Two-Aggregator Topology Optimization without Splitting in Data Center Networks

Soham Das and Sartaj Sahni
Department of Computer and Information Science and Engineering
University of Florida
Gainesville, USA
Email: {sdas, sahni}@cise.ufl.edu

***Abstract*—Data aggregation is a critical operation in many big-data applications; for example, data residing in several source racks (mappers) are to be aggregated into one or more specified racks called aggregators (reducers) in the data center network during the shuffle phase of a map-reduce task. In this paper, we explore algorithms for data aggregation to two aggregators in a data center network under the constraint that data from a source rack must be routed to each aggregator using a single path. We derive bounds on the approximation ratios of two classes of aggregation algorithms– Restricted 1-Round (R1R) and Restricted 2-Round (R2R). For the case when racks have exactly 2 optical links (uplinks in Top-of-Rack switches), we propose another strategy using the 2Chain topology for aggregation and show that the optimal 2Chain cannot have an aggregation time greater than that of the optimal R1R and R2R topologies. For the case when racks have at least 4 optical links, we propose a 1-round aggregation algorithm (1R) that uses tree topology for aggregation. Experimental results indicate that, when racks have 4 optical links, 1R, R2R and R1R reduce the aggregation time by up to 85%, 67% and 67% respectively, relative to the two-round aggregation algorithm proposed by Wang et al. Moreover, the 2Chain can reduce the aggregation time up to 42% and 24% respectively relative to R1R and R2R, when racks have exactly 2 optical links.**

Keywords*—*Data center networks, software defined networking, big data applications, map-reduce tasks

I. Introduction

Thousands of server racks are interconnected using top-of-rack (ToR) switches to form large data center networks. For large-scale big-data applications, these networks can be dynamically reconfigured using software defined networking (SDN) in negligible time compared to the total execution time of the application. Recent research has shown how such a reconfiguration can significantly enhance the performance of a big-data application [1]. Data aggregation is a critical operation in many big-data applications that employ paradigms such as the Map-Reduce. In these paradigms data residing in several source racks (mappers) are to be aggregated into one or more specified racks called aggregators (reducers). Wang et al. [1] have observed that the aggregation time is a dominant component of the overall execution time in many big-data applications. Given the application (i.e. the amount of data in each source rack), this paper focuses on determining an optimal topology of the data center that minimizes aggregation time (the network can be reconfigured to the determined topology using SDN) when there are two aggregators and data from each source rack can be routed to an aggregator using only a single path. The degree of each rack in the data center network is constrained by the number k of optical links at each Top of rack (ToR) switch. We have addressed the single aggregator version of the problem in our previous work [2]. As a first step towards generalizing the problem to multi-aggregator topologies, we focus on the problem with two aggregators here.

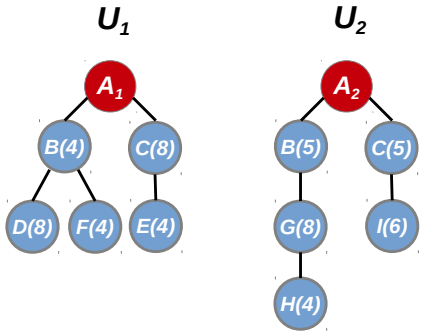


Fig. 1: Example aggregation tree topologies

We illustrate the two-aggregator data aggregation problem using the small example of Fig. 1. There are a total of eight source racks, each denoted by $X(d)$, $X \in \{B, C, D, E, F, G, H, I\}$, where d units of data are to be aggregated from the source rack X into the corresponding aggregator and two aggregator racks A_1 and A_2. Fig. 1 shows a possible aggregation tree topology that may be used by each of the 2 aggregators. Among the source racks, B and C have data to send to both aggregators and so, these racks are present in both the aggregation trees U_1 and U_2. Racks D, E and F have data to send to only A_1 and racks G, H and I have data to send to only A_2 and so, these racks are present in only one aggregation tree. The edges denote optical links between pairs of ToR switches. The shown aggregation trees can be realized when $k \geq 5$ as rack B uses 3 and 2 optical links, respectively, in the two trees.

We assume that the data is packetized and nodes can send and receive packets at the same time using multiple optical links. A node sends its own data and the data aggregated from the subtree below it to its parent, the parent node in turn sends its data along with the data aggregated from the subtree below it to its parent and in this way data from all the sources finally reach the aggregators. Note that each source rack sends its data through a unique path to the corresponding aggregator. In Fig. 1, if each optical link has a bandwidth of 10 units per second, the data from B, D, and F can be aggregated into A_1 in $(4 + 8 + 4)/10 = 1.6s$ and that from C and E can be

aggregated in $(8+4)/10 = 1.2s$. Since A receives data from B and C in parallel, the aggregation time is $\max\{1.6, 1.2\}$ = $1.6s$. Similarly, the data from B, G, and H can be aggregated into A_2 in $(5+8+4)/10 = 1.7s$ and that from C and I can be aggregated in $(5+6)/10 = 1.1s$. Since A_2 receives data from B and C in parallel, the aggregation time is $\max\{1.7, 1.1\}$ = $1.7s$. So the total aggregation time is $\max\{1.6, 1.7\} = 1.7s$.

In the two-aggregator network topology optimization problem (TANTO), our objective is to minimize the overall aggregation time, given the degree k of ToR switches, set of source racks S_1 that send data to only A_1, set of source racks S_2 that send data to only A_2, and set of source racks C that send data to both A_1 and A_2 along with the corresponding units of data to be sent; each source rack can send its data through a single path to an aggregator. The single aggregator version of the problem (SANTO) has been proved to be NP-hard in [2]. Hence, it is evident that TANTO is NP-hard.

Previously, Wang et al. [1] have proposed a 2-round[1] algorithm, W2R, to solve TANTO. In round 1, W2R uses a tree to aggregate data from S_1 to A_1 and another one to aggregate from S_2 to A_2 in parallel. In round 2, W2R uses a 2-D torus to aggregate data from C to both A_1 and A_2. The round 2 torus topology of [1] requires $k \geq 4$ and so cannot be used when $k < 4$.

In this paper, we study two classes of algorithms for TANTO- Restricted 1-round (R1R) and the Restricted 2-Round (R2R). These classes are motivated by the structure of W2R. In R1R, the aggregations $S_1 \rightarrow A_1$, $S_2 \rightarrow A_2$, and $C \rightarrow \{A_1, A_2\}$ are performed in parallel in a single round under the restriction that we use $k_i < k$ links of A_i to aggregate $S_i \rightarrow A_i$ and the remaining $k - k_i$ links to aggregate $C \rightarrow A_i$. As a result, each link of A_i carries data for either racks in S_i or racks in C (but not both). In R2R, the aggregations $S_1 \rightarrow A_1$ and $S_2 \rightarrow A_2$ are performed in parallel in round 1. In round 2, the aggregation $C \rightarrow \{A_1, A_2\}$ is performed. The two rounds are executed serially. So in, each round the aggregators have all k links available. The algorithm W2R of [1] is an example of an R2R algorithm. However, since W2R uses a torus for round 2, W2R cannot be used when $k < 4$. For $k = 2$, we propose another strategy using the 2Chain topology for aggregation and show that the optimal 2Chain cannot have an aggregation time greater than that of the optimal R1R and R2R topologies. Our results lay the foundations for studying the generic m aggregator network topology optimization problem in future.

The main contributions of this work are listed below:

1) We establish the existence of TANTO instances for which the optimal R1R and R2R aggregation times are twice that of the true optimal.
2) We show that for every TANTO instance, the optimal R1R and R2R aggregation times are at most 4 and $11/3$, respectively, times the true optimal when $2 \leq k \leq 3$. For the case $k \geq 4$, we prove a tight bound of 2 on this approximation ratio for both R1R and R2R.
3) For $k = 2$, we propose another strategy using the 2Chain topology for aggregation and show that the optimal 2Chain cannot have an aggregation time greater than that of the optimal R1R and R2R topologies.
4) For $k \geq 4$, we propose solving TANTO using an 1-round (1R) algorithm, based on the LPT (Longest Processing Time) scheduling rule [2], [5], [6].
5) We show via experimentation that, for $k = 4$, 1R, R2R and R1R can reduce aggregation time by up to 85%, 67% and 67% relative to W2R [1]. Moreover, for $k = 2$, the 2Chain can reduce the aggregation time up to 42% and 24% respectively relative to R1R and R2R.

The remainder of this paper is organized as follows. Related work is reviewed in Section II. In Section III, we derive bounds on the optimal aggregation times using R1R and R2R besides results using the 2Chain topology. In Section IV, we describe our 1-round algorithm for TANTO. Experimental results are presented in Section V and we conclude in Section VI.

II. Related Work

Recently, there has been much interest in studying new data center network architectures with the goal of making them more energy efficient [7]. The classic data-center design architecture [8], switch-centric architectures like the one by Al-Fares et. al [9], VL2 [10], the Juniper Qfabric architecture [11], server-centric architectures like BCube [12] and DCell [13] need special mention. Recent research has focused on optical interconnect schemes; Helios [14], C-Through [15], OSA [16] have made significant contributions. Das et al. [17] explore the use of OpenFlow to control routing according to application need and Webb et al. [18] propose to isolate applications and use different routing mechanisms for them in fat-tree based data-centers. Among full optical data centers, some recent architectures, that are of significant importance are Petabit [19], [20], DOS [21], Proteus [22] etc.But the main drawback is the fact that these fully optical architectures require a complete changeover of current data centers. The hybrid architectures reconfigure the network relying on network-level statistics to suit the application, but utilization and application performance can be poor unless we have a true application-level view of traffic demands and dependencies [1], [23].

Our work is motivated by that of Wang, Ng and Shaikh [1]. In [1], Wang et al. describe an "integrated network control for big-data applications" that comprise "OpenFlow-enabled top-of-rack (ToR) switches". They propose heuristics for the single as well as multiple aggregators variants of the network topology optimization problem. In [2], we address single aggregator network topology optimization (SANTO) under the constraint that nodes cannot split their data over multiple paths. The problem is shown to be NP-hard and the approximation ratio of the algorithm of Wang, Ng, and Shaikh [1] is shown to be $(k+1)/2$, where k is the degree of ToR (top-of-rack) switches. We propose a SANTO algorithm that is based on the longest processing time (LPT) scheduling rule, approximation ratio of $(4/3 - 1/(3k))$ [5], [6]. We also prove that the aggregation time using the LPT method is never more than that using the algorithm of [1]. Wang et al. [1] propose a two-round algorithm, W2R, for TANTO which works for $k \geq 4$. They, however, do not establish any properties for this algorithm. In [3], [4], we study one and two-aggregator network topology optimization for the case when data splitting is permitted; i.e., when data from a single source may be split across multiple paths on the way to the aggregator.

[1]In a round, the network topology is fixed.

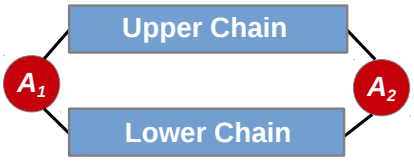


Fig. 2: 2Chain topology

III. Performance Bounds For Restricted Aggregation Topologies

We begin by introducing some of the notations we will be using. We use I to denote an instance of TANTO. $OPT(I,k)$ denotes the overall optimal aggregation time, where k is the maximum degree of a ToR switch. $OPT(R1R,I,k)$ and $OPT(R2R,I,k)$ denote, respectively, the optimal aggregation time for R1R and R2R.

Before proving the bounds, we first prove a result in Lemma 1 that will be used in later proofs. For simplicity of the equations, we have assumed the bandwidth to be 1 unit so that the data transmission time equals the amount of data transmitted through a link.

Lemma 1. *The aggregation time for every aggregation strategy for I is $\geq \max\{D_i(j)\}$, where $D_i(j)$ is the amount of data aggregated through link j of A_i, $1 \leq j \leq k$ and $1 \leq i \leq 2$ over all aggregation rounds.*

Proof: Follows from the definition of aggregation time. ■

Several of our subsequent proofs will employ a 2Chain topology which is comprised of two chains (upper and lower) with the two aggregators A_1 and A_2 at the two ends and source racks in between (Fig. 2), each rack using 2 links. 2Chain may include all racks in $S_1 \cup S_2 \cup C$. Note that a 2Chain is not an R1R or R2R topology. However, a 2Chain that is limited to the racks of C, for example, is a valid R1R and R2R topology. We will show results comparing the 2Chain, R1R and R2R strategies (for $k = 2$) in the subsections to follow.

A. Restricted One Round Aggregation (R1R)

Before going into proving bounds for R1R, we first compare the performance of 2Chain with respect to R1R for $k = 2$ in Theorem 1.

Theorem 1. $OPT(2Chain,I,2) \leq OPT(R1R,I,2)$, *for all* I.

Proof: Consider an optimal R1R topology for I as shown in Fig. 3. Let $t(S_1)$, $t(S_2)$ and $t(C)$ be the aggregation times of the three topologies of R1R. So, $OPT(R1R,I,2) = \max\{t(S_1),t(S_2),t(C)\}$. We assign the racks in S_1 and S_2 to the upper chain of a 2Chain R and those in C to the lower chain, as shown in Fig. 3. Data from S_1 and S_2 are aggregated in parallel into the aggregators through upper chain, and data from the racks in C are aggregated exactly in the same way as in the optimal R1R topology for racks in C. So, the aggregation time using the 2Chain R is $\max\{t(S_1),t(S_2),t(C)\} = OPT(R1R,I,2)$. Hence, $OPT(2Chain,I,2) \leq OPT(R1R,I,2)$, for all I. ■

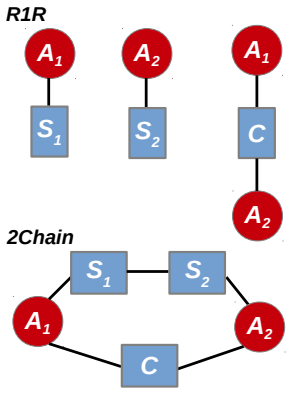


Fig. 3: 2Chain corresponding to an R1R $k = 2$ topology

Next, we prove bounds of the R1R aggregation strategy.

Theorem 2. *For every k, $k \geq 2$, and integer x, $x > 1$, there is an instance $I(x)$ of TANTO such that $OPT(R1R,I(x),k)/OPT(I(x),k) = 2 - 1/x$.*

Proof: The proof is omitted due to lack of space [24]. ■

Note that as $x \to \infty$, the ratio of Theorem 2 approaches 2. We show below, in Theorem 5, that for $k \geq 4$, there is no instance I for which $OPT(R1R,I,k)/OPT(I,k) > 2$. Hence the bound of Theorem 5 is tight. First, we establish bounds for $k = 2$ and 3.

Theorem 3. $OPT(R1R,I,2) \leq 4OPT(I,2)$ *for every* I.

Proof: The proof is omitted due to lack of space [24]. ■

Theorem 4. $OPT(R1R,I,3) \leq 4OPT(I,3)$ *for every* I.

Proof: The proof is omitted due to lack of space [24]. ■

Theorem 5. $OPT(R1R,I,k) \leq 2OPT(I,k)$ *for every I and every k, $k \geq 4$.*

Proof: The proof is omitted due to lack of space [24]. ■

B. Restricted Two Round Aggregation (R2R)

Before going into proving bounds for R2R, we first compare the performance of 2Chain with respect to R2R for $k = 2$ in Theorem 6.

Theorem 6. $OPT(2Chain,I,2) \leq OPT(R2R,I,2)$, *for all* I.

Proof: Consider an optimal R2R topology for I. This topology is comprised of a round 1 topology Y that aggregates from the S_is to the A_is and a round 2 topology Z that aggregates from the racks in C to the A_is. Let $S_i^{R2R}(j)$ be the subset of S_i that sends its data to A_i through the j^{th} link of A_i using the topology Y and let $d(S_i^{R2R}(j))$ be the sum of data in the racks of $S_i^{R2R}(j)$. The round 1 aggregation time using Y is, therefore, $\geq \max\{d(S_1^{R2R}(1)),d(S_1^{R2R}(2)),d(S_2^{R2R}(1)),d(S_2^{R2R}(2))\}$ by Lemma 1. Since $k = 2$, Z must be a 2Chain with A_1 and A_2 at the two ends and the racks of C in between. Let $C^{R2R}(1)$ be the subset of racks in C in the upper chain and $C^{R2R}(2)$

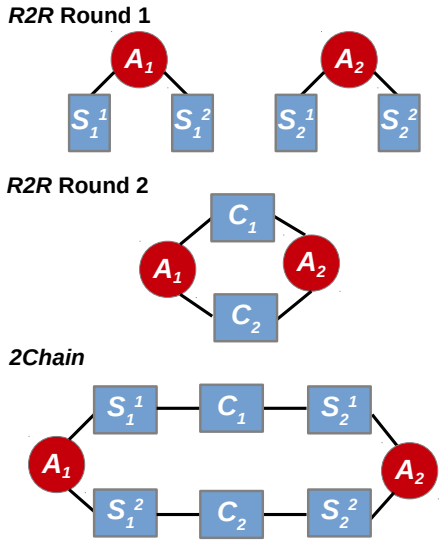


Fig. 4: 2Chain corresponding to an R2R $k = 2$ topology

be those in the lower chain. The aggregation time for C is $max\{tc(1), tc(2)\}$ by Lemma 1, where $tc(1)$ and $tc(2)$ are, respectively, the aggregation times of the upper and lower chains (see Fig. 4).

Consider the 2Chain of Fig. 4. Data is routed from $\{S_1^{R2R}(1), S_1^{R2R}(2)\}$ to A_1 and from $\{S_2^{R2R}(1), S_2^{R2R}(2)\}$ to A_2 using a standard chain routing strategy. Data is routed from $\{C^{R2R}(1), C^{R2R}(2)\}$ to $\{A_1, A_2\}$ via the $S_i^{R2R}(j)$ using the strategy employed in Z. The packets to be aggregated from $C^{R2R}(1)$ to A_1 get to A_1 at most $d(S_1^{R2R}(1))$ units later in the 2Chain than in Z (note that since each rack of S_1 has at least 1 unit of data, $d(S_1^{R2R}(1)) \geq |S_1^{R2R}(1)|$). The same is true for the remaining combinations of C^{R2R} and A_is. So, $OPT(2Chain, I, 2)$
$\leq \max\{d(S_1^{R2R}(1)) + tc(1), d(S_1^{R2R}(2)) + tc(2), d(S_2^{R2R}(1)) + tc(1), d(S_2^{R2R}(2)) + tc(2)\}$ by Lemma 1
$\leq \max\{d(S_1^{R2R}(1)), d(S_1^{R2R}(2)), d(S_2^{R2R}(1)), d(S_2^{R2R}(2))\} + \max\{tc(1), tc(2)\}$
$= OPT(R2R, I, 2)$. ■

Next, we prove bounds of the R2R aggregation strategy.

Theorem 7. *For every k, $k \geq 2$, there is an instance I for which $OPT(R2R, I, k) = 2OPT(I, k)$.*

Proof: The proof is omitted due to lack of space [24]. ■

We show below, in Theorem 10, that for $k \geq 4$, there is no instance I for which $OPT(R2R, I, k)/OPT(I, k) > 2$. Hence the bound of Theorem 10 is tight. First, we establish bounds for $k = 2$ and 3.

Theorem 8. $OPT(R2R, I, 2) \leq 11/3OPT(I, 2)$ *for every I.*

Proof: The proof is omitted due to lack of space [24]. ■

Theorem 9. $OPT(R2R, I, 3) \leq 11/3OPT(I, 3)$*, for all I.*

Proof: The proof is omitted due to lack of space [24]. ■

Theorem 10. $OPT(R2R, I, k) \leq 2OPT(I, k)$ *for every I and every $k \geq 4$.*

Proof: The proof is omitted due to lack of space [24]. ■

IV. The 1-round Algorithm 1R

We propose a 1-round algorithm, 1R, for $k \geq 4$ here. This algorithm, which is specified in Algorithm 1, employs 2 aggregation trees U_1 and U_2 to do the aggregation for A_1 and A_2, respectively. Source racks are first assigned to the k subtrees of U_1 and U_2 using the longest processing time (LPT) rule [5], [6]. A rack that aggregates data to only A_1 (A_2) is assigned to a single subtree of A_1 (A_2) while one that aggregates to both A_1 and A_2 is assigned to one subtree in $U1$ and one in U_2. The racks, assigned to each subtree of U_1 (U_2) are connected to form a chain with A_1 (A_2) as the root of the tree. To construct the two trees U_1 and U_2, we use all k links of A_1 and A_2 and at most 4 links of each source rack (2 in each tree). The unutilized links of the source racks, if any, could be used to optimize secondary measures such as link utilization (see [2]). We do not explore this here.

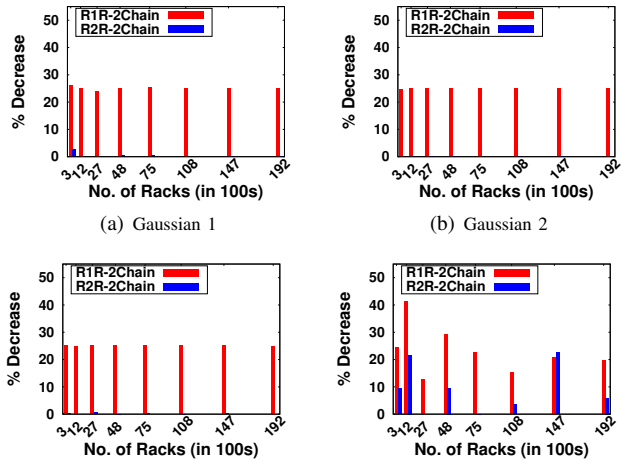
Algorithm 1 One-round Algorithm

Input: S_1, S_2, C, $k \geq 4$.
Output: Aggregation trees U_1 and U_2.
1: **for** each aggregator A_i **do**
2: Sort racks in S_i, C in decreasing order of data for A_i
3: **for** each rack in decreasing order **do**
4: Assign the rack to the subtree of U_i which has the minimum assigned data so far.
5: **end for**
6: Connect racks in each subtree of U_i into a chain with A_i as the root of U_i.
7: **end for**

V. Experiments

The distribution of the amount of data available at each source rack is application specific and may vary widely depending on the particular application. For example, if we are scanning a big text corpus distributed in the network and returning frequencies of a set of keywords for each document, we have the same amount of data to aggregate from each source rack. On the other hand, if we are executing a search query on the same text corpus and returning the documents matching the keywords, the amount of data to be sent to the aggregators from each source rack may vary to a large extent. Since TANTO has the constraint that data from a source rack must be routed to each aggregator using a single path, for racks with large amounts of data, all the data for an aggregator has to reach the latter through exactly one of it's uplink. This increases the aggregation time significantly (Lemma1). So, in order to assess the performance of our algorithms in such real scenarios, we used the following data sets:

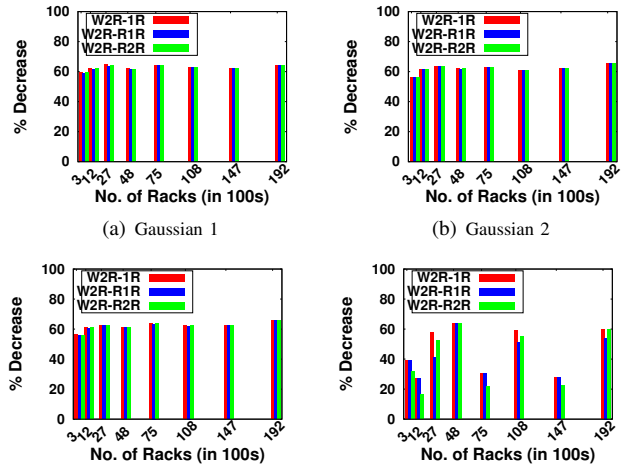
- *Gaussian 1.* Amounts of data in source racks are drawn from a truncated Gaussian distribution with mean 500 and standard deviation 1000 truncated in $[200, 800]$. An alternate data-set
- *Gaussian 2.* Amounts of data in source racks are drawn from a truncated Gaussian distribution with mean 500 and standard deviation 1000 truncated in $[400, 600]$.



(a) Gaussian 1 (b) Gaussian 2

(c) Uniform (d) Zipfian

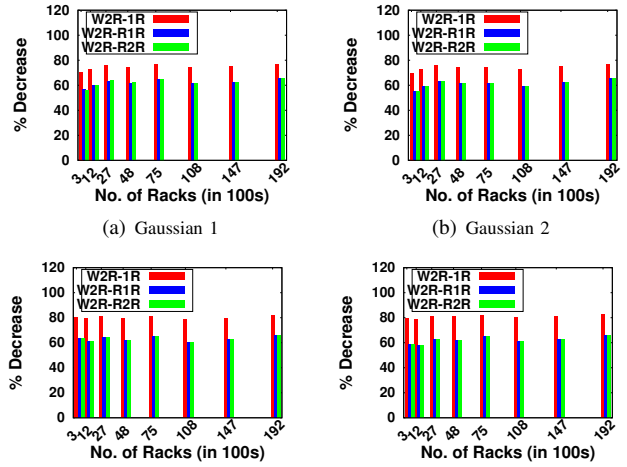
Fig. 5: Average percentage decrease in aggregation time (2Chain vs R1R, R2R), $k = 2$



(a) Gaussian 1 (b) Gaussian 2

(c) Uniform (d) Zipfian

Fig. 6: Average percentage decrease in aggregation time (1R, R1R, R2R vs W2R), $k = 4$



(a) Gaussian 1 (b) Gaussian 2

(c) Uniform (d) Zipfian

Fig. 7: Average percentage decrease in aggregation time (1R, R1R, R2R vs W2R), $k = 4$

- *Uniform.* Amounts of data in source racks are drawn from a uniform distribution with values in [50,100].
- *Zipfian.* Amounts of data in source racks are drawn from a Zipfian distribution with parameter 2.

The link bandwidth is assumed to be 1 unit so that the data transmission time equals the amount of data transmitted through a link. In our experiments, we took $k = 2$ and $k = 4$ to compare our algorithms with the 2Chain and W2R [1] respectively. We varied the total number of source racks ($|S_1|+|S_2|+|C|$) from 300 to 19200, we have assumed $|S_1| = |S_2| = |C|$. Our experiments were conducted on a 64-bit PC with a 2.80 GHz AMD Athlon(tm) II X2 B22 processor and 8GB RAM.

Fig. 5 shows the average percentage decrease in aggregation time of the 2Chain compared to that of R1R and R2R respectively, $k = 2$. In R1R, we assigned 1 link of each aggregator A_i to racks in S_i and the other link to the racks in C. In round 1 of R2R, we assign racks using the longest processing time first rule (LPT) to the two subtrees of the aggregation tree. In round 2, we create a 2Chain topology with each rack using 2 links, and racks are assigned to the upper and lower chains of the 2Chain using LPT on the total amount of data in each rack for the two aggregators. The maximum reduction in aggregation time by the 2Chain compared to R1R was up to 42%, 27%, 26% and 26% for the Zipfian, Gaussian 1, Gaussian 2 and Uniform data-sets, respectively. The maximum reduction in aggregation time by the 2Chain compared to R2R was up to 24%, 4%, 2% and 1% for the Zipfian, Gaussian 1, Gaussian 2 and Uniform data-sets, respectively.

Fig. 6 shows the average percentage decrease in aggregation time by 1R, R1R and R2R respectively compared to W2R [1] when $k = 4$. In R1R, we assigned 2 links of each aggregator A_i to racks in S_i and the other two links to the racks in C. For S_is, racks are assigned to subtrees using LPT rule. Now, since each rack has 4 links, we form a 2Chain topology for aggregating data from racks in C with each chain having two parallel paths, one to A_1, the other to A_2, each source rack using 4 links. So aggregation to A_1 and A_2 can be done in parallel in each chain. Racks in C are assigned to the chains using a modified version of LPT on the total amount of data in each rack for the two aggregators - each source rack in C is assigned to the chain, where it increases the aggregation time (maximum of the aggregation times of A_1 and A_2 in that chain) the least. In round 1 of R2R, we assign racks using the standard LPT rule. In round 2, we create an extended 2Chain topology with four chains (4Chain), each chain having parallel paths to the two aggregators with each rack using 4 links. Racks are assigned to the four chains using the modified LPT, as discussed earlier. The maximum reduction in aggregation time by 1R, compared to W2R was up to 67%, 67%, 67% and 64% for the Gaussian 1, Gaussian 2, Uniform and Zipfian data-sets, respectively. The maximum reduction in aggregation time by R1R, compared to W2R was up to 67%, 67%, 65% and 64% for the Uniform, Gaussian 2, Gaussian 1 and Zipfian data-sets, respectively. The maximum reduction in aggregation time by R2R, compared to W2R was up to 68%, 67%, 65% and 64% for the Uniform, Gaussian 2, Gaussian 1 and Zipfian data-sets, respectively.

To vary the balance among the amounts of data in racks of S_1, S_2 and C, in Fig. 7, we used a variant of each of the original data-sets, where we scaled up the data for racks in S_1 and C_2, while those in racks in S_2 and C_1 are being drawn from the origin distributions. For Uniform, we have drawn data

for S_1 and C_2 from [500,1000], for the other distributions, we have scaled these up by adding 1000 units to racks in S_1 and C_2. Fig. 7 shows the average percentage decrease in aggregation time by 1R, R1R and R2R algorithms compared W2R [1]. As expected, the maximum reduction in aggregation time by 1R, compared to W2R goes up to 85%, 82%, 78% and 78% for the Zipfian, Uniform, Gaussian 1 and Gaussian 2 data-sets, respectively. The maximum reduction in aggregation time by R1R, compared to W2R was up to 67%, 67%, 67% and 65% for the Gaussian 2, Uniform, Zipfian and Gaussian 1 data-sets, respectively. The maximum reduction in aggregation time by R2R, compared to W2R was up to 67%, 67%, 67% and 65% for the Gaussian 2, Uniform, Zipfian Gaussian 1 data-sets, respectively.

VI. Conclusion

In this paper, we have focused on the two-aggregator network topology optimization problem without splitting (TANTO). We have proposed solving TANTO using two classes of algorithms- R1R and R2R, which unlike the existing W2R algorithm do not require k, the degree of a ToR switch, to be ≥ 4. We proved that both R1R and R2R have an approximation ratio of at least 2. We derived upper bounds of 4 for $k = 2$ and $k = 3$ and 2 for $k \geq 4$ for R1R and upper bounds of $11/3$ for $k = 2$ and $k = 3$ and 2 for $k \geq 4$ for R2R. We have also proposed solving TANTO using a 1-round algorithm (1R), by constructing two aggregation trees one for each aggregator and then using the LPT (Longest Processing Time) scheduling rule to place racks in the aggregation trees. This algorithm requires $k \geq 4$. For $k = 2$, we propose another strategy using the 2Chain topology for aggregation and show that the optimal 2Chain cannot have an aggregation time greater than that of the optimal R1R and R2R topologies. Experimental results indicate that, when $k = 4$, 1R, R2R and R1R reduce the aggregation time by up to 85%, 67% and 67% respectively, relative to the two-round aggregation algorithm proposed by Wang et al [1]. Moreover, for $k = 2$, the 2Chain can reduce the aggregation time up to 42% and 24% respectively relative to R1R and R2R. Our results lay the foundations for studying the generic m aggregator network topology optimization problem in future.

Acknowledgment

This research was supported, in part, by the National Science Foundation under grant CNS0905308.

References

[1] Wang, Guohui and Ng, T.S. Eugene and Shaikh, Anees, *Programming your network at run-time for big data applications*, Proceedings of the first workshop on Hot topics in software defined networks HotSDN '12, 2012.

[2] Das, Soham and Sahni, Sartaj *Network topology optimization for data aggregation*, IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid), 2014.

[3] Das, Soham and Sahni, Sartaj *Network topology optimization for data aggregation with Splitting*, IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2014.

[4] Das, Soham and Sahni, Sartaj *Network topology optimization for data aggregation using Multiple Paths*, International Journal on Metaheuristics (IJMHeur), 2015, pages 115–140

[5] Graham, R. L. *Bounds on Multiprocessing Timing Anomalies*, SIAM JOURNAL ON APPLIED MATHEMATICS, 1969, volume 17, number 2, pages 416–429.

[6] Coffman,Jr., E. G. and Sethi, Ravi, *A generalized bound on LPT sequencing*, Proceedings of the 1976 ACM SIGMETRICS conference on Computer performance modeling measurement and evaluation, SIGMETRICS '76, 1976.

[7] Hammadi, A. and Mhamdi, L., *A survey on architectures and energy efficiency in data center networks*, Computer Communications, 2014, vol. 40, no. 0, pp. 1 21.

[8] Kliazovich, D. and Bouvry, P. and Audzevich, Y. and Khan, S., *Greencloud: a packet-level simulator of energy-aware cloud computing data centers*, Global Telecommunications Conference (GLOBECOM 2010), 2010, IEEE, 2010, pp. 1-5.

[9] Al-Fares, Mohammad and Loukissas, Alexander and Vahdat, Amin, *A scalable, commodity data center network architecture*, Proceedings of the ACM SIGCOMM 2008 conference on Data communication, SIGCOMM '08, 2008.

[10] Greenberg, A. and Hamilton, J.R. and Jain, N. and Kandula, S. and Kim, C. and Lahiri, P. and Maltz, D.A. and Patel, P. and Sengupta, S., *Vl2: a scalable and flexible data center network*, Commun. ACM 54 (3) (2011) 95-104.

[11] Revolutionizing Network Design Flattening the Data Center Network with the QFabric Architecture. http://www.itbiz.com.ua/media/docs/Juniper/QFX/The%20Q%Fabric%20Architecture.pdf

[12] Guo, C. and Lu, G. and Li, D. and Wu, H. and Zhang, X. and Shi, Y. and Tian, C. and Zhang, Y. and Lu, S., *Bcube: a high performance, server-centric network architecture for modular data centers*, SIGCOMM Comput. Commun. Rev. 39 (4) (2009) 63-74.

[13] Guo, Chuanxiong and Wu, Haitao and Tan, Kun and Shi, Lei and Zhang, Yongguang and Lu, Songwu, *Dcell: a scalable and fault-tolerant network structure for data centers*, Proceedings of the ACM SIGCOMM 2008 conference on Data communication, SIGCOMM '08, 2008.

[14] Farrington, N. and Porter, G. and Radhakrishnan, S. and Bazzaz, H.H. and Subramanya, V. and Fainman, Y. and Papen, G. and Vahdat, A., *Helios: a hybrid electrical/optical switch architecture for modular data centers*, SIGCOMM Comput. Commun. Rev. 41 (4) (2010).

[15] Wang, G. and Andersen, D.G. and Kaminsky, M. and Papagiannaki, K. and Ng, T.E. and Kozuch, M. and Ryan, M., *C-through: part-time optics in data centers*, SIGCOMM Comput. Commun. Rev. 41 (4) (2010).

[16] Chen, K. and Singla, A. and Singh, A. and Ramachandran, K. and Xu, L. and Zhang, Y. and Wen, X. and Chen, Y., *Osa: an optical switching architecture for data center networks with unprecedented flexibility*, Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation, NSDI12, p. 18.

[17] Das, S. and Yiakoumis, Y. and Parulkar, G. and McKeown, N. and Singh, P. and Getachew, D. and Desai, P.D., *Application-aware aggregation and traffic engineering in a converged packet-circuit network*, Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference, 2011.

[18] Webb, Kevin C. and Snoeren, Alex C. and Yocum, Kenneth, *Topology switching for data center networks*, Proceedings of the 11th USENIX conference on Hot topics in management of internet, cloud, and enterprise networks and services, Hot-ICE'11, 2011.

[19] Chao, H. and Deng, K. L. and Jing, Z., *Petastar: a petabit photonic packet switch*, IEEE J. Sel. Areas Commun. 21 (7) (2003) 1096-1112.

[20] Xia, M.Y.K. and Kaob, Y.H. and Chao, H.J., *Petabit optical switch for data center networks*, Tech. Rep., Polytechnic Institute of NYU, 2010.

[21] Ye, X. and Yin, Y. and Yoo, S.J.B. and Mejia, P. and Proietti, R. and Akella, V., *Dos: a scalable optical switch for datacenters*, Proceedings of the 6th ACM/IEEE Symposium on Architectures for Networking and Communications Systems, ANCS10, ACM, New York, NY, USA, 2010, pp. 24:1-24:12.

[22] Singla, A. and Singh, A. and Ramachandran, K. and Xu, L. and Zhang, Y., *Proteus: a topology malleable data center network*, Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, Hotnets-IX, ACM, New York, USA, 2010, pp. 8:1-8:6.

[23] Bazzaz, H. et al., *Switching the optical divide: Fundamental challenges for hybrid electrical/optical data center networks*, ACM SOCC11, October 2011.

[24] http://www.cise.ufl.edu/~sahni/papers/tantofull.pdf