Public Review for

On Max-min Fair Allocation for Multi-source Transmission

G. Li, Y. Qian, Y. Richard Yang

Max-min fairness is one of the classical objectives for traffic engineering or congestion control schemes. It is well understood and widely used by the community. When we received this submission, we were a bit surprised to see another paper on this rather old topic. In the classical max-min fairness formulation, each source produces its own stream of data and max-min fairness ensures a fair distribution of the ressources among competing flows. The new model proposed in this paper adapts the formulation to the more general case of data streams which can originate from multiple sources. This corresponds to the utilization of replicas such as CDNs where multiple hosts can source the same data stream. The authors formulate this problem as an optimization problem and propose a MultiSource Water-Filling algorithm (MS-WF) to compute the bandwidth allocations and the flow assignments. The proposed algorithm is described and evaluated by simulations. Unfortunately, the authors could not release this simulator and the associated results as software artifacts.

The reviewers discussed the merits of the paper. Some were convinced by the nice formulation of the problem and the initial simulation results. A drawback of the approach is that MS-WF assumes a centralized controller that has a precise information about the traffic matrix, which might be unrealistic in some deployments. The simulation results should be analyzed with this caveat in mind. Other reviewers were more concerned about the applicability and the deployability of the proposed solution. These are clearly left for further work. They also questioned whether max-min was the right goal. The paper was discussed during the rebuttal phase and the authors have revised the paper accordingly and added the proofs in a technical report.

Public review written by
Olivier Bonaventure
UCLouvain

On Max-min Fair Allocation for Multi-source Transmission

Geng Li[+*], Yichen Qian[*], Y.Richard Yang[+]
[+] Yale University, [*] Tongji University
geng.li@yale.edu,92yichenqian@tongji.edu.cn,yry@cs.yale.edu

ABSTRACT

Max-min fair is widely used in network traffic engineering to allocate available resources among different traffic transfers. Recently, as data replication technique developed, increasing systems enforce multi-source transmission to maximize network utilization. However, existing TE approaches fail to deal with multi-source transfers because the optimization becomes a joint problem of bandwidth allocation as well as flow assignment among different sources. In this paper, we present a novel allocation approach for multi-source transfers to achieve global max-min fairness. The joint bandwidth allocation and flow assignment optimization problem poses a major challenge due to nonlinearity and multiple objectives. We cope with this by deriving a novel transformation with simple equivalent canonical linear programming to achieve global optimality efficiently. We conduct data-driven simulations, showing that our approach is more max-min fair than other single-source and multi-source allocation approaches, meanwhile it outperforms others with substantial gains in terms of network throughput and transfer completion time.

CCS CONCEPTS

• **Networks → Packet scheduling**;

KEYWORDS

Max-min Fairness, Traffic Engineering, Multi-source Transmission

1 INTRODUCTION

Max-min fair is a simple, classical and well-recognized sharing principle to define fairness in the field of data networks [14]. In particular, it deals with the flow control problem in network traffic engineering (TE), where available resources (such as link bandwidth) are allocated among different traffic transfers [13]. Aiming at allocating rates to available links as evenly as possible, the max-min fair allocation is such that the result for the transfer with smallest sharing has been maximized over all feasible resource allocation solutions. As the development of software-defined networking (SDN), max-min fair has become the most widely used principle in modern datacenter networks or WANs, such as B4 [9], BwE [11], SWAN [8], OWAN [10]. Leveraging a centralized controller, max-min fairness results in more stable service quality than other principles, *e.g.*, proportional fairness.

Data replication, a technique that amends both data availability and access efficiency, is emerging increasingly in recent datacenter networks and distributed filesystems [7, 15, 17]. To maximize network utilization and improve transmission performance, it is increasingly allowed to convey data in parallel from multiple sources for a transfer with multiple data replicas, *a.k.a* a multi-source transfer [2, 16]. However, a multi-source transfer will result in multiple flows, so multiple potential bottlenecks might be considered simultaneously to achieve transfer-level fairness. Existing TE approaches are no longer applicable because the allocation becomes a joint problem to optimally allocate each flow's bandwidth as well as to assign flows among the sources [8, 9, 11].

In this paper, we present a novel max-min fair allocation approach that jointly optimizes bandwidth allocation and flow assignment. The major challenge stems from the direct formulation which is nonlinear and multi-objective. We cope with this by deriving a novel transformation with simple equivalent canonical linear programming (LP) to achieve global optimality. We perform extensive simulations, which show that our approach leads to better performance on network throughput with a gain of up to 52% and reduces transfer completion time by up to 44%, compared to other single-source and multi-source allocation approaches.

2 NETWORK MODEL AND PROBLEM FORMULATION

A network is comprised of a set of nodes and a set of links $\mathcal{L}$. Let C_j denote the capacity of link L_j ($j \in [1, M]$). Consider N data transfers, coming from one or multiple sources, so each transfer i ($i \in [1, N]$) is assigned with a set of flow paths $\{P_{i1}, ..., P_{ik}\}$, and each such path is identified with the set of links that it uses, *i.e.,* $P_{ik} \subseteq \mathcal{L}$. Now we define the data rate of transfer i as r_i, which is the sum bandwidth of the constituent flows from all sources. We use a variable set $\mathcal{X}_i = \{x_{i1}, ..., x_{ik}\}$ to express the flow assignment proportions from different sources for transfer i, so $\sum_{k=1}^{K_i} x_{ik} = 1$, where K_i is the total source number of transfer i.

Objective and constraints. Our goal is to solve a joint bandwidth allocation and flow assignment problem, *i.e.,* finding the transmission rate r_i and the assignment proportions $\mathcal{X}_i$ of each transfer. When computing allocated bandwidth, the objective is to maximize network utilization while in a max-min fair manner. A vector of transfer rate allocations $\{r_i\}$ is said to be max-min fair if and only if, for any other feasible allocation $\{r_i'\}$, the following has to be true: $\forall r_p' > r_p$ for the data transfer p, there exists another transfer q such that $p, q \in [1, N]$, $r_q' < r_q$, $r_q \leq r_p$. The constraints of this problem are given as below. Constraint (1) is called the capacity constraint, which assures that for any link L_j, its load does not exceed its capacity C_j. Constraints (2) and (3) promise the sum fractions of a transfer from all available sources equal to 1.

$$s.t. \quad \sum_{L_j \subseteq P_{ik}} r_i \cdot x_{ik} \leq C_j \qquad \forall j, \tag{1}$$

$$\sum_{k=1}^{K_i} x_{ik} = 1 \qquad \forall i, \tag{2}$$

$$0 \leq x_{ik} \leq 1 \qquad \forall i, k. \tag{3}$$

Figure 1 shows an example where the six links $\{L_1, ..., L_6\}$ have capacities $\{8, 5, 4, 5, 7, 6\}$ in *Gbps* respectively. Consider three data transfers, among which transfer 1 and 2 have a single data source

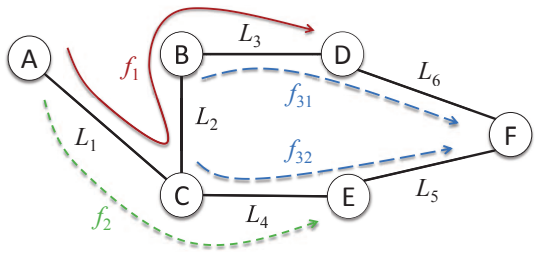

Figure 1: An example with 3 transfer requests. Transfer 3 comes from two feasible sources B and C, corresponding to two parallel flows f_{31} and f_{32}.

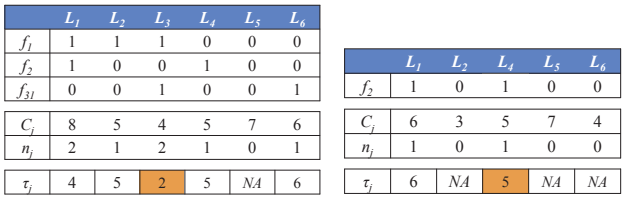

Figure 2: An example of the FL matrix for single-source transmission, where transfer 3 only uses one source. C_j is the bandwidth capacity, $n_j = \sum_i fl_{ij}$ is the total number of flows that use link L_j, and $\tau_j = C_j/n_j$ is the saturated average bandwidth share. (a) Illustration of the first iteration, where L_3 is found as the bottleneck link with minimum τ_j. (b) Illustration of the second iteration with the updated FL matrix, where L_4 is then found as the bottleneck link.

while transfer 3 has two. As a result, there are four potential flows in total, including $f_1 : A \to C \to B \to D$, $f_2 : A \to C \to E$, $f_{31} : B \to D \to F$ and $f_{32} : C \to E \to F$. Given the topology and values of link capacities, we aim at finding the max-min fair solution of the rate vector $\{r_1, r_2, r_3\}$ and the flow assignment $\{x_{31}, x_{32}\}$ for transfer 3.

3 APPROACH SPECIFICATION

One of the biggest challenges for multi-source transmission stems from changing the optimized object from an individual 5-tuple flow into a group of flows that belong to the same transfer. Hence, multiple potential bottlenecks might be considered simultaneously. To this end, we design a MultiSource Water-Filling (MS-WF) algorithm which can jointly compute bandwidth allocation and flow assignment while providing global max-min fairness. The key to MS-WF algorithm is a novel transformation with simple equivalent canonical LP. In this section, we first briefly describe the traditional water-filling algorithm and its limitation. Then we present the MS-WF with one multi-source transfer for clear illustration. A general solution that optimizes arbitrary flow transfers is provided afterward. At last we discuss some practical issues of the multi-source framework.

3.1 Flow-link Mapping Matrix and Traditional Water-Filling Algorithm

The proposed MS-WF algorithm is based on a flow-link mapping matrix (FL matrix) with solvable variables. Here we define the FL matrix, and illustrate the traditional water-filling algorithm with this matrix.

Definition 1 (Flow-link Mapping Matrix)**.** The flow-link mapping matrix *(FL matrix)* $\{fl_{ij}\}$ expresses the flow paths and the traffic assignment of each transfer in matrix form. The element $fl_{ij} \in [0, 1]$ is defined as the proportion of flow i in its belonging transfer that uses link L_j.

Water-Filling (WF) Algorithm. The traditional WF algorithm can compute the bandwidth allocations for single-source transmission. Using the FL matrix as an input, we first denote the saturated average bandwidth allocation as $\tau_j = C_j/n_j$, where n_j is the total number of flows that use link L_j. The WF algorithm iteratively finds the minimum τ^* and the corresponding bottleneck link L_{j^*}. Set the bandwidth of the flows that use link L_{j^*} to τ^*. The FL matrix is then updated by subtracting these flows and the bottleneck link to calculate a new set of $\{\tau_j\}$. Such process repeats until all transfers attain their allocated rates, *i.e.,* all flows have bottleneck links.

Example. Consider a single-source version of Figure 1, where we assume transfer 3 only uses source B, so there are 3 flows over the network. Figure 2 illustrates the allocation algorithm for this single-source example. In the first iteration, L_3 is first saturated because τ_3 is of the minimum value $\tau^* = 2$. We allocate the bandwidth of f_1 and f_{31} that traverse L_3 to 2, and then remove the rows of f_1, f_{31} and column L_3. The FL matrix is updated accordingly for the next iteration, where link L_4 is then found as the bottleneck and bandwidth of f_2 is set to 5. By this point, the ultimate solution to the rate allocation for the 3 transfers is (2, 5, 2).

The traditional WF algorithm is successful in obtaining the max-min fair allocation for single-source transmission. Though it is incapable of dealing with multi-source transfers. This is mainly due to the fact that, instead of transfers, the WF algorithm is based on flows. For the example provided in Figure 1, by using the traditional WF algorithm, transfer 3 will have double weights, which is unfair to the others. A naïve solution is to normalize the weight of each transfer, and then to assign an equal share to the flows from different sources. Regarding the example, the elements for f_{31} and f_{32} become 1/2 and 1/2 in the FL matrix, such that each transfer has the same sum weight of 1. The allocation result turns into (8/3, 10/3, 3), which is slightly better than the (2, 5, 2) - using only source B and (2.5, 4, 2.5) - using only source C, namely the single-source cases. However, as we will soon learn, simply sharing the flow weights equally is still not the optimal solution. *The transfer-level max-min fair allocation is conditioned by the optimal flow assignment.*

3.2 MS-WF Algorithm with One Multi-source Transfer

Given the FL matrix and its application, now let's consider a more complex case by adding one multi-source transfer into the network. We assume there are K_m available sources for the particular transfer m, though the flow assignment X_m is unknown and needs to be solved. The MS-WF algorithm continues to use the FL matrix, but

Algorithm 1 MS-WF Algorithm with one multi-source transfer

Input:
Flow-link mapping matrix: $FL = \{fl_{ij}\}$;
Capacity of each link: $\{C_j\}$, $1 \leqslant j \leqslant M$;
Output:
Transmission rate of each transfer: $\{r_i\}$, $1 \leqslant i \leqslant N$;
Flow assignment of transfer m: $\mathcal{X}_m = \{x_{m1}, \ldots, x_{mk}\}$;

- **Step 1:** ▷ Initiation
  $r_i \leftarrow 0, \quad \forall i = 1, \ldots, N$;
  $\mathcal{L} \leftarrow \{L_1, L_2, \ldots, L_M\}$;
- **Step 2:** ▷ Calculate the saturated average bandwidth
  $n_j(\mathcal{X}_m) \leftarrow \sum_i fl_{ij}, \quad \forall j \in \mathcal{L}$;
  $\tau_j(\mathcal{X}_m) \leftarrow C_j / n_j(\mathcal{X}_m), \quad \forall j \in \mathcal{L}$;
- **Step 3:** ▷ Find the bottleneck fair share τ^*
  if $min\{\tau_j(\mathcal{X}_m)\}$ is a constant **then**
    $\mathcal{X}^* \leftarrow \varnothing$;
  else
    $\mathcal{X}^* \leftarrow \mathcal{X}_m | max\ min\{\tau_j(\mathcal{X}_m)\}$;
  end if
  $\tau^* \leftarrow min\{\tau_j(\mathcal{X}_m)\}$;
- **Step 4:** ▷ Set data rate and update the FL matrix
  $\mathcal{L}_{j^*} \leftarrow \{L_j | \tau_j(\mathcal{X}_m) = \tau^*\}$;
  $\mathcal{L} \leftarrow \complement_{\mathcal{L}_{j^*}} \mathcal{L}$;
  for $i | P_i \cap \mathcal{L}_{j^*} \neq \varnothing$ **do**
    $r_i \leftarrow r_i + \tau^*$;
    Remove f_i from FL;
  end for
  Update FL;
- **Step 5:** ▷ Iteration
  if No transfers left **then**
    return $\{r_i\}$ and $\mathcal{X}_m$;
  else
    goto **Step 2**.
  end if

replaces the elements from 0 and 1 as in the traditional WF into the unknown variables in $\mathcal{X}_m$ for transfer m. The MS-WF algorithm with one multi-source transfer (**Algorithm 1**) is described as follows:

- **Step 1:** Start from zero allocation and build the FL matrix with variables $\mathcal{X}_m$.
- **Step 2:** Compute the saturated average bandwidth $\tau_j(\mathcal{X}_m)$ on each link L_j.
- **Step 3:** Find the bottleneck fair share $\tau^* = min\{\tau_j(\mathcal{X}_m)\}$ by solving $\mathcal{X}^* = \mathcal{X}_m | max\ min\{\tau_j(\mathcal{X}_m)\}$.
- **Step 4:** Set the data rate to τ^* for the flows that traverse the bottleneck links, and update the FL matrix by removing those flows and links.
- **Step 5:** Stop if there are no transfers left; otherwise return to Step 2.

In Step 2, similar to the traditional WF algorithm, MS-WF computes $n_j(\mathcal{X}_m)$ by summarizing the elements of column j in the FL matrix. Here $n_j(\mathcal{X}_m)$ is the number of transfers that traverses link L_j. Since transfer m comes from multiple parallel flows, there might be only a part of it using link L_j. Therefore, n_j becomes a function of $\mathcal{X}_m$ instead of an integer value as in the traditional WF. After that, MS-WF calculates the average bandwidth $\tau_j(\mathcal{X}_m) = C_j / n_j(\mathcal{X}_m)$, which is also a function of $\mathcal{X}_m$.

	L_1	L_2	L_3	L_4	L_5	L_6
f_1	1	1	1	0	0	0
f_2	1	0	0	1	0	0
f_{31}	0	0	x	0	0	x
f_{32}	0	0	0	$1-x$	$1-x$	0
C_j	8	5	4	5	7	6
n_j	2	1	$1+x$	$2-x$	$1-x$	x
τ_j	4	5	$\frac{4}{1+x}$	$\frac{5}{2-x}$	$\frac{7}{1-x}$	$\frac{6}{x}$

Figure 3: An example of MS-WF, where transfer 3 comes as two flows. x is the flow assignment variable to be calculated. τ_j in orange cell is found as the minimum bottleneck share.

In Step 3, potential bottleneck links are found given $\{\tau_j(\mathcal{X}_m)\}$ on the current link set. Specifically, the flow assignment is determined by $\mathcal{X}^* = \mathcal{X}_m | max\ min\{\tau_j(\mathcal{X}_m)\}$. We use τ^* to denote the minimum bandwidth share, and the set of $\{L_{j^*}\}$ is found as the bottleneck links. As all the variables $\mathcal{X}_m$ are within a certain range $[0, 1]$, $min\{\tau_j(\mathcal{X}_m)\}$ is sometimes a constant value. In that case, no flow assignment variable is calculated, *i.e.*, $\mathcal{X}^* = \varnothing$.

Back to the example in Figure 1, transfer 3 has two available sources to access, leading to two parallel flows f_{31} and f_{32}. Without loss of generality, we use only one variable x to denote the proportion of f_{31}, so the proportion of f_{32} will be $1 - x$. Figure 3 illustrates the FL matrix, followed by the saturated average bandwidths $\{\tau_j(\mathcal{X}_m)\}$.

In MS-WF, Step 3 finding $\mathcal{X}^*$ that maximizes $min\{\tau_j(\mathcal{X}_m)\}$, is the major challenge. Specifically, it is to find $x^* = x | max\ min(4, 5, 4/(1+x), 5/(2-x), 7/(1-x), 6/x)$ in Figure 3. First, as formulated in **Problem 1**, it is a nonlinear programming problem, which can not be directly solved. Second, even though there is a linear expression, the solution needs long sequences of LPs for the max-min objective (multi-objective), which are computationally intense in practice.

Problem 1 (The nonlinear optimization problem in Step 3)**.**

$$max \quad min\{\tau_j(\mathcal{X}_m)\}, \tag{4}$$

$$s.t. \quad \sum_{k=1}^{K_m} x_{ik} = 1 \qquad x_{ik} \in \mathcal{X}_m, \tag{5}$$

$$0 \leq x_{ik} \leq 1 \qquad \forall i, k. \tag{6}$$

To this end, we propose a novel transformation which converts the nonlinear optimization problem into a canonical form of LP problem based on **Theorem 1**. The equivalent canonical LP problem is defined in **Problem 2**, which can be efficiently solved under limited computational complexity.

Problem 2 (The equivalent canonical LP problem in Step 3)**.**

$$min \quad t \tag{7}$$

$$s.t. \quad t \geq \tau'_j(\mathcal{X}_m) \qquad \forall j, \tag{8}$$

$$\sum_{k=1}^{K_m} x_{ik} = 1 \qquad x_{ik} \in \mathcal{X}_m, \tag{9}$$

$$0 \leq x_{ik} \leq 1 \qquad \forall i,k. \tag{10}$$

Theorem 1. *Problem 1 is equivalent to Problem 2 as a canonical LP problem, where* $\tau'_j(\mathcal{X}_m) = 1/\tau_j(\mathcal{X}_m)$.

Proof. Given an arbitrary instance of the FL matrix, $\tau_j(\mathcal{X}_m)$ satisfies two conditions: i) $\tau_j(\mathcal{X}_m) \geq 0$, and ii) the inverse of $\tau_j(\mathcal{X}_m)$ is a linear function of $\mathcal{X}_m$. So we let $\tau'_j(\mathcal{X}_m) = 1/\tau_j(\mathcal{X}_m)$, and the objective of $max\, min\{\tau_j(\mathcal{X}_m)\}$ is then equivalent to $min\, max\{\tau'_j(\mathcal{X}_m)\}$, which becomes linear accordingly. Next, we introduce a temporary variable $t = max\{\tau'_j(\mathcal{X}_m)\}$, and use a sequence of inequality constraints $t \geq \tau'_j(\mathcal{X}_m)$ for all j to express t. Since $\tau'_j(\mathcal{X}_m)$ is a linear function of $\mathcal{X}_m$, the constraint (8) as a set of inequalities is also linear. In the end, the optimization problem in Step 3 (**Problem 1**) turns into an equivalent canonical LP problem (**Problem 2**), where the decision variables to be solved are the flow assignment set $\mathcal{X}_m$ and t. □

Consequently, the optimization problem in Figure 3 can be transformed into a simple equivalent LP problem shown as follows.

$$min \quad t \tag{11}$$

$$s.t. \quad t \geq \frac{1}{4}, \tag{12}$$

$$t \geq \frac{1}{5}, \tag{13}$$

$$4t - x \geq 1, \tag{14}$$

$$5t + x \geq 2, \tag{15}$$

$$7t + x \geq 1, \tag{16}$$

$$6t - x \geq 0, \tag{17}$$

$$0 \leq x \leq 1. \tag{18}$$

The results of the above LP come out as $x = 1/3$ and $t = 1/3$. So $\tau^* = 1/t = 3$ is the minimum fair share, and L_3 and L_4 are the bottleneck links that are saturated in this iteration. We set $r_1 = r_2 = 3$ as the bandwidth of f_1 and f_2, and $r_3 = 3$ as the sum bandwidth of f_{31} and f_{32}. Meanwhile, the data volume assignment of transfer 3 concludes with 1/3 from source B and 2/3 from source C. The final rate allocation to the 3 transfers are (3, 3, 3), which is more max-min fair than the allocation (2, 5, 2) where transfer 3 uses only source B (as in Figure 2), as well as the allocation (2.5, 4, 2.5) where transfer 3 uses only source C. In addition, if we don't consider the impact of date volume and assume each transfer has the equal volume of $3Gbits$, then MS-WF outperforms the single source approaches in terms of the average completion time (MS-WF: (3/3+3/3+3/3)/3=1, source B: (3/2+3/5+3/2)/3=1.2) and source C: (3/2.5+3/4+3/2.5)/3=1.05), as well as total completion time (MS-WF: 3/3=1, source B: 3/2=1.5 and source C: 3/2.5=1.2).

3.3 General MS-WF

Having solved the preliminary instance with one multi-source transfer, now we consider the general MS-WF with arbitrary transfer combinations. Here the variables become the flow assignments $\{\mathcal{X}_i\}$ ($i \in [1, N]$) for all transfers. Except for the transfers with a single source, whose $\mathcal{X}_i = \{1\}$ for all the time, the rest of $\{\mathcal{X}_i\}$ are to be computed by MS-WF. The main challenge is that the flow assignments of different transfers correlate to each other and can not be calculated independently. One transfer's assignment plan may affect another's optimal decision. *Therefore, the max-min fair allocation requires joint calculation for all flow assignments.*

The general MS-WF mainly follows the procedures in **Algorithm 1**. Exceptionally, we put all sets of the variables $\{\mathcal{X}_i\}$ into the FL matrix, such that τ_j turns into a function of $\{\mathcal{X}_1, ..., \mathcal{X}_N\}$. By the same token, we transform the nonlinear optimization problem in Step 3 into an equivalent canonical LP problem with the help of one additional decision variable t. In each iteration, parts of the flow assignment sets are solved by LP based on **Theorem 2**. Then we plug the values into the FL matrix and remove them from $\{\mathcal{X}\}$. Continue iterating until all flow assignment variables $\mathcal{X}_i$ are determined. Finally, we sum up the constituent flow rates as the multi-source transfer rate, *i.e.,* $r_i = \sum_{k=1}^{K_i} r_{ik}$, where K_i is the total source number of transfer i. The detailed proof of **Theorem 2** can be found in our technical report [1].

Theorem 2. *Multiple sets of variables* $\mathcal{X}_i$ *can be jointly calculated by MS-WF.*

3.4 Discussion

Extension. The MS-WF algorithm is scalable and extensible for more complex use cases. For instance, differentiated qualities of service lead to transfers with variations in priority or other requirements, such that MS-WF is capable of supporting weighted fair allocation by taking priority factors into account. The max-min fair principle can also be applied to different optimization objectives (*e.g.,* transfer completion time), and the adaption of MS-WF with consideration of data size can likewise yield the optimal results for multi-source transfers.

Applicability. The driving application scenario for the MS-WF algorithm is the Large Hadron Collider (LHC) network, which requires deadline scheduling of large-scale datasets (*e.g.,* petabytes) to be transferred around over 180 member sites all over the world [5]. Fairness among large-scale science dataflows is one of the most important metrics for the LHC network, and the current scheduling system performs poorly in terms of fairness due to missing a fairness-aware scheduling framework for the multi-source transfers. Our multi-source transmission framework is part of the preproduction deployment of Unicorn [18], an SDN geo-distributed data analytic system in the CMS (one of the largest scientific experiments in the LHC network).

The experimental SDN system relies on a logically centralized controller to orchestrate bulk transfers. Since the system is for large-scale datasets, the average flow duration may vary from hours to several days. The computation complexity is significantly reduced in MS-WF by transforming a nonlinear multi-objective problem into a single LP. Experimental results show that the computation time for 1000 concurrent flows in MS-WF is at most 40 seconds, and it can be further reduced by removing the redundant constraints in implementation [12]. Therefore, our approach is scalable to practically handle large-scale datasets and networks.

4 PERFORMANCE EVALUATION

To evaluate the performance of MS-WF algorithm on a large-scale network, we develop a simulator on a datacenter network.

4.1 Simulation Methodology

Topologies: Our experiments were conducted by emulating a 3-tier datacenter network topology with 8:1 oversubscription. The topology contains 64 servers; the capacity of each edge link is $1Gbps$; the capacity of the aggregated link is $10Gbps$.
Workloads: We synthesize a stream of transmission requests with a total number of 1000. A Poisson process is used to model the arrival of requests; the arrival rate λ is defined as the average number of new transfers per time slot. We set the slot length to be only a second for fast simulation. A transfer has multiple sources with probability ρ. We assume that a multi-source transfer have a random number of replicas between [2,5], which are randomly placed in servers. We ignore the fluctuation of transfer size in the simulations, and assume a uniform size V for all transfers.
Performance metrics: We use *average transfer completion time* and *network throughput*to show the improvements of MS-WF.
Alternative approaches: We compare the following bandwidth allocation approaches, each of which adopts the traditional WF algorithm.

- **Best-source:** This approach selects a best replica source based on the algorithm in [16] for multi-source transmission.
- **Equal-share:** This approach splits a transfer across different sources equally. For instance, each replica will send 1/3 of the data, if a transfer has 3 replicas.
- **Random-source:** This approach randomly selects an available source for each transfer.

4.2 Simulation Results

Figure 4 displays the simulation results of network throughput, for which the arrival rate $\lambda = 2$ and the data size $V = 10Gbits$ for all of the 1000 transfers. Shown by the results, Random-source approach, disregarding the dissimilarity of the sources, performs the worst, and retains a constant throughput value. Equal-share approach takes advantage of source diversity in a naïve manner; when we have limited diversity to a small number of multi-source transfers (at low multi-source probabilities), equal flow sharing can approximate the optimal assignment, therefore obtaining a similar performance as MS-WF. But as the multi-source proportion rises, the 1000 transfers lead to more potential flows. The effect of "bad flows" enlarges, and therefore pulls down the overall throughput improvement. This way, Best-source approach outperforms Equal-share. MS-WF achieves a much higher throughput than the others by jointly optimizing the bandwidth allocation and flow assignment, leading to higher network utilization. When all transfers have multiple sources ($\rho = 1$), compared with the Random-source transmission, MS-WF obtains a substantial throughput gain for up to 52%.

Figure 5 compares the transfer completion time versus three factors respectively: the multi-source probability ρ, the transfer size V and the transfer arrival rate λ. We observe directly from the figure: across all parameter configurations, MS-WF achieves the smallest transfer completion time. This improvement is mainly acquired from a more max-min fair allocation by MS-WF.

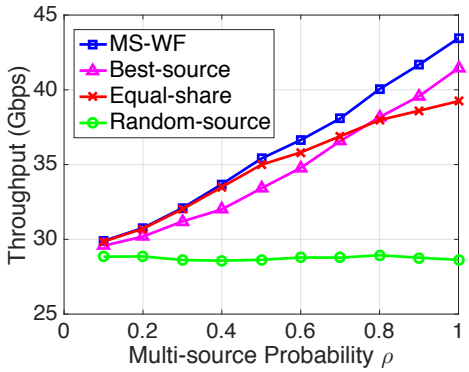

Figure 4: Network throughput vs. multi-source probability ρ, where the arrival rate $\lambda = 2$ and the data size $V = 10Gbits$.

Figure 5(a) illustrates the impact of ρ, which approximately equals to the proportion of multi-source transfers. The constant gap between Random-source and the other approaches demonstrates that, leveraging multiple sources is more efficient for data transmission and can perform much better by placing more replicas in the network. Compared to single-source transmission, MS-WF cuts the average completion time down by up to 44%. Figure 5(b) shows the relationship with V. As transfer size goes up, the transfer backlog begins to cause more bottleneck links, which results in degradation of transmission rates. Therefore, the completion time surges super-linearly along with the transfer size for all the allocation approaches. But by completing transfers as quickly as possible, MS-WF is able to achieve almost linear completion time growth. This way, it is capable of optimizing data transfers for both small and large files. Figure 5(c) illustrates the impact of λ. As expected, at higher arrival rates, the number of flows is likely to be increased, and links are more likely to become congested. Accordingly, the performance degrades quickly for all approaches other than MS-WF. The smaller growth in completion time demonstrates that, by effectively avoiding the congestion point, MS-WF manages to handle a relatively larger amount of traffic without degrading the performance.

5 RELATED WORK

Distributed filesystems. Several high-performance distributed filesystems with sufficient data replicas have been developed, including GFS [7], HDFS [17] and Quantacast File System [15]. Leveraging SDN, Mayflower [16] performs global optimizations to make intelligent replica selection and flow scheduling decisions based on both filesystem and network information. Nevertheless, current solutions focus heavily on best replica selection and data replication placement, instead of multi-source transmission as in our work. As shown in the proceeding sections, single-source transmission fails to achieve transfer-level max-min fairness, therefore provides sub-optimal performance.
Coflow scheduling. The works that schedule parallel flows have been developed to optimize transfers at the level of coflow rather than individual ones. Coflow [3], Varys [4] and Barrat [6] improve

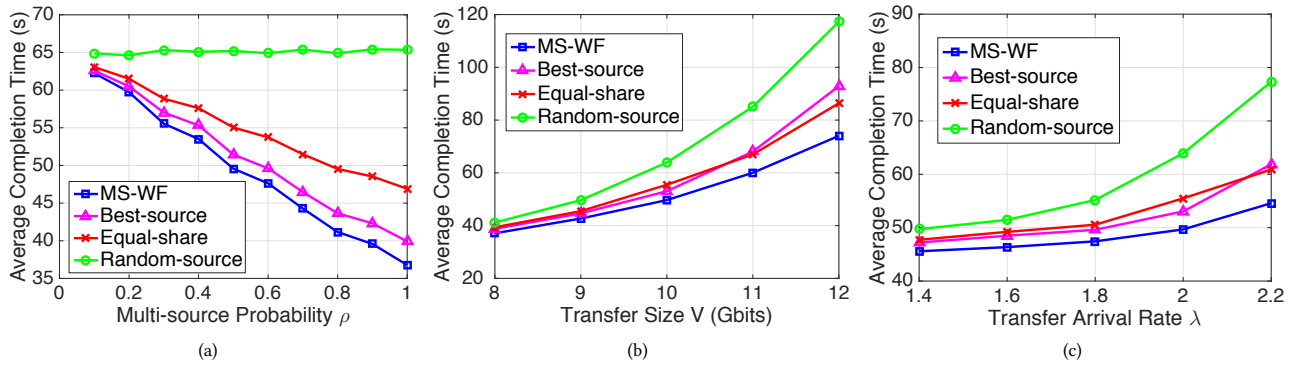

Figure 5: Impact of (a) the multi-source probability ρ, (b) the data size V and (c) the transfer arrival rate λ on average completion time. In each subfigure, we adjust one factor and fixed the other two. The three default values are $\rho = 0.5$, $V = 10$ and $\lambda = 2$.

application-level performance by minimizing coflow completion times and guaranteeing predictable completions. However, their basic assumption is that the flows are streamed for different data and the volume of each flow is designated in advance, so they can easily predict the completion time and allocate the rate to meet their deadlines. The improvement of our approach over them is that the flow volume assignment is jointly optimized with bandwidth allocation to achieve global optimality.

6 CONCLUSION

We present a novel max-min fair allocation approach for multi-source transmission which conveys data in parallel from multiple sources and dynamically adjusts the flow volumes to maximize network utilization. The allocation relies on a MultiSource Water-Filling algorithm that jointly computes the bandwidth allocation and flow assignment with simple equivalent canonical LP to achieve global optimality. Extensive simulations validate that, compared to other single-source and multi-source allocation approaches, our approach achieves a better throughput gain of up to 52% and decreases transfer completion time by up to 44% for large-scale transfers. We believe this approach is applicable to various traffic management systems that orchestrate arbitrary bulk transfers. Developing such systems will be the next step of this research.

ACKNOWLEDGMENT

This research was supported in part by NSFC #61701347, NSFC #61702373 and NSFC #61672385; NSF grant #1440745, CC*IIE Integration; Google Research Award, SDN Programming Using Just Minimal Abstractions. This research was also sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

REFERENCES

[1] Anoymous Technical Report. https://github.com/technical-report-2018/CCR2018.
[2] Mosharaf Chowdhury, Srikanth Kandula, and Ion Stoica. Leveraging endpoint flexibility in data-intensive clusters. In *ACM SIGCOMM CCR*, volume 43, 2013.
[3] Mosharaf Chowdhury and Ion Stoica. Coflow: A networking abstraction for cluster applications. In *Proceedings of HotNet*, 2012.
[4] Mosharaf Chowdhury, Yuan Zhong, and Ion Stoica. Efficient coflow scheduling with varys. In *ACM SIGCOMM CCR*, 2014.
[5] CMS Collaboration et al. The cms experiment at the cern lhc (journal of instrumentation 3 s08004, 2008) p. 158.
[6] Fahad R Dogar, Thomas Karagiannis, Hitesh Ballani, and Antony Rowstron. Decentralized task-aware scheduling for data center networks. In *ACM SIGCOMM CCR*, 2014.
[7] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. In *ACM SIGOPS operating systems review*, 2003.
[8] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. Achieving high utilization with software-driven wan. In *SIGCOMM CCR*, 2013.
[9] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, et al. B4: Experience with a globally-deployed software defined wan. *SIGCOMM CCR*, 2013.
[10] Xin Jin, Yiran Li, Da Wei, Siming Li, Jie Gao, Lei Xu, Guangzhi Li, Wei Xu, and Jennifer Rexford. Optimizing bulk transfers with software-defined optical wan. In *Proceedings of SIGCOMM 2016 Conference*.
[11] Alok Kumar, Sushant Jain, Uday Naik, Anand Raghuraman, Nikhil Kasinadhuni, Enrique Cauich Zermeno, C Stephen Gunn, Jing Ai, Björn Carlin, Mihai Amarandei-Stavila, et al. Bwe: Flexible, hierarchical bandwidth allocation for wan distributed computing. In *SIGCOMM CCR*, 2015.
[12] Geng Li, Yichen Qian, Lili Liu, and Y Richard Yang. Jms: Joint bandwidth allocation and flow assignment for transfers with multiple sources. In *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. IEEE, 2018.
[13] Qingming Ma, Peter Steenkiste, and Hui Zhang. Routing high-bandwidth traffic in max-min fair share networks. In *Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*, 1996.
[14] Dritan Nace and Michal Pioro. Max-min fairness and its applications to routing and load-balancing in communication networks: a tutorial. *IEEE Communications Surveys & Tutorials*, 10(4):5–17, 2009.
[15] Michael Ovsiannikov, Silvius Rus, Damian Reeves, Paul Sutter, Sriram Rao, and Jim Kelly. The quantcast file system. *Proceedings of the VLDB Endowment*, 2013.
[16] Sajjad Rizvi, Xi Li, Bernard Wong, Fiodar Kazhamiaka, and Benjamin Cassell. Mayflower: Improving distributed filesystem performance through sdn/filesystem co-design. In *Distributed Computing Systems (ICDCS), 2016 IEEE 36th International Conference on*, 2016.
[17] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on*, 2010.
[18] Qiao Xiang, X. Tony Wang, J. Jensen Zhang, Harvey Newman, Y. Richard Yang, and Y. Jace Liu. Unicorn: Resource Orchestration for Multi-Domain, Geo-Distributed Data Analytics. https://datatracker.ietf.org/doc/draft-xiang-alto-multidomain-analytics, 2018.