of our derandomized algorithm is its simplicity and efficiency in practice (see Section 8), unlike almost all of the other derandomized algorithms in the literature, which turn out to be theoretical exercises since their implementations in practice are cumbersome and very ineffective. Our deterministic algorithm uses the method of pessimistic estimators first introduced by Raghavan [26] to efficiently compute conditional expectations, which will guide the construction of the (α, β)-approximate solution. Given the analysis in Section 2.2 and Corollary 5.1, in the forthcoming analysis, we always assume $\alpha = 1 - 1/m$ and $\beta = 3b \ln m/\ln \ln m$ for $m \geq 9$ and $b = 1.85$.

We first introduce the following notation. Let $z_i = 0$ if Algorithm 2 has not selected commodity i to be routed, and let $z_i = 1$ if i was admitted. Now, let $\mathsf{fail}(z_1, \ldots, z_k) \to \{0, 1\}$ denote the failure function of not constructing an (α, β)-approximate solution, i.e., $\mathsf{fail}(z_1, \ldots, z_k) = 1$ if and only if the constructed solution either does not achieve an α-fraction of the LP's (weighted) throughput or the capacity of some edge is exceeded by a factor larger than β. We use Z_i to denote the $\{0, 1\}$-indicator random variable for whether commodity i is routed or not in one execution of Steps 3-4 of Algorithm 2, i.e., $\mathsf{Pr}(Z_i = 1) = \tilde{f}_i$ and $\mathsf{Pr}(Z_i = 0) = 1 - \tilde{f}_i$. We have shown in Section 2.2 that $\mathsf{Ex}(\mathsf{fail}(Z_1, \ldots, Z_k)) < 1$ holds (cf. Theorem 2.6), implying the existence of an (α, β)-approximate solution. Given the above definitions, we employ the following notation to denote the conditional expectation of a function $\mathsf{f} : \{0, 1\}^k \to \{0, 1\}$:

$$\mathsf{Ex}(\mathsf{f}(z_1, \ldots, z_i, Z_{i+1}, \ldots, Z_k)) = \mathsf{Pr}(\mathsf{f}(Z_1, \ldots, Z_k) = 1 \mid Z_1 = z_1, \ldots, Z_i = z_i) \ .$$

As computing $\mathsf{Ex}(\mathsf{fail}(z_1, \ldots, z_i, Z_{i+1}, \ldots, Z_k))$ is generally computationally prohibitive, we will now derive a pessimistic estimator $\mathsf{est} : \{0, 1\}^k \to \mathbb{R}_{\geq 0}$, such that the following holds for all $i \in [k]$ and all $(z_1, \ldots, z_i) \in \{0, 1\}^i$:

Upper Bound $\mathsf{Ex}(\mathsf{fail}(z_1, \ldots, z_i, Z_{i+1}, \ldots, Z_k)) \leq \mathsf{Ex}(\mathsf{est}(z_1, \ldots, z_i, Z_{i+1}, \ldots, Z_k)).$ (8)

Algorithm 3: Deterministic Approximation for the All-or-Nothing Flow Problem

Input : Directed Graph $G(V, E)$
Source-Sink Pair (s_i, t_i) for each satisfiable commodity $i \in [k]$
Capacity $c(u, v)\ \forall (u, v) \in E$
Estimator $\mathsf{est}_\beta^\alpha : \{0,1\}^k \to \mathbb{R}_{\geq 0}$ for obtaining an (α, β)-approximate sol.
Output : (α, β)-approximate solution to the ANF instance

```
compute optimal solution f̃⃗ to compact edge-flow LP (cf. Figure 3)
let Z_i ∈ {0,1} be the random variable s.t. Pr(Z_i = 1) = f̃_i and Pr(Z_i = 0) = 1 − f̃_i for i ∈ [k]
compute failure_estimate ← Ex(est^α_β(Z_1, ..., Z_{i−1}, Z_i, ..., Z_n))
foreach i ∈ [k] do                                  // iterate over all commodities
    if Ex(est^α_β(z_1, ..., z_{i−1}, 0, Z_{i+1}, ..., Z_n)) < failure_estimate then
        set z_i ← 0                                 // commodity i is not routed
    else
        set z_i ← 1                                 // commodity i is routed
    update failure_estimate ← Ex(est^α_β(z_1, ..., z_i, Z_{i+1}, ..., Z_n))
return solution given by z⃗: if z_i = 1 then f_i = 1 and f_{i,e} = f̃_{i,e}/f̃_i, else f_i = f_{i,e} = 0, ∀i ∈ [k]
```

Efficiency $\quad \mathsf{Ex}\left(\mathsf{est}(z_1, \ldots, z_i, Z_{i+1}, \ldots, Z_k)\right)$ can be computed efficiently. (9)

Furthermore, the estimator's value must initially be strictly less than 1 for the derandomization:

Base Case $\quad \mathsf{Ex}\left(\mathsf{est}(Z_1, \ldots, Z_k)\right) < 1$ holds initially. (10)

In the following, we discuss how such a pessimistic estimator is used to derandomize the decisions the algorithm informally introduced in Section 2 before introducing the actual estimator $\mathsf{est}_\beta^\alpha$ in Lemma 6.1. Algorithm 3 first computes an LP solution just as Algorithm 2, but then uses the pessimistic estimator to guide its decision towards deterministically constructing an approximate solution. Specifically, each commodity is either routed or rejected such that the conditional expectation $\mathsf{Ex}\left(\mathsf{est}_\beta^\alpha(z_1, ..., z_i, Z_i, ..., Z_n)\right)$ is minimized. Given that initially $\mathsf{Ex}\left(\mathsf{est}_\beta^\alpha(Z_1, \ldots, Z_k)\right) < 1$, this procedure terminates with a solution $(z_1, \ldots, z_k)$ such that the failure function $\mathsf{fail}(z_1, \ldots, z_k)$ is strictly upper bounded by 1. Specifically, $1 > \mathsf{Ex}\left(\mathsf{est}_\beta^\alpha(Z_1, \ldots, Z_k)\right) \geq \mathsf{Ex}\left(\mathsf{est}_\beta^\alpha(z_1, Z_2, \ldots, Z_k)\right) \geq \ldots \geq \mathsf{Ex}\left(\mathsf{est}_\beta^\alpha(z_1, \ldots, z_k)\right)$ is guaranteed and therefore, for the binary function fail, $\mathsf{fail}(z_1, \ldots, z_k) = 0$ must hold. Furthermore, the algorithm is efficient (i.e., runs in polynomial time) as long as the pessimistic estimator function $\mathsf{est}_\beta^\alpha$ can be evaluated in polynomial time. W.l.o.g., we assume that only commodities that can be satisified in G are given as input to Algorithm 3.

We now introduce the following specific pessimistic estimator $\mathsf{est}_\beta^\alpha$ for which the above three correctness criteria (upper bound, efficiency, base case) are proven in Appendix C.

Lemma 6.1 (Pessimistic Estimator). *The function* $\mathsf{est}_\beta^\alpha$ *is a pessimistic estimator for the ANF:*

$$\mathsf{est}_\beta^\alpha(Z_1, \ldots, Z_k) = \mathsf{est}_\alpha(Z_1, \ldots, Z_k) + \sum_{(u,v) \in E} \mathsf{est}_\beta^{(u,v)}(Z_1, \ldots, Z_k), \text{ where}$$

$$\mathsf{est}_\alpha(Z_1, \ldots, Z_k) = e^{-\theta_\alpha (1-\delta_\alpha)\tilde{\mu}} \prod_{l \in [k]} \mathsf{Ex}\left(e^{\theta_\alpha Z_i \frac{w_i}{w_{\max}}}\right), \text{ with } \delta_\alpha = \frac{1}{m}, \ \tilde{\mu} = \frac{w_{LP}}{w_{\max}}, \ \theta_\alpha = \ln(1 - \delta_\alpha);$$

and $\text{est}_\beta^{(u,v)}(Z_1,\ldots,Z_k) = e^{-\theta_\beta(1+\delta_\beta)\hat{\mu}} \prod_{l\in[k]} \text{Ex}\left(e^{\theta_\beta Z_i \frac{(f_{i,(u,v)}/\tilde{f}_i)}{c_{(u,v)}}}\right)$, *with* $\delta_\beta = \dfrac{3b\ln m}{\ln\ln m - 1}$, $b = 1.85$, $\hat{\mu} = 1,\ \theta_\beta = \ln(1+\delta_\beta)$.

Given the above outlined intuition of the derandomization process and the correctness of the pessimistic estimator due to Lemma 6.1, the following main theorem of this section is obtained.

Theorem 6.2. *Using* est_β^α *as a pessimistic estimator, Algorithm 3 is a deterministic* (α,β)*-approximation for the ANF problem with* $\alpha = 1 - 1/m$ *and* $\beta = 3b\ln m/\ln\ln m$, *with* $b = 1.85$, *for* $m \geq 9$.

7 POTENTIAL PROBLEM EXTENSIONS

The packing formulation for the ANF was introduced in Section 2 together with a simple randomized rounding algorithm. Besides the practical tractability established in Section 4, the proposed packing framework for the ANF has further advantages. Specifically, it can be easily adapted to cater for problem extension such as when flows are restricted to k-splittable flows, must obey fault-tolerance criteria, or are restricted to shortest paths.

In the following we describe some of these extensions and how the packing formulation may be adapted together with the separation procedure. Notably, some problem extension allow for compact LP formulations, however, casting the problems in terms of the packing formulation is generally less complex and therefore helps in establishing whether a problem extension can be efficiently approximated in the first place.

Henceforth, our goal is to solve the maximum throughput problem in the all-or-nothing model while restricting the nature of flows that are allowed for each commodity. While the ANF allows flow for each commodity to be split in arbitrary ways while unsplittable flow requires all the flow for a commodity to use a single path. However, there are several intermediate settings of interest, and other constraints, that occur in practice. Recall that in setting up the formulation in Figure 1b, $\mathcal{F}_i$ for each pair i is the set of valid s_i-t_i flows in G. This is a large implicit set, and the way we solve the LP relaxation is via the separation oracle. The separation oracle corresponds to finding a minimum-cost flow from $\mathcal{F}_i$ given some edge lengths/costs. The MWU algorithm can be viewed as an efficient, albeit approximate, way to solve the large implicit LP relaxation via the separation oracle. Moreover, once the LP is solved, the randomized rounding step picks one of the flows per commodity. This flexibility allows us to solve the LP and round even when $\mathcal{F}_i$ is restricted in some fashion. We outline a few extensions that can be addressed via this framework.

Integer flows: Recall that in ANF we allow splittable flows. However in some settings it is useful to have flow for each commodity on each edge to be integer valued; here we assume that d_i is an integer for each i. In order to handle this we can set $\mathcal{F}_i$ to be the set of all integer s_i-t_i flows. Now the min-cost flow routine needs to find an integer flow between s_i and t_i of d_i units. This is easy to ensure since there always exists an integer valued min-cost flow as long as demand is integer valued and capacities are integer valued. We reduce each $c(e)$ to $\lfloor c(e) \rfloor$ without loss of generality.

Splitting into a small number of paths: In some applications it is important that the flow for each pair is not split by too much. How do we quantify this? One way is to consider h-splittable flows where h is an integer parameter. This means that flow for each pair can be decomposed into at most h paths. When $h = 1$ we obtain unsplittable flow and if we set $h = |E|$ we obtain ANF. We can handle the special case where each of the h flow paths has to be used to send the same amount of flow which is d_i/h. For this purpose we define $\mathcal{F}_i$ to be the set of all such flows. To compute a min-cost flow in $\mathcal{F}_i$ we simply need to find a min-cost flow of h units from s_i to t_i in the graph with capacities adjusted as follows: for each edge e with capacity $c(e)$ we change it to $\lfloor hc(e)/d_i \rfloor$. Baeier et al. [5] considered this maximum throughput problem, however, they only considered uni-criteria approximation algorithms and provided a reduction to the unsplittable case; the approximation ratios that one can obtain without violating capacities are very poor while our focus here is on bicriteria approximation that achieve close to optimum throughput.

Fault-tolerance and routing along disjoint paths: In some settings the flow for a pair (s_i, t_i) needs to be fault-tolerant to edge and/or node failures. There are several ways this is handled in the networking literature. One common approach is to send the flow for each commodity along h disjoint paths, each carrying d_i units. This can be handled by an approach very similar to the preceding paragraph where we compute min-cost flow on h disjoint paths; note that in the preceding paragraph the h paths could share edges. Another approach to fault-tolerance is to use what are called h-route flows [1, 20]. One can find a min-cost h-route flow in polynomial time [1]. Hence, one can also use the framework to maximize throughput while each routed pair uses h-route flow.

Using few edges or short paths: We now consider the setting when the flow for a commodity is required to use small number of edges or flow has to be routed along paths with small number of hops. These constraints not only arise in practice but also help improve the theoretical bounds on congestion. One can show that if each flow uses only d edges then the bicriteria approximation can be improved; the congestion required for a constant factor approximation becomes $O(\log d/\log\log d)$ rather than $O(\log m/\log\log m)$; for single paths the analysis can be seen from [6] and we can generalize it to our setting. Suppose we wish to route flow for each commodity whose support consists only of some given number h of edges. As above we need to solve for min-cost s_i-t_i flow that satisfies this extra constraint. However this additional constraint is no longer so easy to solve and in some cases can be NP-Hard. However, if one allows for a constant factor relaxation for the number of edges h, and an additional constant factor in the edge congestion one can address this more complex constraint by using linear programming based ideas (see [8] for an example).

8 SIMULATION RESULTS

In this section we study the performance of our approximation algorithms for the ANF problem on real-world networks. Our proof-of-concept computational evaluation is meant to provide general guidelines about the relative efficacy of the algorithms in terms of the *achieved throughput approximation factor* α and the *edge capacity violation ratio* β. The achieved throughput approximation ratio is taken as the solution obtained by the run divided by the optimal LP solution (which is a lower bound on the exact approximation ratio based on the optimal IP solution rather that its LP relaxation's). Notably, due to the bi-criteria nature of our approximations with solutions being allowed to exceed edge capacities (by at most a factor of β), solutions may yield empirical throughput approximation factors of $\alpha > 1$.

Beyond analyzing the performance of our randomized rounding and derandomized algorithms, we also investigate the impact of varying the methodology by which the LP is solved. Specifically, we study the performance of solving the compact LP formulation directly, of the multiplicative weight update algorithm (MWU), and of the MWU-based Permutation Routing (PR) heuristic described below. While the runtime of our prototypical MWU implementation generally exceeds the runtime of solving the compact LP formulation using a commercial solver, our MWU implementation serves as a proof of concept of its practical applicability and will also enable the extensions outlined in Section 7, which depend on the packing formulation. In addition, we remark that MWU may be useful for larger networks in practice (larger than the ones considered here), as it does not suffer from the same space complexity limitations as solving the compact LP via standard LP solvers.

Note that the simulation results for the current state-of-the-art algorithm for constant-throughput approximations for the ANF problem [21] — adapted here to handle non-uniform demands, edge capacities and weights — have been reproduced in this paper when running the randomized rounding algorithm with the compact edge-flow LP, since this algorithm is in essence the same as the algorithm in [21] (albeit some fine tuning optimizations). Our theoretical approximation results in this paper actually also validate the experimental results in [21], since the simulations in [21] already suggested that the edge capacity violations incurred by randomized rounding based on the compact edge-flow LP were logarithmic (and not polynomial as the theory of [21] suggested).

8.1 Permutation Routing Heuristic

Without proper optimization, the runtime of the MWU algorithm can be slow due to the computation of k minimum cost flows as a separate procedure in each iteration. As a practical solution, we introduce a heuristic based on Algorithm 1 that provides a significant reduction in computational cost, while in practice it still yields comparable solutions. We refer to this as the *Permutation Routing (PR)* algorithm. In the following we outline how this new algorithm differs from the original MWU algorithm and we refer the reader to Appendix D for the complete pseudocode description.

Our algorithm is motivated by theoretical algorithms for maximum throughput packing problems in the *online* arrival model and the *random arrival order* models. It is known that, for packing problems, in the random arrival model, one can obtain arbitrarily good performance compared to the offline optimal solution when the resource requirements of the arriving items (these correspond to flows in our setting) are sufficiently small when compared to the capacities [2, 17, 19]. The analytical ideas are related to online learning and MWU.

We develop our heuristic as follows: Recall that we are seeking a fractional solution. We take each commodity pair i with demand d_i and split it into r "copies", each with a demand of d_i/r. Here r is a sufficiently large parameter to ensure the property that d_i/r is "small" compared to the capacities. From the MWU analysis, and also the analysis in random arrival order models, one sees that $r = \Omega(\ln m/\gamma^2)$ suffices. Given the k original pairs, we create $k \cdot r$ total pairs from the copies. We now randomly permute these pairs and consider them by one-by-one. When considering a pair, the algorithm evaluates the "goodness" of the pair in a fashion very similar to that of the MWU algorithm. It maintains a length for each edge that is exponential in its current loads, and computes a minimum cost flow for the current pair (note that the pair's demand is only a $1/r$ fraction of its original demand); it accepts this pair if the cost of the flow is favorable compared to an estimate of the optimum solution. If it accepts the pair, it routes its entire demand (which is the $1/r$'th fraction of the original demand). Otherwise this pair is rejected and never considered again. Thus the total number of minimum cost flow computations is $O(k \cdot r)$ when compared to $O(k \cdot m \cdot \log m/\gamma^2)$ in the MWU algorithm. As mentioned above, a worst-case theoretical analysis requires $r = \Omega(\log m/\gamma^2)$ to guarantee a $(1 - \gamma)$-approximation, however, in practice a smaller value of r can be chosen. Note that an original pair (s_i, t_i) with demand d_i is routed to a fraction r_i/r where r_i is the number of copies of i that are admitted by the random permutation algorithm. The algorithm requires an estimate of the optimum solution which can be obtained via binary search or other methods.

8.2 Methodology

We now describe the problem instances and the implementations of our approximation algorithms.

Problem Instances. Following [22], we study real-world networks together with corresponding real-world source-sink pairs obtained from the survivable network design library (SNDlib) [25]. We randomly perturb the uniform weights, demands and edge capacities of the chosen networks to test our algorithms' ability to accommodate variable weights and demands on networks with varying edge capacities. Due to this choice, we find that only a fraction of the given commodities can be routed. Moreover, the choice ensures that few (if any) commodities can be fully routed through a single path without over-saturating the network, but still allows for a non-trivial fraction of the flows to be routed. Our choice of networks from the SNDlib is given in Table 1, covering several general scenarios, e.g. a small network with large number of commodities, or a dense network with large number of commodities. We chose independent uniform random network capacities from 20 to 60, commodity demands from 25 to 75, and commodity weights from 1 to 10 (the benchmark SNDlib data has all edge capacities at 40, demands at 50, and weights at 1).

Network	Vertices	Edges	Commodities	General Description
Atlanta	15	44	210	Small network, high commodity count
Germany50	50	176	662	Sparse network, high commodity count
Di-yuan	11	84	22	Dense network, low commodity count
Dfn-gwin	11	94	110	Dense network, high commodity count

Table 1. List of studied adapted instances from SNDlib [25]

Algorithms. We have implemented both the randomized and derandomized rounding algorithms detailed in Sections 5 and 6. We solve the compact formulation via CPLEX V12.10.0 and approximately solve the packing LP via the MWU algorithm or the faster permutation routing heuristic. We choose $\epsilon = \frac{1}{9}$ and $b = 1.85$ in Algorithms 2 and 3, implying a target throughput approximation factor of $\alpha \geq 1-\epsilon = \frac{8}{9}$ and target edge capacity violation ratio of $\beta \leq 3b \ln m / \ln \ln m = 5.5 \ln m / \ln \ln m$, where m is the number of network edges, for the algorithms. More specifically, for the Atlanta and Germany50 networks, we target edge capacity violations $\beta \leq 15.78$ and 17.47, respectively.

We define an experiment as the execution of a higher level algorithm (either randomized or derandomized rounding) in concert with an LP-solving subroutine (CPLEX for compact LP or our MWU and PR implementations) on a particular network. For an experiment that includes randomized rounding, we execute this algorithm 10 times to obtain a total of 10 different samples per experiment. For each of these 10 executions, 100 rounds of rounding are recorded and of these rounded solutions, we report on the solution of highest throughput whose capacity violations lie below our theoretical bounds. We consider three different γ values, namely 0.15, 0.2, and 0.3, to study performance vs. runtime trade offs of the MWU algorithm and the PR heuristic. Due to the slow convergence of MWU, we introduce speed-up mechanisms where (i) during any iteration, if the post-update smallest mincost flow solution is not at least 50 percent larger than the pre-update smallest mincost flow solution, then we do not recompute this in the subsequent iteration, and (ii) the maximum number of iterations is capped at 10k.

8.3 Experimental Results

In this section we report on our computational results. For the sake of brevity, we focus our attention on the performance of the Atlanta and Germany50 networks and defer the results of the other two smaller networks to Appendix E. We report results in terms of the achieved throughput factor α, edge capacity violation factor β, and the wall-clock running times.

The results of all experiments on these networks are summarized visually in Figure 5, and we will refer to this figure for the remainder of this section. The qualitative plots at the top and the middle show the empirical throughput and edge capacity violation ratios obtained by executions of the various algorithms. Note that we report on 10 data points when applying randomized rounding in contrast to the single data point for the derandomized algorithm. For reference, we include a red star data point indicating the optimal LP solution.

We see for both the Atlanta and Germany50 networks that the compact LP solver combined with both the randomized and derandomized algorithms produces β values that are well within our established theoretical bounds, although we see noticeably larger values of α and marginally larger values of β when the derandomized rounding is employed. The values of α and β obtained from

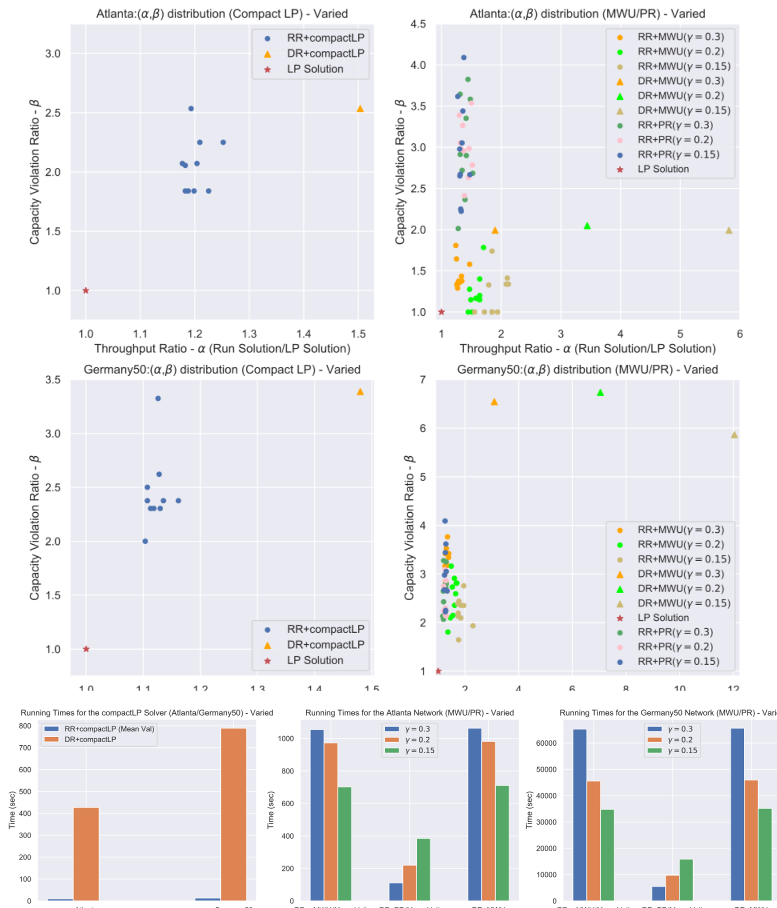

Fig. 5. Experimental results on the Atlanta (top) and Germany50 (middle) networks and their runtimes (bottom); RR refers to the randomized and DR to the derandomized rounding algorithm. Network capacities vary from 20 to 60, commodity demands vary from 25 to 75, and commodity weights vary from 1 to 10.

MWU are similarly concentrated around their means. Interestingly, in combination with the MWU algorithm, the deterministic rounding shows for Germany50 a significant increase in edge capacity violations while also achieving a much higher throughput. For the deterministic rounding of MWU, we observe that α increases as γ decreases under roughly constant capacity violations. With respect to the permutation routing subroutines, we observe more variance over the parameter space, and we typically see much higher capacity violation without a significant gain in throughput.

Regarding the runtimes, we remark that the compact LP solver is in general significantly faster than our MWU or PR implementations. Furthermore, the randomized rounding algorithm significantly outperforms our efficient deterministic rounding algorithm. We believe this to be mainly due to our naive implementation of the pessimistic estimators, that does not cache intermediate results.

Regarding the performance of the permutation routing algorithm, we note the following: For both networks we see a runtime decrease of a factor of at least half compared to the vanilla MWU algorithm while slightly compromising on the generally less favorable higher capacity violations. For the MWU algorithm, we expect in general to see the runtime increase as γ decreases, however, due to our speed-up mechanism, the opposite may be true. This is attributed to the fact that with a smaller γ, the increase in flow is likewise smaller, and thus the updates to edge costs are smaller, implying the threshold for skipping mincost flow calculations is met more often. Thus runtime is reduced for smaller values of γ, though at the expense of better throughput approximations.

Concluding, we note that we see our results as a first step towards efficiently approximating the ANF and its potential extensions. While randomized rounding wins in terms of runtime, the deterministic rounding generally achieves slightly higher throughputs. Furthermore, while solving the compact LP is shown to be much quicker in practice, the proposed MWU algorithm will render tackling the problem extensions of Section 7 tractable and our proposed permutation routing heuristic can in practice substantially reduce runtimes.

9 CONCLUSION

We presented a novel and significantly improved approximation of the maximum throughput routing problem for all-or-nothing multiple commodities with arbitrary demands. To this end, we derived formal bi-criteria approximation bounds and presented a proof of concept on efficient implementations of our algorithms in practice. We also showed that our packing framework is very flexible and may hence be of interest beyond the specific model considered in this paper, and apply, e.g., to scenarios where flows should only be split into a small number of paths or use few edges.

In future research, it would be interesting to develop improved rounding approaches, e.g., using resampling ideas from Lovasz-Local-Lemma work, to explore the additional applications introduced by our packing framework, as well as to study opportunities for algorithm engineering, further improving the performance of our algorithms in practice.

REFERENCES

[1] Charu C. Aggarwal and James B. Orlin. On multiroute maximum flows in networks. *Networks*, 39(1):43–52, 2002. URL: http://dx.doi.org/10.1002/net.10008, doi:10.1002/net.10008.

[2] Shipra Agrawal and Nikhil R Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1405–1424. SIAM, 2014.

[3] 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.

[4] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[5] Georg Baier, Ekkehard Köhler, and Martin Skutella. The k-splittable flow problem. *Algorithmica*, 42(3):231–248, 2005.

[6] Amit Chakrabarti, Chandra Chekuri, Anupam Gupta, and Amit Kumar. Approximation algorithms for the unsplittable flow problem. *Algorithmica*, 47(1):53–78, 2007.

[7] Chandra Chekuri and Alina Ene. The all-or-nothing flow problem in directed graphs with symmetric demand pairs. *Math. Program.*, 154(1-2):249–272, 2015. doi:10.1007/s10107-014-0856-z.

[8] Chandra Chekuri and Mark Idleman. Congestion minimization for multipath routing via multiroute flows. In *1st Symposium on Simplicity in Algorithms (SOSA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[9] Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. The all-or-nothing multicommodity flow problem. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '04, page 156–165, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/1007352.1007383.

[10] Chandra Chekuri, Sanjeev Khanna, and F Bruce Shepherd. Multicommodity flow, well-linked terminals, and routing problems. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 183–192, 2005.

[11] Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. Multicommodity flow, well-linked terminals, and routing problems. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 183–192. ACM, 2005. doi:10.1145/1060590.1060618.

[12] Chandra Chekuri, Sanjeev Khanna, and F Bruce Shepherd. The all-or-nothing multicommodity flow problem. *SIAM Journal on Computing*, 42(4):1467–1493, 2013.

[13] Julia Chuzhoy, Venkatesan Guruswami, Sanjeev Khanna, and Kunal Talwar. Hardness of routing with congestion in directed graphs. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 165–178, 2007.

[14] Julia Chuzhoy, David HK Kim, and Rachit Nimavat. Almost polynomial hardness of node-disjoint paths in grids. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1220–1233, 2018.

[15] Thomas. Erlebach and Klaus. Jansen. The maximum edge-disjoint paths problem in bidirected trees. *SIAM Journal on Discrete Mathematics*, 14(3):326–355, 2001. arXiv:https://doi.org/10.1137/S0895480199361259, doi:10.1137/S0895480199361259.

[16] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1):3–20, 1997.

[17] Anupam Gupta and Marco Molinaro. How the experts algorithm can help solve lps online. *Mathematics of Operations Research*, 41(4):1404–1431, 2016.

[18] Venkatesan Guruswami, Sanjeev Khanna, Rajmohan Rajaraman, Bruce Shepherd, and Mihalis Yannakakis. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. *Journal of Computer and System Sciences*, 67(3):473 – 496, 2003. URL: http://www.sciencedirect.com/science/article/pii/S0022000003000667, doi:https://doi.org/10.1016/S0022-0000(03)00066-7.

[19] Thomas Kesselheim, Klaus Radke, Andreas Tonnis, and Berthold Vocking. Primal beats dual on online packing lps in the random-order model. *SIAM Journal on Computing*, 47(5):1939–1964, 2018.

[20] Wataru Kishimoto. A method for obtaining the maximum multiroute flows in a network. *Networks: An International Journal*, 27(4):279–291, 1996.

[21] Mengxue Liu, Andrea Richa, Matthias Rost, and Stefan Schmid. A constant approximation for maximum throughput multicommodity routing and its application to delay-tolerant network scheduling. In *INFOCOM 2019 - IEEE Conference on Computer Communications*, Proceedings - IEEE INFOCOM, pages 46–54. Institute of Electrical and Electronics Engineers Inc., 4 2019. doi:10.1109/INFOCOM.2019.8737402.

[22] Mengxue Liu, Andréa W. Richa, Matthias Rost, and Stefan Schmid. A constant approximation for maximum throughput multicommodity routing and its application to delay-tolerant network scheduling. In *2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*, pages 46–54. IEEE, 2019. doi:10.1109/INFOCOM.2019.8737402.

[23] Jeffrey C Mogul and Lucian Popa. What we talk about when we talk about cloud network performance. *ACM SIGCOMM Computer Communication Review*, 42(5):44–48, 2012.

[24] Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms*. Chapman & Hall/CRC, 2010.

[25] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski. Sndlib 1.0—survivable network design library. *Netw.*, 55(3):276–286, May 2010.

[26] Prabhakar Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *Journal of Computer and System Sciences*, 37(2):130 – 143, 1988. URL: http://www.sciencedirect.com/science/article/pii/0022000088900037, doi:https://doi.org/10.1016/0022-0000(88)90003-7.

[27] Prabhakar Raghavan and Clark D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, December 1987. URL: http://dx.doi.org/10.1007/BF02579324, doi:10.1007/BF02579324.

[28] M. Rost and S. Schmid. Virtual network embedding approximations: Leveraging randomized rounding. In *2018 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 1–9, May 2018. doi:10.23919/IFIPNetworking.2018.8696623.

[29] Matthias Rost, Elias Döhne, and Stefan Schmid. Parametrized complexity of virtual network embeddings: Dynamic & linear programming approximations. *ACM SIGCOMM Computer Communication Review*, 49:3–10, 02 2019. doi:10.1145/3314212.3314214.

[30] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C Snoeren. Inside the social network's (datacenter) network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pages 123–137, 2015.

[31] F. Bruce Shepherd and Adrian Vetta. The inapproximability of maximum single-sink unsplittable, priority and confluent flow problems. *CoRR*, abs/1504.00627, 2015. URL: http://arxiv.org/abs/1504.00627, arXiv:1504.00627.

A ALTERATION APPROACH

We describe an alternative rounding approach to the one presented in Section 2 that has some advantages in certain settings and gives a better tradeoff in terms of repetitions. The algorithm consists of two phases. In the first phase, for each pair i we randomly pick at most one flow as described above (note that we may not pick any flow in which case we think of it as an empty flow).

For pair i, let g_i be the chosen flow. The second phase is an alteration phase to ensure that the constraints are not violated more than the desired amount. We order the pairs arbitrarily (without loss of generality from 1 to k) and consider them one by one. When considering pair i we try to route i via flow g_i if it is not empty. If adding the g_i to the already chosen flows for pairs 1 to $i-1$ does not violate any edge capacity by more than a factor $(1 + 3b \log m/\log\log m)$, we add i to the routed pairs, otherwise we discard pair i. Note that a pair i that was chosen in the first phase may get discarded in this second *alteration* step. Let S be the random set of routed pairs. From the construction it is clear that we can route pairs in S without violating any edge's capacity by a factor larger than $(1 + 3b \log m/\log\log m)$. Note that unlike the basic randomized rounding algorithm we have a deterministic guarantee on this property. Now we lower bound the expected weight of S.

Lemma A.1. *Let S be set of pairs routed at the end of the alteration phase. Then $E[w(S)] \geq (1 - 1/m^{\Omega(b)})W_{LP}$. Moreover, if y is a fractional solution sucht that $W_{LP} \geq cOPT_{LP}$, for some $c \leq 1$ then with probability at least $\frac{\epsilon - m^{-\Omega(b)}}{c(1+3b\ln m/\ln m)}$, $w(S) \geq (1-\epsilon)W_{LP}$.*

Proof. Consider a pair i. Let Y_i be a binary random variable that is 1 if a a non-empty flow is chosen in the random rounding stage. Let Z_i be binary random variable that is 1 if $i \in S$, that is, if i is routed after the alteration phase. We have $w(S) = \sum_i w_i Z_i$ and hence by linearity of expectation we have $E[w(S)] = \sum_i w_i \Pr[Z_i = 1]$.

We now lower bound $\Pr[Z_i = 1]$. We observe that $\Pr[Y_i = 1] = \sum_{f \in \mathcal{F}_i} y(f)$ by the random choice in the first step. We have $\Pr[Z_i = 1] = \Pr[Y_i = 1](1 - \Pr[Z_i = 0 \mid Y_i = 1])$. The quantity $\Pr[Z_i = 0 \mid Y_i = 1]$ is the probability that pair i is rejected in second phase of the algorithm conditioned on the event that it is chosen in the first stage. Pair i is rejected *only if* there is some edge e such that the total flow on e is more than $(3b \log m/\log\log m)c(e)$ from the flows chosen in the first step of the algorithm; from Lemma 2.5 this probability is at most $1/m^{\Omega(b)}$. Thus $\Pr[Z_i = 0 \mid Y_i = 1] \leq 1/m^{\Omega(b)}$ and hence $\Pr[Z_i = 1] \geq (1 - 1/m^{\Omega(b)}) \sum_{f \in \mathcal{F}_i} y(f)$. Since $E[w(S)] = \sum_i w_i \Pr[Z_i = 1]$, we see that $E[w(S)] \geq (1 - 1/m^{\Omega(b)})W_{\mathrm{LP}}$.

We now argue the second part. Let $\mathrm{OPT_{LP}}$ denote the value of an optimum solution to the LP relaxation. Consider the random variable $w(S)$. We claim that $w(S) \leq (1{+}3b \log m/\log\log m)\mathrm{OPT_{LP}}$ deterministically. To see this recall that S admits a routing that satisfies the capacity constraints to within a congestion of $(1 + 3b \log m/\log\log m)$. Therefore, by scaling down the routing of each demand in S by a $(1 + 3b \log m/\log\log m)$ factor, one obtains a feasible fractional solution to the LP relaxation whose value is at least $w(S)/(1 + 3b \log m/\log\log m)$. This implies that $\mathrm{OPT_{LP}} \geq w(S)/(1 + 3b \log m/\log\log m)$, which proves the claim. Thus, if $W_{\mathrm{LP}} \geq c\mathrm{OPT_{LP}}$ we have $E[w(S)] \geq (1 - 1/m^{\Omega(b)})W_{\mathrm{LP}} \geq c(1 - 1/m^{\Omega(b)})\mathrm{OPT_{LP}}$, and $w(S) \leq (1 + 3b \log m/\log\log m)\mathrm{OPT_{LP}}$. Let α be the probability that $w(S) < (1-\epsilon)W_{\mathrm{LP}}$. Then we have the following.

$$\begin{aligned} E[w(S)] &\leq (1-\alpha)(1 + 3b \log m/\ln\ln m)\mathrm{OPT_{LP}} + \alpha(1-\epsilon)W_{\mathrm{LP}} \\ &\leq (1-\alpha)(1 + 3b \log m/\ln\ln m)W_{\mathrm{LP}}/c + (1-\epsilon)W_{\mathrm{LP}} \end{aligned}$$

However $E[w(S)] \geq (1 - 1/m^{\Omega(b)})W_{\mathrm{LP}}$. Rearranging and simplifying we have

$$(1-\alpha) \geq c(\epsilon - m^{-\Omega(b)})/(1 + 3b \ln m/\ln\ln m).$$

Note that $(1-\alpha)$ is the probability that $w(S) \geq (1-\epsilon)W_{\mathrm{LP}}$. This finishes the proof. □

From the preceding lemma we see that the alteration algorithm guarantees deterministically that the congestion is $O(b \ln m/\ln\ln m)$ while the expected weight of the commodities routed by it is very close to W_{LP}. In fact with probability roughly $\Omega(\epsilon/(1 + b \ln m))$ the value of the routed pairs is at least $(1-\epsilon)W_{\mathrm{LP}}$ assuming that W_{LP} is a constant factor of $\mathrm{OPT_{LP}}$. In most settings we would want to start with a fractional solution that is very close to $\mathrm{OPT}_{\mathrm{LP}}$ and hence the assumption is reasonable. To guarantee a high probability bound[3] on achieving at least $(1-\epsilon)W_{\mathrm{LP}} \geq (1-\epsilon)\mathrm{OPT}_{\mathrm{IP}}$, it suffices to repeat the algorithm $O(\ln m/\epsilon^2)$ times.

B PROOFS OF CHERNOFF CONCENTRATION BOUNDS

In this appendix we prove extended versions of the classic Chernoff bounds presented in Theorem 2.3 to derandomize our approximation algorithm. The specific Chernoff extensions can be found in Appendix B.2, we first state some common convexity arguments below.

B.1 Some Convexity Arguments

Lemma B.1. *The following holds for any $\theta \in \mathbb{R}$ and $x \in [0,1]$:*

$$\exp(\theta \cdot x) \leq 1 + (\exp(\theta) - 1)$$

Lemma B.2. *The following holds for any $\theta \in \mathbb{R}$ and $x \in [0,1]$:*

$$1 + (\exp(\theta) - 1) \cdot x \leq \exp((\exp(\theta) - 1) \cdot x)$$

Lemma B.3. *Let $X_l \in [0,1]$ denote a single random variable of expectation $\mu_l = \mathsf{Ex}(X_l)$. For any $\theta \in \mathbb{R}$ the following holds for the random variable $Y_l = \exp(\theta \cdot X_l)$:*

$$\mathsf{Ex}(Y_l) \leq \exp((\exp(\theta) - 1) \cdot \mu_i) \tag{11}$$

Lemma B.4. *The following inequality holds for any $x \in [0,1]$:*

$$(1+x)\ln(1+x) - x \geq x^2/3\,. \tag{12}$$

Lemma B.5. *The following inequality holds for any $x \in [0,1)$:*

$$-x - (1-x)\ln(1-x) \leq -x^2/2\,. \tag{13}$$

Lemma B.6. *The following inequality holds for any $x > 0$:*

$$\ln\ln x - \ln\ln\ln x \geq 0.5 \cdot \ln\ln x\,. \tag{14}$$

B.2 Chernoff Bounds

In the following, Theorems B.7 and B.8 extend the classic Chernoff bounds of Theorem 2.3, enabling us to obtain pessimistic estimators for the sake of derandomization.

Theorem B.7. *Let X be the sum of k independent random variables $X_1, \ldots, X_k$ with $X_l \in [0,1]$ for $l \in [k]$. Denoting by $\tilde{\mu}_l \leq \mu_l = \mathsf{Ex}(X_\ell)$ lower bounds on the expected value of random variable X_l, $l \in [k]$, the following holds for any $\delta \in (0,1)$ with $\tilde{\mu} = \sum_{l \in [k]} \tilde{\mu}_l$ and $\theta = \ln(1-\delta)$;*

$$\Pr(X \leq (1-\delta)\cdot\tilde{\mu}) \overset{(a)}{\leq} e^{-\theta\cdot(1-\delta)\cdot\tilde{\mu}} \cdot \prod_{l\in[k]} \mathsf{Ex}\left(e^{\theta\cdot X_l}\right) \overset{(b)}{\leq} e^{-\delta^2\cdot\tilde{\mu}/2} \tag{15}$$

Proof. We first prove the inequality $\Pr(X \leq (1-\delta)\cdot\tilde{\mu}) \leq \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\tilde{\mu}}$ for $\delta \in (0,1)$. Let $Y_l = \exp(\theta \cdot X_l)$, $l \in [k]$, for $\theta = \ln(1-\delta)$. Note that $\theta < 0$ holds.

By Lemma B.3 Equality 11 holds. As $\exp(\theta) - 1 = -\delta < 0$ holds for $\theta < 0$, the exponential function $f(z) = \exp((\exp(\theta)-1)\cdot z)$ is monotonically decreasing. Using the lower bound $\tilde{\mu}_l \leq \mu_l$ and $\tilde{\mu} = \sum_{l\in[k]} \tilde{\mu}_l$ the following is obtained:

$$\mathsf{Ex}(Y_l) \leq \exp((\exp(\theta) - 1)\cdot\tilde{\mu}_l) \tag{16}$$

[3] We say that an event occurs with high probability, if it occurs with probability at least $1 - 1/x^c$, where x is the size of the input and $c > 0$ is a constant.

As the variables $X_1, \dots, X_k$ are pairwise independent, the variables $Y_1, \dots, Y_k$ are also pairwise independent. Accordingly, the following holds for $Y = e^{\theta \cdot X}$:

$$\mathsf{Ex}\,(Y) = \mathsf{Ex}\left(e^{\theta \cdot \sum_{l \in [k]} X_l}\right) = \mathsf{Ex}\left(\prod_{l \in [k]} e^{\theta \cdot X_l}\right) = \prod_{l \in [k]} \mathsf{Ex}\,(Y_l)\ . \tag{17}$$

Accordingly, the following is obtained:

$$\begin{aligned}
&\Pr[X \leq (1-\delta) \cdot \tilde{\mu}] && && (18)\\
&= \Pr[e^{\theta \cdot X} \geq e^{\theta \cdot (1-\delta) \cdot \tilde{\mu}}] && [\text{as } \theta < 0] && (19)\\
&\leq \frac{\mathsf{Ex}\,(\exp[\theta \cdot X])}{e^{\theta \cdot (1-\delta) \cdot \tilde{\mu}}} && [\text{by Markov's inequality}] && (20)\\
&= \frac{\prod_{l \in [k]} \mathsf{Ex}\,(Y_l)}{e^{\theta \cdot (1-\delta) \cdot \tilde{\mu}}} && [\text{by Equation 29}] && (21)\\
&\leq \frac{\prod_{l \in [k]} e^{(\exp(\theta)-1) \cdot \tilde{\mu}_i}}{e^{\theta \cdot (1-\delta) \cdot \tilde{\mu}}} && [\text{by Equation 16}] && (22)\\
&= e^{\left(\sum_{l \in [k]} e^{\theta} - 1 \cdot \tilde{\mu}_l\right) - \left(\theta \cdot (1-\delta) \cdot \tilde{\mu}\right)} && [\text{one exponent}] && (23)\\
&= e^{\left(\sum_{i \in [N]} ((1-\delta)-1) \cdot \tilde{\mu}_i\right) - \left(\ln(1-\delta) \cdot (1-\delta) \cdot \tilde{\mu}\right)} && [\text{using } \theta = \ln(1-\delta)] && (24)\\
&= e^{\left(-\delta \cdot \tilde{\mu}\right) - \left(\ln(1-\delta) \cdot (1-\delta) \cdot \tilde{\mu}\right)} && [\text{definition of } \tilde{\mu} = \textstyle\sum_{l \in [k]} \tilde{\mu}_l] && (25)\\
&= \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\tilde{\mu}} && && (26)
\end{aligned}$$

Given Equation 26, Inequality (b) is a corollary of Lemma B.5, which showed the following for $\delta \in (0, 1)$

$$-\delta - (1-\delta) \ln(1-\delta) \leq -\delta^2/2\,. \tag{27}$$

Multiplying both sides with $\tilde{\mu}$ and exponentiating both sides yields the desired result.

Regarding the Inequality (a), we note that this follows by the above proof from Equation 21. □

THEOREM B.8. *Let X be the sum of k random variables $X_1, \dots, X_k$ with $X_l \in [0, 1]$ for $l \in [k]$. Denoting by $\hat{\mu}_l \geq \mu_l = \mathsf{Ex}\,(X_\ell)$ upper bounds on the expected value of random variable X_l, $l \in [k]$, the following holds for any $\delta > 0$ with $\hat{\mu} = \sum_{l \in [k]} \hat{\mu}_l$ and $\theta = \ln(\delta + 1)$:*

$$\Pr\,(X \geq (1+\delta) \cdot \hat{\mu}) \overset{(a)}{\leq} e^{-\theta \cdot (1+\delta) \cdot \hat{\mu}} \cdot \prod_{l \in [k]} \mathsf{Ex}\left(e^{\theta \cdot X_l}\right) \overset{(b)}{\leq} \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\hat{\mu}}\ . \tag{28}$$

PROOF. Let $Y_i = \exp(\theta \cdot X_i)$ for $\theta = \ln(\delta + 1)$, $l \in [k]$. Note that $\theta > 0$ holds. As the variables $X_1, \dots, X_k$ are pairwise independent, the variables $Y_1, \dots, Y_k$ are also pairwise independent. Considering $Y_l = \exp(\theta \cdot X_l)$, the following holds due to the pairwise independence of the variables $Y_1, \dots, Y_k$:

$$\begin{aligned}
\mathsf{Ex}\,(Y_l) =& \mathsf{Ex}\,(\exp(\theta \cdot X)) = \mathsf{Ex}\left(e^{\theta \cdot \sum_{l \in [k]} X_l}\right)\\
=& \mathsf{Ex}\left(\prod_{l \in [k]} e^{\theta \cdot X_l}\right) = \prod_{l \in [k]} \mathsf{Ex}\,(Y_i)\ .
\end{aligned} \tag{29}$$

By Lemma B.3 the Inequality 11 holds. As $\theta > 0$ holds, the exponential function $f(z) = \exp((\exp(\theta) - 1) \cdot z)$ is monotonically increasing. Using the upper bound $\hat{\mu}_l \geq \mu_l = \mathsf{Ex}(X_l)$ on the expectation of X_l, $l \in [k]$, with $\hat{\mu} = \sum_{l \in [k]} \hat{\mu}_l$, the following is obtained:

$$\mathsf{Ex}(Y_l) \leq \exp((\exp(\theta) - 1) \cdot \hat{\mu}_l)\,. \tag{30}$$

Using Equations 29 and 30 together with Markov's inequality, the left inequality of the Chernoff bound 28 is obtained for any $\delta > 0$ by setting $\theta = \ln(\delta + 1) > 0$:

$$\Pr[X \geq (1+\delta) \cdot \hat{\mu}] \tag{31}$$

$$= \Pr[\exp(\theta \cdot X) \geq \exp(\theta \cdot (1+\delta) \cdot \hat{\mu})] \tag{32}$$

$$\leq \frac{\mathsf{Ex}(\exp(\theta \cdot X))}{\exp(\theta \cdot (1+\delta) \cdot \hat{\mu})} \qquad \text{[by Markov's inequality]} \tag{33}$$

$$= \frac{\prod_{i \in [N]} \mathsf{Ex}(Y_i)}{\exp(\theta \cdot (1+\delta) \cdot \hat{\mu})} \qquad \text{[by Equation 29]} \tag{34}$$

$$\leq \frac{\prod_{i \in [N]} \exp((\exp(\theta) - 1) \cdot \hat{\mu}_i)}{\exp(\theta \cdot (1+\delta) \cdot \hat{\mu})} \qquad \text{[by Equation 30]} \tag{35}$$

$$= \exp\left(\Big(\sum_{i \in [N]} (\exp(\theta) - 1) \cdot \hat{\mu}_i\Big) - \big(\theta \cdot (1+\delta) \cdot \hat{\mu}\big)\right) \tag{36}$$

$$= \exp\left(\Big(\sum_{i \in [N]} ((1+\delta) - 1) \cdot \hat{\mu}_i\Big) - \big(\ln(1+\delta) \cdot (1+\delta) \cdot \hat{\mu}\big)\right) \qquad \text{[using } \theta = \ln(\delta+1)\text{]} \tag{37}$$

$$= \exp\left(\big(\delta \cdot \hat{\mu}\big) - \big(\ln(1+\delta) \cdot (1+\delta) \cdot \hat{\mu}\big)\right) \qquad \text{[def. of } \hat{\mu} = \sum_{i \in [N]} \hat{\mu}_i\text{]} \tag{38}$$

$$= \frac{\exp\left(\delta \cdot \hat{\mu}\right)}{\exp\left(\ln(1+\delta) \cdot (1+\delta) \cdot \hat{\mu}\right)} = \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\hat{\mu}} \tag{39}$$

This completes the proof of inequality (b). Regarding the Inequality (a), we note that this follows by the above proof from Equation 34. □

C PROOF OF LEMMA 6.1

As in Section 6, in the following we always assume $\alpha = 1 - 1/m$ and $\beta = 3b \ln m / \ln\ln m$ for $m \geq 9$ and $b = 1.85$.

Lemma 6.1 (Pessimistic Estimator). *The function* $\mathsf{est}^{\alpha}_{\beta}$ *is a pessimistic estimator for the ANF:*

$$\mathsf{est}^{\alpha}_{\beta}(Z_1, \dots, Z_k) = \mathsf{est}_{\alpha}(Z_1, \dots, Z_k) + \sum_{(u,v) \in E} \mathsf{est}^{(u,v)}_{\beta}(Z_1, \dots, Z_k), \ \textit{where}$$

$$\mathsf{est}_{\alpha}(Z_1, \dots, Z_k) = e^{-\theta_\alpha (1-\delta_\alpha)\tilde{\mu}} \prod_{l \in [k]} \mathsf{Ex}\left(e^{\theta_\alpha Z_i \frac{w_i}{w_{\max}}}\right), \ \textit{with } \delta_\alpha = \frac{1}{m},\ \tilde{\mu} = \frac{w_{LP}}{w_{\max}},\ \theta_\alpha = \ln(1 - \delta_\alpha);$$

$$\textit{and } \mathsf{est}^{(u,v)}_{\beta}(Z_1, \dots, Z_k) = e^{-\theta_\beta (1+\delta_\beta)\hat{\mu}} \prod_{l \in [k]} \mathsf{Ex}\left(e^{\theta_\beta Z_i \frac{(f_{i,(u,v)}/\tilde{f}_i)}{c_{(u,v)}}}\right), \ \textit{with } \delta_\beta = \frac{3b \ln m}{\ln\ln m - 1},\ b = 1.85,$$

$$\hat{\mu} = 1,\ \theta_\beta = \ln(1 + \delta_\beta)\,.$$

Proof. The following three properties are to be shown: (i) upper bound, (ii) efficiency, and (iii) base case (cf. Equations 8 - 10). We first discuss properties (i) and (iii).

The analysis in Section 5 has demonstrated that the probability of obtaining an (α, β)-approximate solution via randomized rounding is bounded from below by $1/(6 \cdot m^2)$ (cf. Corollary 5.1). To obtain this result, a union bound argument was employed, which used probabilistic bounds on not achieving at least an α fraction of the optimal throughput and exceeding the capacity of each single edge by a factor of β.

For the throughput, the Chernoff bound of Theorem B.7 was applied, while for each edge's capacity violation, the Chernoff bound of Theorem B.8 was used. The pessimistic estimators est_α and $\mathrm{est}_\beta^{(u,v)}$ are a direct result of these respective theorems:

- est_α is obtained from the application of the Chernoff bound of Theorem B.7 within the proof of Lemma 2.4. Specifically, the application of the Chernoff bound in Lemma 2.4 yielded the following — restated over the variables Z_i — with the parameters δ_α, θ_α and $\tilde{\mu}$ as specified above:

$$\mathsf{Pr}\left(\sum_{l \in [k]} w_l \cdot Z_l < \alpha \cdot w_{LP}\right) \leq e^{-\theta_\alpha \cdot (1-\delta) \cdot \tilde{\mu}} \cdot \prod_{i \in [k]} \mathsf{Ex}\left(e^{\theta_\alpha \cdot Z_i \cdot w_i / w_{\max}}\right) \leq e^{-1/(2 \cdot m^2)}$$

  The middle expression directly yields the pessimistic estimator for the throughput.
- $\mathrm{est}_\beta^{(u,v)}$ is analogously obtained from the application of the Chernoff bound of Theorem B.8 in the Lemma 2.5 for each edge $(u, v) \in E$. Specifically, for a single edge (u, v), the following is obtained when using the constants defined above:

$$\mathsf{Pr}\left(\sum_{i \in [k]} f_{i,(u,v)} > \beta \cdot c(u,v)\right) \leq e^{-\theta_\beta \cdot (1+\delta_\beta) \cdot \hat{\mu}} \cdot \prod_{i} \mathsf{Ex}\left(e^{\theta_\beta \cdot Z_i \cdot f_{i,(u,v)} / c_{(u,v)}}\right) \leq 1/(6 \cdot m^2)$$

  Again, the middle expression is used to obtain the pessimistic estimator $\mathrm{est}_\beta^{(u,v)}$ for the specific edge $(u, v) \in E$.

Revisiting the union bound argument, we obtain that $\mathrm{est}_\beta^\alpha$ indeed yields an upper bound on the failure probability to construct an (α, β)-approximate solution, and that initially $\mathsf{Ex}\left(\mathrm{est}_\beta^\alpha(Z_1, \ldots, Z_k)\right) \leq 1 - 1/(6 \cdot m^2) < 1$ holds for $m \geq 9$. This shows that properties (i) and (iii) are satisfied.

Considering the efficiency property (ii), we note the following. Both est_α and $\mathrm{est}_\beta^{(u,v)}$ consist of products, where expectations for different commodities can be computed independently. Given the binary nature of the variables Z_i, these expectations can be computed in constant time. □

D PERMUTATION ROUTING PSEUDOCODE

Algorithm 5: Permutation Routing Algorithm

Inputs: $\gamma \in \mathbb{R}^+$, Directed Graph G(V,E), $c : E \longrightarrow \mathbb{R}^+$, a set S of k pairs of commodities (s_i, t_i) each with demand d_i, weight w_i, an estimate Est of the optimal fractional ANF solution for (G, S)

Initialize an empty flow $f(e) \leftarrow 0, \forall e \in E$
Set $\eta \leftarrow \frac{\ln |E|}{\gamma}$
Set $r \leftarrow \frac{\ln |E|}{\gamma^2}$
Let $f_{i,e} \leftarrow 0$ be the fractional flow assignment for commodity i on edges e for all $i \in S, e \in E$
Define edge costs $\ell(e) = 1, \forall e \in E$
Make r copies the k commodities of S and let A be a list of these rk commodities
Let B be a random permutation of A
for each commodity copy j in B **do**
    Let commodity j in B correspond to original demand (s_i, t_i)
    Compute min-cost flow of d_i units from s_i to t_i with edge costs defined by ℓ and obtain flow assignment f' and solution cost $\rho = \sum_{e \in E} \ell(e) f'(e)$
    Compute $\tau = \sum_{e \in E} \ell(e) c(e)$
    if $\frac{w_j}{\rho} \geq \frac{Est}{\tau}$ and none of the following updates cause an edge capacity violation **then**
        for each edge $e \in E$ **do**
            Update $f(e) \leftarrow f(e) + \frac{f'(e)}{r}$
            Update $f_{i,e} \leftarrow f_{i,e} + \frac{1}{r}$
            Update $\ell(e) \leftarrow \exp\left(\frac{\eta \cdot f(e)}{c(e)}\right)$
Return $f(e), f_{i,e}$ for all $i \in S, e \in E$

E ADDITIONAL EXPERIMENTAL RESULTS

In this section, we present supplemental experimental results. Figure 6 summarizes the results on networks Atlanta and Germany50 under the default uniform weights, demands and edge capacities given by [25]. This figure also replicates experiments from [21], though here we additionaly test Algorithms 1,5 in conjuction with Algorithms 2,3. In Figure 7, we present experimental results on two additional networks from [25], namely DFN-GWIN and DI-YUAN. In these experiments, we test uniform commodity weights, demands and edge capacities. Lastly, in Figure 8, we again present experimental results on DFN-GWIN and DI-YUAN, though we randomly perturb the weights, demands and edge capacities in the same manner given in Figure 5.

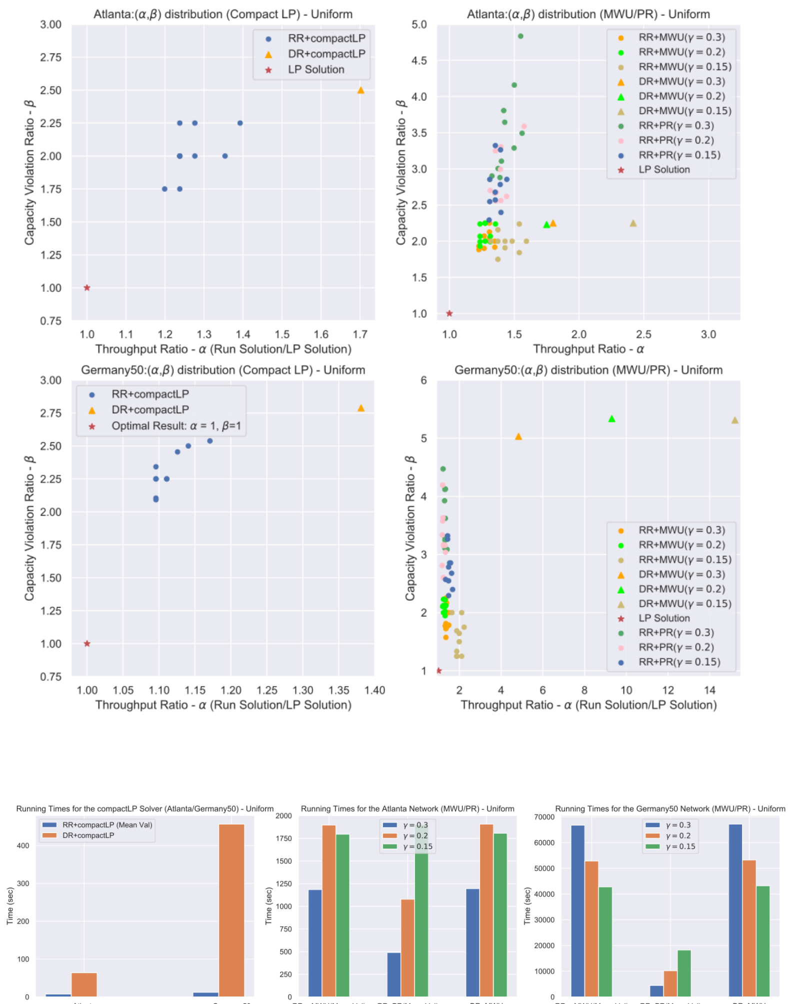

Fig. 6. Experimental results on the Atlanta (top) and Germany50 (middle) networks and their runtimes (bottom); RR refers to the randomized and DR to the derandomized rounding algorithm. Network capacities, commodity weights and commodity demands are fixed to 40,1 and 50, respectively per [25].

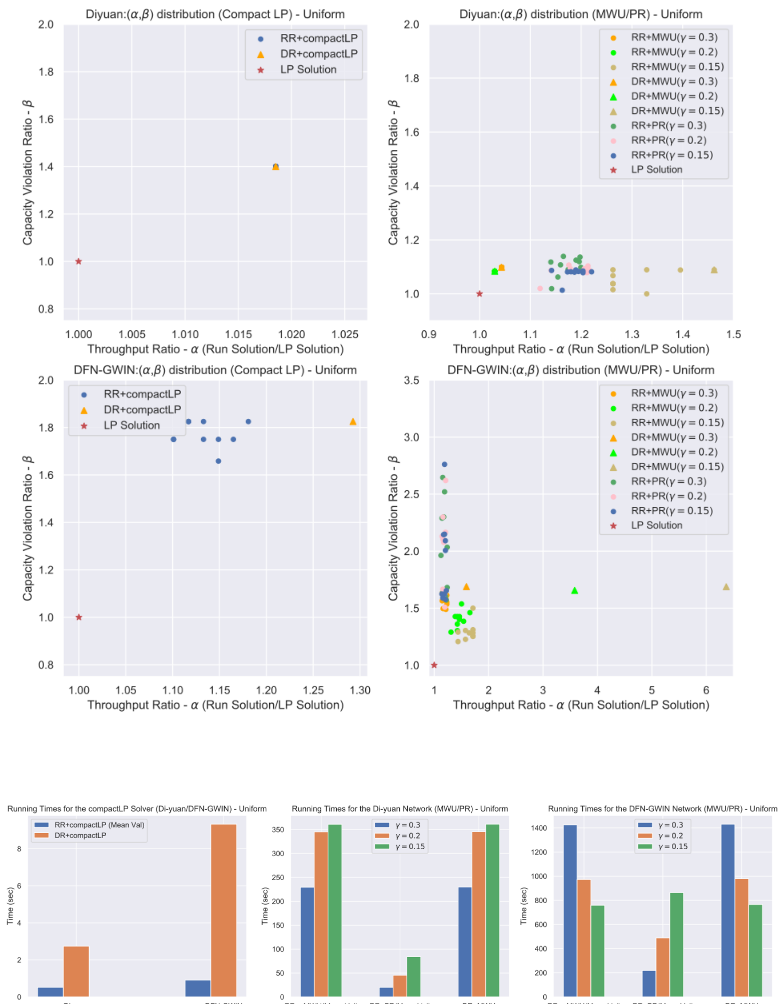

Fig. 7. Experimental Results: α/β plots and running times. Note that our theory maintains that $\beta \leq 16.52$ and $\beta \leq 16.66$ for the Di-yuan and DFN-GWIN networks, respectively. Network capacities, commodity weights and commodity demands are fixed to 40,1 and 50, respectively per [25].

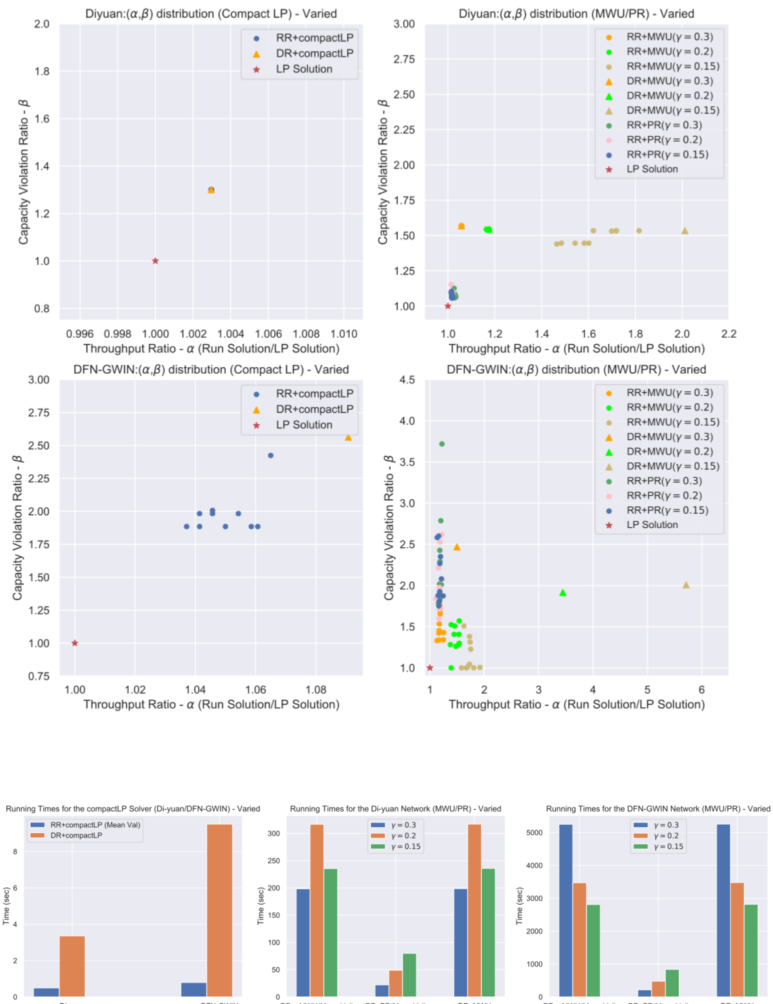

Fig. 8. Experimental Results: α/β plots and running times. Note that our theory maintains that $\beta \leq 16.52$ and $\beta \leq 16.66$ for the Di-yuan and DFN-GWIN networks, respectively. Network capacities vary from 20 to 60, commodity demands vary from 25 to 75, and commodity weights vary from 1 to 10.

F RESPONSE TO SIGMETRICS 2021 REVIEWS

This paper was submitted as #26 to the Winter deadline of Sigmetrics 2021. In the following, we would like to take the opportunity to respond to four major comments received from the reviewers, related to the novelty, the scope, the experiments, and the presentation.

Novelty. The reviewers raised the question of how novel the theoretical contribution is. Indeed, randomized rounding is a standard framework (since Raghavan and Thomson) and LP formulations have been given for our problem before. While this is true in general, we make several novel contributions on how this framework is employed in our context. While some of these adaptions may look like "details", they have important implications. For example, both in the packing formulation and in the compact edge-flow formulation, we are rounding arbitrary flows and not just flow paths (as in prior work on the ANF problem, with the exception of [21] which already considered a compact edge-flow formulation), thus allowing us to accommodate arbitrarily large ratios of (nonuniform) demands to edge capacities and also to obtain tighter approximation bounds. A central contribution of our work is related to the equivalence of LP relaxations of the packing IP and the compact edge-flow formulation IP (Section 3, page 9). Without Constraints (5) in the compact IP formulation, the relaxation of that IP would be significantly weaker, i.e., have a much larger integrality gap and hence would not allow for as good approximation bounds on throughput and edge-capacity violation, when compared to the relaxation of the packing IP. In fact, because of Constraint (5), we obtain the (at first sight perhaps even surprising) result that the two LP relaxations are in fact equivalent (equivalence as formally defined in the paper), allowing us to "get the best of both worlds": we can leverage all the proofs that are based on the packing LP relaxation also for the compact LP relaxation while being able to solve the much more compact and efficient LP (polynomial vs exponential size) in our randomized rounding algorithm. Thus, while these constraints are completely redundant for the IP formulation, they significantly strengthen the LP relaxation of the compact edge-flow formulation IP (see below) and their role in the relaxation is hence not "trivial". This equivalence is also a key novelty over [21], which provides significantly weaker approximation results. In conclusion, we would like to emphasize that all our proofs, the packing framework, the pessimistic estimators and the LP equivalence result, are new and in our opinion, non-trivial. Lastly, the MWU method presented in this paper, based on the packing LP, allows the LP to be solved in reasonable space, which becomes a problem in practice for large instances using standard LP solvers; the MWU method also allows us to address the extensions presented in Section 7, since those are easy and natural to formulate based on the packing view.

Scope. The reviewers also raised the question whether our work would not be a better match for conferences such as SODA or STOC. While indeed one could see the main contribution of the paper as being on the theoretical front, we believe that the paper suits conferences such as Sigmetrics and Performance better, for the following reasons. First, our model revolves around a fundamental question related to the throughput performance achievable in communication networks, accounting for several novel aspects which are practically relevant, such as directed links and large demands, hence also making a modeling contribution. Moreover, in addition to the theoretical results, we also contribute insights on the algorithm engineering front, proposing efficient algorithmic techniques which are motivated by the performance bottlenecks we identified with our implementations. Finally, we report on experimental results in realistic settings. We hope that especially this combination of methodologies makes our paper of interest to the Sigmetrics/Performance community.

Experiments. We noted a misunderstanding in the reviews we obtained by one of the Sigmetrics reviewer who raised the concern that our paper does not compare its results with [21]. Note that the simulation results for the algorithm in [21], adapted here to handle non-uniform demands, edge capacities and weights, have been reproduced in this paper when running the randomized rounding algorithm with the LP relaxation given by the compact edge-flow formulation LP, since that variation of the algorithm is in essence the same as the algorithm in [21] (albeit certain fine tuning done in this paper), as mentioned in the paper. In Appendix E, we present simulation results for networks that have uniform edge capacities and demands, following benchmark data in SNDlib, which is exactly the more restricted scenario considered in [21]. The main novelty in this paper when compared to the algorithm in [21] is that we are able to prove much tighter approximation bounds on the edge-capacity violation (we prove a logarithmic bound on this paper, instead of the polynomial bound in [21]) for that algorithm, confirming what the authors had already observed empirically in the simulations in [21]: the results indicated that the violation followed more of a logarithmic pattern rather than polynomial. We added further explanation on how our simulation results relate to those for [21] in the main text.

Presentation. One reviewer mentioned that there are some issues and errors in the presentation, but unfortunately, without giving details. While we give our best to avoid such errors, we would be grateful about specific examples, so we can make sure we address all of them.