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

Anya Chaturvedi [1] Chandra Chekuri [2] Andréa W. Richa [1]
Matthias Rost [3] Stefan Schmid [4] Jamison Weber [1]

ABSTRACT

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.

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 obtain a polynomial-time randomized algorithm that yields an arbitrarily good approximation on the weighted throughput while violating the edge capacity constraints by only a small multiplicative factor. 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.

1. INTRODUCTION

The study of routing and multicommodity flow problems is motivated by many real-world applications. In this paper, we are interested in throughput optimization in the context of communication networks serving multiple commodities. 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 according to their respective demands.

In contrast to most existing literature on this *all-or-nothing (splittable) multicommodity flow* problem, 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 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 [16]. 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 $1, \dots, k$ 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 formulations later).

[1]Arizona State U., USA. Research partially supported under NSF CCF-1637393 and CCF-1733680 and DoD-ARO MURI No.W911NF-19-1-0233 awards.
[2]U. of Illinois at Urbana-Champaign, USA. Partially supported by NSF grant CCF-1910149
[3]TU Berlin, Germany.
[4]Faculty of Computer Science, Univesity of Vienna, Austria; TU Berlin, Germany; and Fraunhofer SIT, Germany. Research supported by the ERC Consolidator project AdjustNet, grant agreement No. 864228.

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

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 [7] 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 [7, 9]. 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 [8]. When demands can exceed the minimum capacity, strong approximation lower bounds exist even in very restricted settings [17]. Hence, the literature has followed a bi-criteria optimization approach where edge capacities can be violated slightly. Namely, in this 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[2]. The parameter β hence provides an upper bound on the *edge capacity violation ratio* (or *congestion*) incurred by the algorithm.

1.1 Our Contributions

On the theoretical side, we obtain substantially improved bi-criteria 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. [10] 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 $O(\min\{k, \log n/\log\log n\})$, w.h.p.[2]
- 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.

As a proof of concept, we show how to *engineer our algorithms* for practical scenarios, and provide a short evaluation. In addition, our packing framework for ANF has interesting networking applications, beyond the specific model considered in this paper. In [3], 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*.

Many details, including all the proofs and more at length discussions of the algorithms, were omitted here due to space limitations; we refer to [3] for the full version of the paper.

1.2 Novelty and Related Work

Liu et al. [10] 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. Our current work significantly improves and generalizes the randomized rounding framework outlined in [10], 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,[3] and significantly improves on the bound of $O(\sqrt{k \log n})$ in [10]; 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. [7, 6, 5] 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 $\max_i d_i \leq \min_e c_e$. 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 $2 \leq d_i \leq n-1$, for all i. Hence, besides the fact that our approximation guarantees differ from those of [7] (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 [1]. 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. [8] 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 [4], Chekuri and Ene presented a a poly-logarithmic approximation with constant congestion for a special case of the ANF problem — the *Symmetric All or Nothing Flow (SymANF)* problem — in *directed* graphs with *symmetric unit demand pairs and unit edge capacities*. 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. However, their approach, like the one for undirected graphs is limited to the setting where $\max_i d_i \leq \min_e c_e$.

[2]With high probability, i.e., with probability at least $1 - 1/n^c$, for some constant $c > 0$.

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

Finally, our work leverages randomized rounding techniques presented by Rost et al. [15, 14] in the different context of virtual network embedding problems (i.e., in their context, flow endpoints are subject to optimization).

2. A PACKING FRAMEWORK FOR ANF

We develop two non-trivial mixed integer programming (MIP) formulations for the ANF problem: One is a packing formulation presented in this section, and the other is a generalization of the compact edge-flow formulation of [10], presented in Figure 1(c) and described in detail in [3, 10]. In our approach, we compute a solution to the relaxed linear program (LP) in polynomial time and then convert this solution into an integer solution via appropriate randomized rounding. The "packing view" of the ANF problem presented allows for a more flexible approach, that can also solve several extensions of interest in the networking domain, such as the ones listed in Section 1.1. 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."

We assume that all commodities can be routed to satisfy their demands in isolation in G, since if pair i cannot be routed at a value of d_i in isolation, then we may as well discard it (since there are no flows that can satisfy the demand for i in G). 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.

In Figure 1(a), we describe our packing MIP formulation for 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. .

In [3], we prove that the linear programming (LP) relaxation of the ANF packing formulation presented in Figure 1(b) is *equivalent* to the relaxation of the compact edge-flow MIP in Figure 1(c). By equivalence of LPs in this context we specifically mean the following: For any feasible solution to one LP, there exists a feasible solution to the other LP that routes the exact same amount of flow for each commodity in each edge, and vice-versa. This of course also implies that the values of the corresponding feasible solutions for the two LPs must be the same, and that the approximation bicriteria results that we will prove based on the relaxation of the packing formulation also apply if instead we used the relaxation of the compact edge-flow formulation, which we do in the simulations.

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

(a)

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

(b)

$$
\begin{aligned}
&\max \sum_{i=1}^{k} w_i f_i && \\
&\sum_{(s_i,v) \in E} f_{i,(s_i,v)} = f_i && \forall i \in [k] \\
&\sum_{(u,v) \in E} f_{i,(u,v)} = \sum_{(v,u) \in E} f_{i,(v,u)} && \forall i \in [k], \forall v \in V - \{s_i, t_i\} \\
&\sum_{i=1}^{k} f_{i,(u,v)} \cdot d_i \le c_{(u,v)} && \forall (u,v) \in E \\
&f_{i,(u,v)} \cdot d_i \le f_i \cdot c_{(u,v)} && \forall i \in [k], \forall (u,v) \in E \\
&f_{i,(u,v)} \ge 0 && \forall i \in [k], \forall (u,v) \in E \\
&f_i \in \{0,1\} && \forall i \in [k]
\end{aligned}
$$

(c)

Figure 1: (a) Mixed integer programming (MIP) packing formulation for ANF, based on "flow" variables; (b) its LP relaxation; (c) compact edge-flow formulation.

Solving the Packing LP. 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 there is an *efficient separation oracle for the dual LP of the LP-relaxation in Figure 1(b)* and the other is to solve the *equivalent compact (polynomial-size) formulation* to the ANF LP. In this section, we will discuss the former approach, which gives us a more flexible formulation that leads to interesting extensions and also to simpler proofs. The compact edge-flow formulation has size polynomial in n and k and hence its relaxation can be directly solved in polynomial time.

One can use a *polynomial-time separation oracle for solving the dual* of the relaxation of the packing ANF MIP presented in Figure 1(b), as shown in [3]. Standard techniques allow one to obtain an optimal solution to the primal LP from an optimum solution to the dual LP. However, since the Ellipsoid method is impractical, one could use the generalization of the compact edge-flow formulation in [10] or the algorithm presented in Section 3 to more efficiently solve the ANF packing LP in practice, which we will do in our implementations.

Rounding the Packing LP. In this section, we show how to round a (fractional) solution to the packing ANF MIP formulation. 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 [13]. 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 $\mathrm{OPT}_{\mathrm{LP}}$ and the value of an optimum integer solution as $\mathrm{OPT}_{\mathrm{IP}}$. We observe that $\mathrm{OPT}_{\mathrm{LP}} \geq \mathrm{OPT}_{\mathrm{IP}}$ and $\mathrm{OPT}_{\mathrm{LP}} \geq W_{\mathrm{LP}}$. Note that when solving the formulation in Figure 1(b) or the compact formulation in Figure 1(c), the LP solution obtained will be optimal and hence $W_{\mathrm{LP}} = \mathrm{OPT}_{\mathrm{LP}}$; however, the solution obtained via the multiplicative-weight update algorithm of Section 3 may only approximate $\mathrm{OPT}_{\mathrm{LP}}$ and hence one could indeed have $\mathrm{OPT}_{\mathrm{LP}} > W_{\mathrm{LP}}$. We will also assume that $\mathrm{OPT}_{\mathrm{LP}} \geq w_{\max}$, 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 $w_{\max} \leq \mathrm{OPT}_{\mathrm{IP}} \leq \mathrm{OPT}_{\mathrm{LP}}$. Using standard Chernoff bounds [11], we get

LEMMA 2.1. *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, $\Pr[Z < (1-\delta)W_{LP}] < e^{-\delta^2/2}$.*

Furthermore, we can show

LEMMA 2.2. *For $m \geq 9$ and $b > 1$ the probability that the total flow on an edge e is more than $(\frac{3b \ln m}{\ln \ln m})c(e)$ 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 $(\frac{3b \ln m}{\ln \ln m})c(e)$ is at most $e^{-(1.5b-1) \ln m - 3b \ln b \ln m / \ln \ln m - 1}$.*

We can now put together the preceding lemmas to derive our bicriteria approximation, stated in our main theorem. We will henceforth assume that $m \geq 9$, with all proofs appearing in [3].

THEOREM 2.3. *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 w.h.p. Furthermore, we guarantee a $(1-1/m, O(\ln m/ \ln \ln m))$-approximation w.h.p., by setting $\epsilon = 1/m$.*

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 2.4. *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 w.h.p.*

3. 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 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 [2], we refer to [3] for a full description of the specific MWU algorithm that we implement for the ANF packing formulation and just state the performance guarantees of the algorithm here.

It is known that the MWU algorithm, as suggested above, terminates in $O(m \log m/\gamma^2)$ iterations [2]. In our implementation of the MWU algorithm, 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 runtime by $O(\mathrm{MCF}(n,m)km \log m/\gamma^2)$ where $\mathrm{MCF}(n,m)$ 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.

4. RANDOMIZED ROUNDING

Algorithm 1 describes the actual randomized rounding algorithm that we 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 (since any feasible solutions to the compact LP can trivially be viewed as a feasible solutions to the packing LP). 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.

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 for each commodity, 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

Algorithm 1: 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.

all $e \in E$). We repeat Steps 2-3 of Algorithm 1 $\Theta((\ln m)/\epsilon^2)$ 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 between the packing and the compact LP, which implied among other things that any feasible solution to the packing LP can be translated into a feasible solution to the compact LP of equal objective function value and that Algorithm 1 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 2.3:

COROLLARY 4.1. *Algorithm 1, when run on an optimum solution to the ANF LP, achieves a* $((1-\epsilon), 3b \ln m / \ln \ln m)$-approximation *for the ANF problem on* arbitrary networks *w.h.p, 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 1 using the solution output by the MWU algorithm, which only guarantees a $(1 - \gamma)$ approximation on the throughput for $\gamma \in (0, 1)$. Note that the throughput approximation guarantee for Algorithm 1 using MWU will be $(1 - \epsilon)(1 - \gamma)$.

Another advantage of Algorithm 1 is that it leads to a surprisingly simple *derandomized algorithm*, which we describe in full in [3] and which we also implemented.

5. 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). Notably, due to the bi-criteria nature of our approximations, that allows violations of edge capacities, 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 a MWU-based Permutation Routing (PR) heuristic. 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 certain extensions that depend on the packing formulation, as mentioned in Section 1.1. 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 [10] — adapted here to handle non-uniform demands, edge capacities and weights — have been reproduced when running the randomized rounding algorithm with the compact edge-flow LP, since this algorithm is in essence the same as the algorithm in [10] (albeit some fine tuning optimizations). Our theoretical approximation results in this paper actually also validate the experimental results in [10], that already suggested that the logarithmic edge capacity violations.

Methodology. Following [10], we study real-world networks together with corresponding real-world source-sink pairs obtained from the survivable network design library (SNDlib) [12]. 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 (non-trivial) 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. We selected networks from the SNDlib that cover several general scenarios. 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).

We have implemented both the randomized and derandomized rounding algorithms. We solve the compact formulation optimally via CPLEX V12.10.0, and the packing LP approximately via the MWU algorithm or the faster permutation routing heuristic. We choose $\epsilon = \frac{1}{9}$ and $b = 1.85$, implying target throughput approximation factor of $\alpha \geq 1 - \epsilon = \frac{8}{9}$ and 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. More specifically, for the Germany50 network, we target an edge capacity violation $\beta \leq 17.47$.

We define an experiment as the execution of a higher level algorithm (either randomized rounding or derandomized algorithm) 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 MCF solution is not at least 50 percent larger than the respective pre-update MCF solution, then we do not recompute this in the subsequent iteration, and (ii) the maximum number of iterations is capped at 10k.

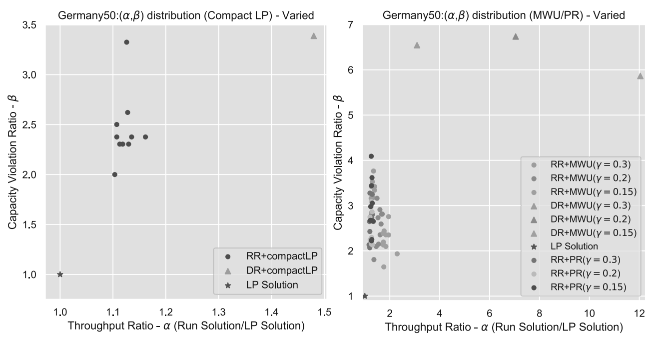

Figure 2: Experimental results on the Germany50 network; RR refers to the randomized and DR to the derandomized rounding algorithms.

Experimental Results. We report results of our experiments in terms of the achieved throughput factor α, edge capacity violation factor β, and the wall-clock running times. The results of the experiments specifically on the Germany50 network from [12] are summarized visually in Figure 2. The qualitative plots 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. The red star on the graph serves a reference point, since it indicates the optimal LP solution. Although in this paper we show only the results from the Germany50 network, [3] presents results for three other networks from [12] with comparable findings.

We see for the Germany50 network 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 MWU are similarly concentrated around their means. Interestingly, in combination with the MWU algorithm, the deterministic rounding shows 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 naïve implementation of the pessimistic estimators, that does not cache intermediate results.

Concluding, 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 higher throughputs at the expense of higher edge capacity violations. Furthermore, the proposed MWU algorithm will render tackling problem extensions tractable and our proposed permutation routing heuristic can substantially reduce runtimes in practice. See [3] for more comprehensive simulation results.

6. REFERENCES

[1] M. Andrews, J. Chuzhoy, V. Guruswami, S. Khanna, K. Talwar, and L. Zhang. Inapproximability of edge-disjoint paths and low congestion routing on undirected graphs. *Combinatorica*, 30(5):485–520, 2010.

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

[3] A. Chaturvedi, C. Chekuri, A. W. Richa, M. Rost, S. Schmid, and J. Weber. Improved bi-criteria approximation for the all-or-nothing multicommodity flow problem in arbitrary networks. *CoRR*, abs/2005.04533, 2021.

[4] C. Chekuri and A. Ene. The all-or-nothing flow problem in directed graphs with symmetric demand pairs. *Math. Program.*, 154(1-2):249–272, 2015.

[5] C. Chekuri, S. Khanna, and F. B. Shepherd. Multicommodity flow, well-linked terminals, and routing problems. In *ACM STOC*, pages 183–192, 2005.

[6] C. Chekuri, S. Khanna, and F. B. Shepherd. The all-or-nothing multicommodity flow problem. *SIAM Journal on Computing*, 42(4):1467–1493, 2013.

[7] C. Chekuri, S. Khanna, and F. Bruce Shepherd. The all-or-nothing multicommodity flow problem. In *ACM STOC*, page 156–165, 2004.

[8] J. Chuzhoy, V. Guruswami, S. Khanna, and K. Talwar. Hardness of routing with congestion in directed graphs. In *ACM STOC*, pages 165–178, 2007.

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

[10] M. Liu, A. Richa, M. Rost, and S. Schmid. A constant approximation for maximum throughput multicommodity routing and its application to delay-tolerant network scheduling. In *IEEE INFOCOM*, pages 46–54, 2019.

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

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

[13] P. Raghavan and C. D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[14] M. Rost, E. Döhne, and S. Schmid. Parametrized complexity of virtual network embeddings: Dynamic & linear programming approximations. *ACM SIGCOMM*, 49:3–10, 2019.

[15] M. Rost and S. Schmid. Virtual network embedding approximations: Leveraging randomized rounding. In *IFIP Networking*, pages 1–9, 2018.

[16] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren. Inside the social network's (datacenter) network. In *ACM Conference on Special Interest Group on Data Communication*, pages 123–137, 2015.

[17] F. Bruce Shepherd and A. Vetta. The inapproximability of maximum single-sink unsplittable, priority and confluent flow problems. *CoRR*, abs/1504.00627, 2015.