Postcard: Minimizing Costs on Inter-Datacenter Traffic with Store-and-Forward

Yuan Feng, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{*yfeng, bli*}*@eecg.toronto.edu*

Bo Li
Department of Computer Science
Hong Kong University of Science and Technology
bli@cs.ust.hk

***Abstract*—It is typical for cloud providers to operate a number of geographically distributed datacenters, where inter-datacenter traffic constitutes a large portion of a cloud provider's traffic demand over the Internet. Though such inter-datacenter traffic incurs substantial operational costs that are charged by ISPs, it varies significantly across different overlay links, raising the opportunity to optimize both routing and scheduling of inter-datacenter traffic to minimize costs. As data is being transmitted multiple sources to their respective destinations, an intermediate datacenter is also able to store data and forward them at a later time, so that peak traffic demand can be reduced. The cost minimization problem with such store-and-forward is challenging to solve in the general case, due to a large number of variables that determine how data can be sent from a source to a destination, and when should they be "paused" (stored) at an intermediate node, subject to a required maximum transfer time.**

In this paper, we present *Postcard*, an online optimization problem carefully formulated to minimize operational costs on inter-datacenter traffic with store-and-forward at intermediate nodes. To solve the optimization problem with an acceptable number of variables in *Postcard*, we have simplified the general problem by restricting data transmission to a time-slotted model, such that the problem can be modelled on a time-expanded graph. With extensive simulations, we compare results from solving *Postcard* to those from solving a flow-based problem without store-and-forward, and present the advantages and drawbacks of store-and-forward when it comes to minimizing costs on inter-datacenter traffic.

I. Introduction

Large-scale datacenters, offering the highest level of reliability, are deployed in large numbers around the world across different geographical regions, by not only cloud providers such as Amazon and Google, but also hosting and colocation providers such as Equinix. These cloud and colocation providers host enterprise-class infrastructures that offer a scalable computing environment to enterprise customers, at a much lower marginal cost due to the sharing nature of resources. With substantial upfront investments, it is important for cloud and colocation providers to minimize their operational costs while keeping customers satisfied.

Greenberg *et al.* [1] have revealed that traffic costs amount to around 15% of operational costs incurred to a cloud provider. In particular, Chen *et al.* [2] have pointed out that inter-datacenter traffic accounts for up to 45% of the total traffic going through datacenter egress routers. Since most cloud providers today rely on multiple Internet Service Providers (ISPs) to connect their geographically dispersed datacenters [3], operational costs can be effectively reduced if costs charged by these ISPs on inter-datacenter traffic can be minimized.

With diverging cost structures used by various ISPs and unique characteristics of inter-datacenter traffic, it is feasible to minimize operational costs incurred to datacenter operators, for a number of reasons. *First*, the cost of transmitting the same amount of traffic varies significantly across different inter-datacenter overlay links. For example, domestic traffic is substantially cheaper than traffic to global destinations [4], and traffic within the backbone network built by cloud providers themselves incurs much lower operational costs. *Second*, a large portion of inter-datacenter traffic is delay-tolerant, including backups, propagation of large updates, and migration of customer data. To reduce costs, we may re-route such traffic with intermediate datacenters as relay nodes, splitting traffic into smaller fractions, and transmitting them along multiple routing paths [5].

Last but not the least, intermediate datacenters are able to temporarily *store* the data to be relayed, and *forward* them at a later time to its downstream relay node or to the destination. Such an ability for intermediate nodes to *store-and-forward* does not reduce the amount of traffic to be relayed; yet the delayed forwarding may reduce peak traffic demand over an overlay link, and as such reduce operational costs over a longer charging period, if the ISP's charging scheme hinges upon such peak traffic demand.

At first glance, we only need to formally formulate the problem of minimizing operational costs, in the general case that data, from multiple sources to their destinations, can be temporarily stored at intermediate datacenters for a period of time, and be fractionally split to multiple paths. Unfortunately, as we show in this paper, the cost minimization problem in the general case, and in fact even a much simplified problem based on data flows without temporal storage on intermediate nodes, is challenging to be solved.

In this paper, we present *Postcard*, an online optimization problem carefully formulated to minimize operational costs on inter-datacenter traffic, with intermediate nodes being able to store incoming data and forward them at a later time to reduce peak traffic demand. *Postcard* is formulated as a tractable

This research was supported in part by a grant from NSFC/RGC under the contract N_HKUST610/11, by grants from HKUST under contracts under the contract RPC11EG29, SRFI11EG17-C and SBI09/10.EG01-C.

convex optimization problem only with linear constraints (*e.g.,* on link capacity and traffic conservation), to be solved with a standard solver (*e.g.,* with interior-point methods). In order to achieve this goal, *Postcard* is formulated with minimal simplification, much like a time-slotted model, where a data file starts its transmission at the beginning of a time interval, and finishes at the end of it. The key idea towards making the problem tractable to solve is to construct a time-expanded graph over multiple time intervals. With extensive simulations, we compare results from solving *Postcard* to those from solving a flow-based problem, and present the advantages and drawbacks of store-and-forward when it comes to minimizing costs on inter-datacenter traffic. To our knowledge, *Postcard* represents the first attempt to systematically study and formulate the problem of minimizing operational costs on inter-datacenter traffic, with the ability of intermediate nodes to store incoming data and forward them at a later time.

II. Rationale, Challenges, and a Flow-Based Approach

We first present the rationale and challenges of minimizing operational costs on inter-datacenter traffic, followed by a brief discussion on a flow-based simplification to address some of the challenges, yet still capturing the essence of the problem.

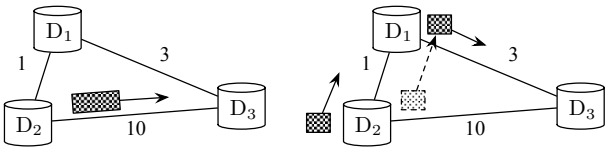
A. Rationale and Challenges

Cloud providers deploying a number of geographically distributed datacenters today, usually lease bandwidth from multiple ISPs for their inter-datacenter traffic. They are charged based on the predominant percentile-based charging scheme, *i.e.*, the q-th percentile charging scheme, in which operational costs are determined by the amount of traffic each cloud provider generates. To be more precise, an ISP records the traffic volume a cloud provider generates during each 5-minute interval. At the end of a complete charging period, all 5-minute traffic volumes are *sorted* in an ascending order, and the q-th percentile is used as the charging volume x to derive the cost by a piece-wise linear non-decreasing function $c(x)$ [6]. For example, if the 95-th percentile charging scheme is in use and the charging period is one year, then the charging volume x of the cost function corresponds to the traffic volume sent during the 99864-th sorted interval ($95\% \times 365 \times 24 \times 60/5 = 99864$).

Since cloud providers usually over-provision bandwidth resources to guarantee their peak-hour performance, percentile-based charging schemes may cause a substantial waste of capital investment, especially when a strong diurnal pattern is observed in inter-datacenter traffic [2]. When a certain amount of traffic is already generated between two datacenters during their peak hours, the same costs are incurred in subsequent time intervals even if the link between these two datacenters is idle or under-utilized (*e.g.,* during off-peak hours), as the cloud provider will have to pay for subsequent time intervals based on the already generated traffic volume nonetheless.

The geographically distributed locations of datacenters have provided a possible solution to reduce such costs on inter-datacenter traffic, as different ISPs charge traffic based on different prices per unit of traffic, and traffic demand differs at different datacenters at any given time (perhaps due to distinct time zones they reside). If traffic from one datacenter is allowed to be fractionally split into sub-flows, even to be stored temporarily by intermediate datacenters, cloud providers may benefit from "cheaper" routing alternatives, and time intervals along links that are already "paid" may be utilized more efficiently. In other words, if routing paths, flow assignments and scheduling strategies for each source-destination traffic pair are carefully designed, operational costs on inter-datacenter traffic can be efficiently reduced, or even minimized.

The following example illustrates the benefits by considering routing and scheduling strategies for inter-datacenter traffic. Shown in Fig. 1, datacenter D_2 needs to send a file of size 6 MB within 15 minutes to datacenter D_3, and the same cloud provider also operates another datacenter D_1, which is inter-connected with the other two datacenters. For simplicity, we assume that the 100-percentile charging scheme is in use, *i.e.*, the *maximum* traffic volume sent during time intervals will be the traffic volume to be charged; the cost function between each datacenter pair is the volume to be charged multiplied by a flat price (per MB) shown on the link; and the link capacity is sufficiently large, *e.g.*, 1 Gbps for conventional optical links, so that it is not a constraint in this example.



(a) Without routing and scheduling (b) With routing and scheduling

Fig. 1. How traffic costs can be reduced with routing and scheduling strategies: a motivating example.

As we can see, the flat price between each datacenter pair differs from one another, due to possible reasons of going through several paid peering ISPs, domestic regional pricing, and pricing discounts due to ISP backplane peering relationships [4]. Without routing or scheduling considerations, as shown in Fig. 1 (a), the file will be sent directly to D_3 within three 5-minute intervals, and the cost per time interval is $10 \times 2 = 20$. However, if routing and scheduling strategies are considered, the file can be split evenly into two blocks, and transmitted sequentially along the path $\{\mathrm{D}_2 \rightarrow \mathrm{D}_1 \rightarrow \mathrm{D}_3\}$ through three 5-minute intervals. As Fig. 1 (b) indicates, the cost per time interval can be reduced to $1 \times 3 + 3 \times 3 = 12$, which is much lower than that in the former case.

Unfortunately, the design of such routing and scheduling strategies in datacenters with multiple source-destination traffic pairs and different transfer time requirements involves formidable challenges. By allowing possible storage at intermediate datacenters, the complexity of scheduling strategies is increased significantly. Upon receiving data blocks, each intermediate datacenter has to make decisions on not only to relay them immediately or to store them for a while, but also how much time should each data block be stored and

at what rate should it be transmitted to the next "hop" along the path. Since both waiting times at intermediate datacenters and the transmission rate on every link along the routing path will affect the transfer completion time of each traffic pair, those decisions are hard to make, especially if an optimization objective is involved. The problem is further complicated by the lack of synchronization among all inter-datacenter traffic pairs: traffic can be generated at any time in a charging period; each data block can be stored at intermediate datacenters for an arbitrary period of time; and its transmission can be completed at any time, as long as the total transfer completion time does not exceed its maximum tolerable transfer time.

B. A Flow-Based Approach

Due to these difficulties involved when designing optimal routing and scheduling strategies with the objective of minimizing traffic costs, simplifications are necessary to make the problem tractable. One possible way is to completely *eliminate* the possibility of temporal storage at intermediate datacenters, so that each source-destination traffic pair can be represented by a *flow* with its desired transmission rate. The resulted routing and flow assignment problem for all inter-datacenter flows can be formulated by a *flow-based model*, which is practically solvable with efficient combinatorial algorithms, yet the essence of which is still captured.

In this approach, the desired transmission rate of each traffic pair can be obtained by the quotient of the traffic size divided by its maximum tolerable transfer time, which equals the minimum tolerable transmission rate in the interest of minimizing traffic costs. For example, the desired transmission rate of the file in Fig. 1 is 6 MB / 15 min = 54.6 kbps. If each inter-datacenter flow can still be transmitted to its destination datacenter via multiple paths, each with multiple hops, the original problem becomes a simpler problem of how routing paths are determined and how flows are assigned to these paths, such that the incurred traffic costs are minimized.

It turns out that, even the solution to such a simpler problem, based on the flow-based model, is still quite elusive. Although the problem shares some similarity with the combinatorial minimum-cost multicommodity flow problem [7], the non-linear cost function caused by the percentile-based charging scheme makes it impossible to directly apply existing algorithms, such as [8], to solve this kind of problems. We believe that this problem can be decoupled into two sub-problems. The first problem solves the routing and flow assignment problem for a subset of traffic pairs, with the objective of fully utilizing the already paid traffic volume in the inter-datacenter network; and the second one finds the optimal routing paths and flow assignments for the rest of traffic pairs, aiming at minimizing additionally incurred traffic costs. Since these two sub-problems fall in the form of the *maximum concurrent flow problem* and the *minimum-cost multicommodity flow problem*, respectively, the original cost minimization problem with the flow-based model can be solved by solving these two sub-problems sequentially.

III. System Model

In *Postcard*, we are interested in making optimal decisions to minimize the cost on inter-datacenter traffic to a cloud provider in an *online* fashion. Since inter-datacenter traffic can not be accurately predicated beyond a few seconds [9], we seek to answer the following question: at a certain time t, when multiple source-destination traffic pairs are to be transferred among datacenters, what is the optimal routing and scheduling strategy we can apply to minimize the cost incurred by this traffic, under the assumption that all routing paths and flow assignments for previous traffic pairs are already known? Before formulating this problem formally, we first introduce our system model in this paper.

In our model, *files* are used to represent all inter-datacenter traffic. The term is used in a very generic fashion, in that it means a block of data to be transmitted across the boundary of datacenters with its own size and maximum tolerable transfer time, and does not necessarily have to be related to file systems. For example, a *file* in our context can be a set of intermediate results in MapReduce tasks. Let $\mathcal{K}(t)$ be the set of files to be transmitted at time t. For each file $k \in \mathcal{K}(t)$, we use a four-tuple specification (s_k, d_k, F_k, T_k) to describe it. Here, vertices s_k and d_k indicate the source and the destination datacenter from and to which file k is being transmitted, F_k is the size of file k, and T_k describes the maximum tolerable transfer time of file k. The general case that a file can have multiple destinations can be handled by introducing a separate file to each source-destination pair, with the same source datacenter, file size, and maximum tolerable transfer time.

We consider an inter-datacenter network that consists of multiple geographically distributed datacenters, operated by a single cloud provider. Such an inter-datacenter network can be denoted by a complete directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ indicates the set of datacenters, and $\mathcal{E}$ indicates the set of overlay links inter-connecting datacenters. Each datacenter is connected to all other datacenters through several ISPs. For each link $\{i, j\} \in \mathcal{E}$, we use a positive function $c_{ij}(t)$ to denote the *available link capacity* at time t, which is the maximum residual link capacity left after some of the capacity is used for the transmission of files before t. Since each file's ongoing transfer will remain in the network for some time, the transmission of a previously generated file will affect the available link capacity at the current time. We also use a nonnegative real-value function a_{ij} to denote the cost per traffic unit transferred from datacenter i to j.

Since traffic volume is computed periodically in every 5-minute time interval in percentile-based charging schemes, we slot the time dimension into multiple time intervals with the same duration, denoted by $\bar{t}$. If the number of time intervals in a charging period is I, the time dimension in this charging period can be represented by $\{t|0 \leq t \leq I\}$. To make the problem tractable, we assume that files in $\mathcal{K}(t)$ is sufficiently small, such that each file is guaranteed to be completely received over an overlay link by the downstream datacenter within one time interval. Such an assumption is

valid in general, since most data to be transferred across datacenters are within a few hundred Gigabytes, and overlay links that interconnect datacenters are usually designed with high capacities, such as OC-192 with thousands of miles of connected fiber in SevenL Networks [10], one of the datacenter infrastructures in North America. OC-192 is a network link with transmission rates of up to 1.2 GB/second, allowing the complete transfer of 360 GB of data within one time interval of 5 minutes. For even larger files, they can be divided into smaller pieces, each of which can be considered as a new file with the same four-tuple specification.

We further assume that the inter-datacenter transmission of a file starts at the beginning of a time interval, and finishes at the end of that interval. It is obviously feasible for a file to be completely transferred to the downstream datacenter with a higher rate, so that it is received in less time than the duration of a time interval. Yet, for the sake of minimizing traffic costs incurred on inter-datacenter links, and with the assumption that shorter transfer times do not lead to any higher utility to the cloud provider due to the delay-tolerant nature of inter-datacenter traffic, it is desirable to finish the transfer with exactly one time interval. More formally, if a fraction of file k with size $M_{ij}^k(t)$ to be transferred from datacenter i to datacenter j at time interval t, the corresponding flow rate of file k on link $\{i,j\}$ at time interval t equals $M_{ij}^k(t)/\bar{t}$. During its transmission, a file can be stored temporarily in intermediate datacenters before being relayed to the destination datacenter, and can be transferred to the destination datacenter over multiple paths, as well as over multiple hops within a path.

IV. Formulating the Problem: the First Try

Based on our time-slotted model, the problem of minimizing costs on inter-datacenter traffic in *Postcard* can be formally stated as: with all routing paths and flow assignments of files transferred before t known, for every file to be transferred at time interval t, we seek to make optimal decisions with respect to a few dimensions of the design space. At the source datacenter, we will decide the proper number of fractions a file should be divided into, and compute forwarding paths via intermediate datacenters for each fraction to follow. At each intermediate datacenter, upon receiving a fraction of a file, we need to decide whether it needs to be *forwarded* immediately to a downstream datacenter along its path, or *held* temporarily for later forwarding, following the philosophy of store-and-forward.

Without loss of generality, we assume the cost function to be linear, *i.e.*, $c(x) = a \cdot x$, where x is the traffic volume to be charged. We also assume that the 100-th percentile charging scheme is being used for simplicity. To be specific, if we use $M_{ij}^k(n)$ to denote the size of a fraction of file k to be transferred along link $\{i,j\}$ at time interval n, the inter-datacenter traffic volume to be charged on link $\{i,j\}$ after transmitting files generated *up to* time t equals

$$X_{ij}(t) = \max\{X_{ij}(t-1), \max_{\max_k T_k} \sum_{k\in\mathcal{K}(t)} M_{ij}^k(n)\},$$

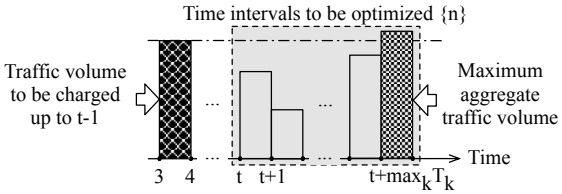


Fig. 2. An illustration of the traffic volume to be charged up to time t.

where $\max_{\max_k T_k} \sum_{k\in\mathcal{K}(t)} M_{ij}^k(n)$ is the maximum aggregate traffic volume on link $\{i,j\}$ of time intervals $\{n | t \le n \le t + \max_k T_k\}$. Illustrated in Fig. 2, the shadowed area represents time intervals to be optimized. For simplicity, we assume all files can finish their transmission within a charging period, *i.e.*, $t + \max_k T_k \le I, \forall k$. As a special case, $M_{ii}(n)$ represents the volume of data that is stored temporarily from time interval n to $n+1$ — but not forwarded — at datacenter i, referred to as the *holdover*. More formally, the problem of minimizing costs on inter-datacenter traffic in *Postcard* can be formulated as the following optimization problem:

$$\min_{M_{ij}^k(n)} \quad \sum_{\{i,j\}\in\mathcal{E}} a_{ij} X_{ij}(t) I \tag{1}$$

$$\text{s.t.} \quad \sum_{k\in\mathcal{K}(t)} M_{ij}^k(n) \le c_{ij}(n)\bar{t},\ \forall\{i,j\}\in\mathcal{E} \tag{2}$$

$$\sum_{j\in\mathcal{V}} \sum_{n=t}^{t+\max_k T_k} \left(M_{s_k j}^k(n) - M_{j s_k}^k(n-1)\right) = F_k,$$

$$\sum_{j\in\mathcal{V}} \sum_{n=t}^{t+\max_k T_k} \left(M_{d_k j}^k(n) - M_{j d_k}^k(n-1)\right) = -F_k,$$

$$\sum_{j\in\mathcal{V}} M_{ij}^k(n) - \sum_{j\in\mathcal{V}} M_{ji}^k(n-1) = 0,$$

$$\forall k \in \mathcal{K}(t),\ \forall i \in \mathcal{V}/\{s_k, d_k\} \tag{3}$$

$$M_{ij}^k(n) \ge 0,\ \forall k \in \mathcal{K}(t),\ \forall\{i,j\}\in\mathcal{E} \tag{4}$$

$$T_k' \le T_k,\ \forall k \in \mathcal{K}(t). \tag{5}$$

In this formulation, the objective is to find optimal traffic allocation and scheduling strategies for to-be-transferred files generated at time interval t, such that the costs on inter-datacenter traffic after transmitting all files generated up to t are minimized. The optimization variables are $\{M_{ij}^k(n) | \forall\{i,j\} \in \mathcal{E}, \forall k \in \mathcal{K}(t), t \le n \le t + \max_k T_k\}$, each of which indicates the size of a fraction of file k to be transferred along link $\{i,j\}$ from time interval n to $n+1$.

At time t, we can observe that $M_{ij}^k(t)$ represents the size of fractions that files generated at datacenter i should be divided into. At time n ($t \le n \le t + \max_k T_k$), $M_{ii}^k(n)$ reflects the scheduling strategy of whether or not a fraction of a file k should be temporarily held at an intermediate datacenter i. As a result, $M_{ij}^k(n)$ over all datacenters at time intervals $[t, t+\max_k T_k]$ naturally describes both routing and scheduling strategies for file $k \in \mathcal{K}(t)$. Due to the assumption that flows assigned on every link for files transferred in past time intervals are known *a priori*, the traffic volume to be charged

on each link after transmitting files generated up to time $t-1$ is known, *i.e.*, $X_{ij}(t-1)$ is known at time t.

Inequality (2) represents the link capacity constraint, which ensures that the total data volume transmitted on a link during a time interval does not exceed the link capacity. Equation (3) represents the traffic conservation constraint, which ensures the traffic volume going into an intermediate datacenter at each time interval equals to that going out of it at the next time interval, and the traffic volume coming from the source datacenter and going to the destination datacenter over the entire charging period is exactly the same as the file size. Inequality (4) ensures that fractions of any file are of non-negative sizes. Inequality (5) restricts that the actual transfer time of each file T'_k — including both the transmission time and the waiting time at intermediate datacenters — has to be within its maximum tolerable transfer time.

However, representing T'_k analytically appears to be very difficult, if not impossible. Since the actual transfer time of each file is determined by both its waiting time at intermediate datacenters and lengths of its routing paths, T'_k is a function of the optimization variable $M^k_{ij}(n)$. To be exact, the actual transfer time of file k equals the maximum number of hops among all paths in use for the transmission of all file k's fractions from its source datacenter to its destination one, plus the number of time intervals each fraction is stored in intermediate datacenters, *i.e.*, time intervals with $\{M^k_{ii}(n) > 0 | \sum_j M^k_{ji}(n') > 0, \forall n' < n\}$ over all datacenters, times the duration of one time interval. The result is a non-linear transfer time constraint (5), which increases the complexity of the problem substantially.

Since the optimization problem (1) resembles the traditional dynamic flow problem that is first proposed by Ford and Fulkerson in 1958 [11], their solutions shed some light on how our problem may be tackled. The dynamic flow problem is to answer the following question: what is the maximal amount of traffic that can be transferred from a source to a destination in any given number of time intervals, in a network where each link is associated with a flow capacity and a traversal time?

The model used in the dynamic flow problem is much simpler than ours. We have traffic-dependent transfer times, in that the transfer time of each file is determined by the number of hops along the path rather than traversal times on each link. We also have different transfer time constraints for each file, whereas the dynamic flow problem assumes the same constraint for all flows. Despite these discrepancies, a variant of the dynamic flow problem — taking link costs into consideration — has a similar formulation as our optimization problem (1).

Ford and Fulkerson proposed to solve the dynamic flow problem by representing the time dimension through time expansion. By introducing a "virtual" copy of all nodes at each time interval, they transformed a dynamic problem over time into an equivalent static problem in a time-expanded graph [11]. Inspired by their gadget of time expansion, we seek to construct a well defined time-expanded graph for the optimization problem (1) as well, with the hope that the transfer time constraint for each file (5) can be represented analytically in the constructed time-expanded graph. If successful, our dynamic problem of minimizing costs on inter-datacenter traffic may also be solved by a corresponding static problem in the time-expanded graph. We now begin to explore such a possibility.

V. Postcard: Problem Formulation on a Time-Expanded Graph

The key idea in our approach to solve the traffic cost minimization problem in a store-and-forward inter-datacenter network is to construct a time-expanded graph for all datacenters in the network over the considered time period. We use $\mathcal{G}(t) = (\mathcal{V}(t), \mathcal{E}(t))$ to represent the time-expanded graph for inter-datacenter network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ over time intervals $[t, t + \max_k T_k]$. $\mathcal{G}(t)$ contains one virtual copy of each node i, $\forall i \in \mathcal{V}$ at the beginning of each time interval, which builds a time layer; it also contains a copy for all links $\{i, j\} \in \mathcal{E}$ between each pair of time layers, with the same bandwidth capacity and cost per traffic unit.

To be rigorous, let $\{i^n | \forall i \in \mathcal{V}, t \le n \le t + \max_k T_k\}$ be the nodes in $\mathcal{G}(t)$, and define directed links $\{i^n j^{n+1} | \forall \{i, j\} \in \mathcal{E}, t \le n \le t + \max_k T_k - 1\}$ to be links in $\mathcal{G}(t)$. For each link $\{i^n j^{n+1}\}$ with $i \neq j$, it is associated with the available bandwidth capacity c_{ijn} and the cost per traffic unit a_{ij}, corresponding to link $\{i, j\}$ in the original inter-datacenter network $\mathcal{G}$; for each link $\{i^n i^{n+1}\}$, it is associated with bandwidth capacity ∞ and zero cost per traffic unit, reflecting the ability to store data at intermediate datacenters. Since the source and destination datacenters of file $k \in \mathcal{K}(t)$ in the time-expanded graph will be s^t_k and $d^{t+T_k}_k$, file k can be described by a three-tuple specification $(s^t_k, d^{t+T_k}_k, F_k)$ in the new graph.

An illustration of constructing the time-expanded graph for an inter-datacenter network is shown in Fig. 3. The example network on the left hand side has 4 datacenters and 2 files to be transferred. a on each link denotes the cost per traffic unit, the link capacity c is 5 for all links, and the duration of each time interval is 1. At the beginning of time interval $t = 3$, File 1 is to be transferred from datacenter 2 to datacenter 4, with a size of 8 and a maximum tolerable transfer time of 4 time intervals; File 2 needs to be transferred from datacenter 1 to datacenter 4, with a size of 10 and a maximum tolerable transfer time of 2 time intervals. The right hand side shows the time-expanded graph for the example network from $t = 3$ to $t + \max_k T_k = 7$. Nodes with colours and patterns indicate the source and destination datacenters, implicitly showing the corresponding tolerable transfer times of two files.

In the constructed time-expanded graph, if we use M^k_{ijn} to represent the size of a fraction of file k to be transferred from node i^n to j^{n+1}, it corresponds to the optimal variable $M^k_{ij}(n)$ in the original dynamic optimization problem (1). The transfer time constraint can be reflected by the restriction that for each file k, $M^k_{ijn} = 0$ for all nodes after the time layer $n = t + T_k$, after file k completes its transmission. That is to say, we only consider the traffic allocation and scheduling strategy for each

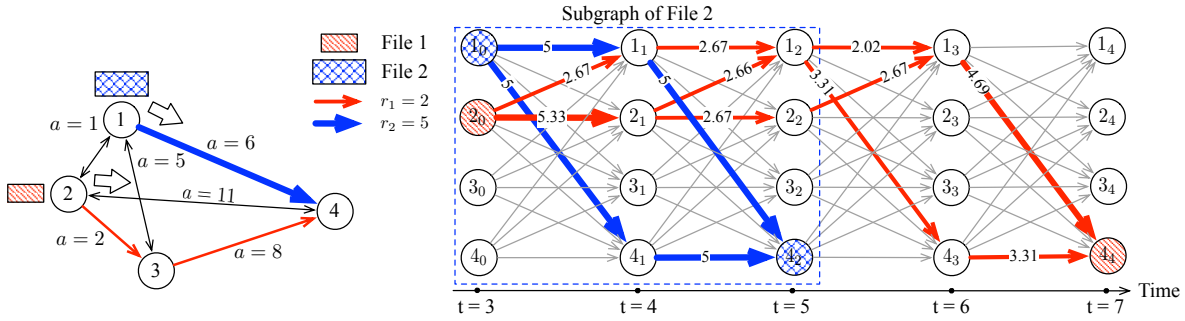


Fig. 3. An example inter-datacenter network and its corresponding time-expanded graph.

file in a subgraph of the time-expanded graph. The dashed square in Fig. 3 shows the subgraph corresponding to File 2. As a result, the *dynamic* traffic allocation and scheduling problem in an inter-datacenter network can be solved by a *static* traffic allocation problem in the constructed time-expanded graph, which has the form of:

$$\min_{M^k_{ijn}} \quad \sum_{\{i^n j^{n+1}\} \in \mathcal{E}(t)} a_{i^n j^{n+1}} X_{ijt} I \tag{6}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{K}(t)} M^k_{ijn} \le c_{ijn} \bar{t}, \ \forall \{i^n j^{n+1}\} \in \mathcal{E}(t) \tag{7}$$

$$\sum_{j^n \in \mathcal{V}(t)} (M^k_{s_k jt} - M^k_{j s_k (t-1)}) = F_k,$$

$$\sum_{j^n \in \mathcal{V}(t)} (M^k_{d_k j(t+T_k)} - M^k_{j d_k (t+T_k-1)}) = -F_k,$$

$$\sum_{j^n \in \mathcal{V}(t)} M^k_{ijn} - \sum_{j^n \in \mathcal{V}(t)} M^k_{ji(n-1)} = 0,$$

$$\forall k \in \mathcal{K}(t), \forall i^n \in \mathcal{V}(t) \setminus \{s^t_k, d^{t+T_k}_k\} \tag{8}$$

$$M^k_{ijn} \ge 0, \forall k \in \mathcal{K}(t), \forall \{i^n j^{n+1}\} \in \mathcal{E}(t) \tag{9}$$

$$M^k_{ijn} = 0, \forall k \in \mathcal{K}(t), \forall \{i^n j^{n+1} | n > t + T_k\} \tag{10}$$

where $X_{ijt} = \max\{X_{ij(t-1)}, \max_{\max_k T_k} \sum_{k \in \mathcal{K}(t)} M^k_{ijn}\}$ represents the traffic volume to be charged on link $\{i, j\}$ after transmitting files generated up to time t.

The formulated optimization problem is equivalent to the original problem (1), with its presentation in the time-expanded graph. Inequality (7), which shows the link capacity constraint, equations (8), which state the traffic conservation constraints, and inequality (9), which represents the non-negative traffic allocation constraint, correspond to constraints (2) (3) and (4) in the original optimization problem. The transfer time constraint for each file is now represented by equation (10), which guarantees that the transmissions of all files are done within their corresponding maximum tolerable transfer times. It is not difficult to find out that problem (6) is a convex optimization problem, since both the operations of non-negative weighted sums and pairwise maximum preserve convexity [12]. With linear constraints only, classic algorithms such as subgradient projection methods and interior-point methods can be applied to solve this problem.

For the example shown in Fig. 3, optimal results obtained by solving the formulated static traffic allocation problem are shown in the time-expanded graph. As we can see, Instead of sending File 1 immediately when it is generated, part of it can be stored at the source datacenter 2 and the intermediate datacenter 1 for later transmission, taking advantage of the already paid link $\{1, 4\}$ when transmitting File 2 during the first two time intervals. By doing so, the cost per time interval can be reduced to 32.67, compared to 52 if no routing and scheduling strategy is considered. If we solve the same problem by using the flow-based approach, the desired transmission rates of both files are $r_1 = 2$, and $r_2 = 5$. The optimal flow assignments for both files are shown in the left hand side of Fig. 3. Since File 2 will be transferred via the cheapest path $\{\mathrm{D}_1 \to \mathrm{D}_4\}$, File 1 will not be able to take the cheaper path $\{\mathrm{D}_2 \to \mathrm{D}_1 \to \mathrm{D}_4\}$ due to the link capacity constraint. As a result, it will have to take the cheapest *available* path $\{\mathrm{D}_2 \to \mathrm{D}_3 \to \mathrm{D}_4\}$, which results in a cost per interval of 50.

As illustrated in the example, allowing temporal storage at intermediate datacenters may increase the possibility of fully utilizing links that have already been paid for. With percentile-based charging schemes, if some traffic has already been transmitted along a link during a time interval, the same cost will be incurred even if the link is idled or used for transmitting less traffic in later time intervals. With the store-and-forward approach, traffic is "time-shifted" to later time intervals as much as possible, with the objective of minimizing costs by using links that have already been paid for, perhaps due to the transmission of other files.

VI. The Time Expansion Approach

In the *Postcard* problem formulation, the construction of a time-expanded graph is used to solve the traffic cost minimization problem in a store-and-forward inter-datacenter network. The time expansion approach, however, can be applied to ease the formulation of a class of similar problems.

For example, an interesting problem is to utilize leftover bandwidth during non-peak hours for the benefit of bulk "background" traffic, such as backups and data migration, first proposed by Laoutairs *et al.* [5]. Since the cloud provider only use the leftover bandwidth, which is assumed to be covered by the already paid traffic volume generated during peak hours, there is no longer a concern on traffic costs. The problem is to find the optimal traffic allocation and scheduling strategy, such that as many bulk files as possible can be transferred within a maximum transfer time constraint for each of them.

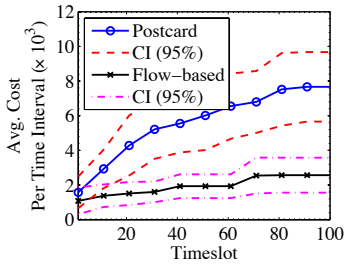


Fig. 4. Average costs per $\bar{t}$ with $c_{ij} = 100$ GB/$\bar{t}$ and $\max_k T_k = 3$.

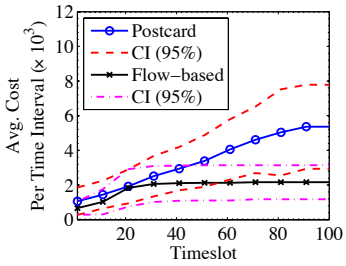


Fig. 5. Average costs per $\bar{t}$ with $c_{ij} = 100$ GB/$\bar{t}$ and $\max_k T_k = 8$.

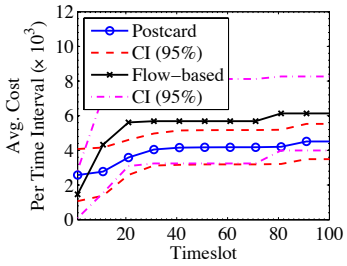


Fig. 6. Average costs per $\bar{t}$ with $c_{ij} = 30$ GB/$\bar{t}$ and $\max_k T_k = 3$.

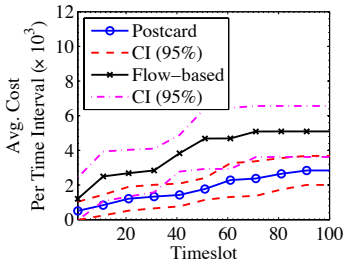


Fig. 7. Average costs per $\bar{t}$ with $c_{ij} = 30$ GB/$\bar{t}$ and $\max_k T_k = 8$.

Using the time expansion approach, with the same optimization variables $\{M_{ijn}^k | i^n j^{n+1} \in \mathcal{E}(t), k \in \mathcal{K}(t), t \leq n \leq t + \max_k T_k\}$ indicating the size of a fraction of file k to be transferred from node i^n to j^{n+1}, the corresponding optimization problem can be formulated by replacing the objective function in problem (6) with the following one:

$$\max_{M_{ijn}^k} \sum_{n=1}^{t+\max_k T_k} \sum_{\{i^n j^{n+1}\} \in \mathcal{E}(t)} \sum_{k \in \mathcal{K}(t)} M_{ijn}^k, \qquad (11)$$

which indicates the maximum of the transmitted traffic volume up to time interval t. With all constraints remaining the same, problem (11) is a linear optimization problem that can be efficiently solved by linear programming solvers. Laoutairs *et al.* [5] has also proposed a very similar time expansion approach to solve the problem in a store-and-forward inter-datacenter network. However, they only consider the case of making an optimal decision for a single file, while we consider the case of making optimal decisions for transmitting *multiple* files with different transfer time constraints, which is more generic and much more challenging.

Another possible problem faced by a cloud provider might be: given a certain budget on costs incurred by inter-datacenter traffic, what is the *maximum* number of files that a cloud provider can transfer among geographically distributed datacenters up to a certain time interval t? This problem makes more sense during the peak traffic hours, when more files are waiting to be transferred, compared to the limited capacity restricted by a cloud provider's budget. The cloud provider would wish to satisfy as many inter-datacenter file transfer requests as possible to provide a more competitive service, as long as the incurred traffic costs are within its budget.

Such problem can also be easily formulated by using our time expansion approach. With the same objective function (11), this problem takes the form by adding an additional constraint on traffic costs, $\sum_{\{i^n j^{n+1}\} \in \mathcal{E}(t)} a_{i^n j^{n+1}} X_{ijt} I \leq B$, where B is the budget on traffic costs. The resulted problem is also a convex optimization problem with a linear objective function and a convex inequality constraint [12].

VII. *Postcard:* Performance Evaluation

We dedicate this section to investigate how *Postcard* performs in reducing costs on inter-dataceter traffic. Specifically, we seek to investigate its advantages and drawbacks compared to the flow-based approach.

The evaluation of *Postcard* is based on our implementation in a time-slotted simulator, using the `fmincon` function provided by MATLAB. We simulate in a system with 20 datacenters, forming a complete graph. The cost per traffic unit on each link is set to be uniformly random within $[1, 10]$. In each time interval, the number of files to be transferred is uniformly random between $[1, 20]$, each of which has a uniformly random size of $[10, 100]$ GB. The source and destination datacenters of each file are chosen from the datacenter set uniformly. We conduct our simulations 10 times, each lasts for 100 time slots. For comparisions, we also implement the flow-based approach introduced in Sec. II-B with the same evaluation setup. Still, we assume that the 100-th percentile charging scheme is in use in our simulation.

We consider four different simulation settings. The first two settings are with sufficient link capacities, *i.e.* $c_{ij} = 100$ GB/$\bar{t}$ for all $\{i, j\} \in \mathcal{E}$, one of which has more urgent files ($\max_k T_k = 3$), whereas the other has more delay tolerant files ($\max_k T_k = 8$). The last two settings are with limited link capacities, *i.e.* $c_{ij} = 30$ GB/$\bar{t}$ for all $\{i, j\} \in \mathcal{E}$, and again, one has more urgent files and the other has more delay tolerant files. Average costs per time interval on inter-datacenter traffic and their 95% confidence intervals with both *Postcard* and the flow-based approach in each setting are shown in Fig. 4, Fig. 5, Fig. 6, and Fig. 7, respectively. The results reveal that the flow-based approach outperforms *Postcard* significantly when there are sufficient link capacities, while *Postcard* demonstrates superior performance when link capacities are throttled. We also discover that *Postcard* leads to lower costs when there are more delay tolerant files in the system, with either sufficient or limited link capacities.

The reason is that, store-and-forward incurs bursty traffic on relay paths compared to the flow-based approach. Take the network in Fig. 3 as an example. If we wish to transfer a file with a size of 10 from datacenter D_2 to D_4 in two time intervals via the path $\{\mathrm{D}_2 \rightarrow \mathrm{D}_1 \rightarrow \mathrm{D}_4\}$, which is the cheapest path between these two datacenters, the maximum traffic volume per time interval with the flow-based approach is 5, while that with *Postcard* will be 10. Instead of forwarding the file immediately after receiving the first byte in the flow-based approach, datacenter D_1 has to wait until it has fully received the entire file in *Postcard*. This results in bursty traffic on both links, and hence, higher costs with percentile-based charging schemes.

However, when link capacities are limited, cheaper links may be occupied by urgent traffic for some time intervals. As a result, they are unavailable with the flow-based approach, even if they will idle for most subsequent time intervals, *e.g.,* link $\{1, 4\}$ in the example shown in Fig. 3. On the contrary, store-and-forward provides possibilities to fully utilize those cheaper links by taking advantage of files with longer tolerable transfer times. Shown in our simulations, *Postcard* exhibits better performance when link capacities are limited. Since the more delay tolerant files in the network, the more opportunities exist for the "time-shifting" of inter-datacenter traffic, costs are reduced with *Postcard* when there are more delay tolerant files in the system, with either sufficient or limited link capacities.

VIII. Related Work

Geographically distributed datacenters have been an active research topic recently. Regarding the traffic characteristics across multiple datacenters, [13] and [14] discussed good designs of caching and load balancing strategies through measuring traffic dynamics in the Google cloud. Chen *et al.* presented their measurement results on inter-datacenter traffic characteristics using five Yahoo! datacenters, which was the first measurement study on inter-datacenter traffic [2].

Most analytical papers on inter-datacenter networks focus on dynamic load distribution across datacenters. Rao *et al.* [15] sought to minimize the total electricity or energy costs in a cloud with multiple geographically dispersed datacenters using geographical load balancing. DONAR was proposed to optimally direct cloud users' requests, based on datacenters' current loads and users' performance penalty [16]. Mahimkar *et al.* proposed to provide bandwidth on demand among distributed datacenters, with the objective of facilitating inter-datacenter communication [17].

Regarding cloud providers' operational costs on traffic, Zhang *et al.* designed a routing algorithm using the flow-based model to optimize the costs on datacenter-to-client traffic in each time interval [3]. However, they simplified the problem substantially by eliminating the consideration of the time dimension. Laoutaris *et al.* [5] proposed NetStitcher, which is the most relevant to our work in this paper. By taking advantage of the leftover bandwidth at night, NetStitcher transfers bulk data in a single file among datacenters through store-and-forward to reduce costs on inter-datacenter traffic. As we have discussed in Sec. VI, our model considers minimizing costs with the co-existence of multiple source-destination traffic pairs, which is more general and more challenging than the case of transferring a single file.

IX. Concluding Remarks

In this paper, we have presented *Postcard*, an online optimization problem meticulously formulated to minimize operational costs on inter-datacenter traffic, with the ability for intermediate nodes to *store* data and *forward* them at a later time. An important observation that serves as the foundation of this paper is that, with a percentile-based charging scheme, part of the inter-datacenter traffic can be transferred free of charge, if a link has been previously used for data transmission in the same charging period. The highlight of this paper is our proposed way to simplify the cost optimization problem with store-and-forward in the general setting: by restricting data transmission to a time-slotted model, it becomes feasible to formulate the problem by modelling the inter-datacenter network with a time-expanded graph. By solving the optimization problem with convex optimization solvers, *Postcard* allows us to compare the cost of allowing store-and-forward on intermediate datacenters with that of a flow-based model in our simulations. We have observed that *Postcard* exhibits better performance when link capacities are limited, or when the data to be transferred among datacenters are more delay tolerant.

References

[1] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The Cost of a Cloud: Research Problems in Data Center Networks," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 68–73, 2009.

[2] Y. Chen, S. Jain, V. Adhikari, Z.-L. Zhang, and K. Xu, "A First Look at Inter-Data Center Traffic Characteristics via Yahoo! Datasets," in *Proc. IEEE INFOCOM*, 2011, pp. 1620 –1628.

[3] Z. Zhang, M. Zhang, A. Greenberg, Y. C. Hu, R. Mahajan, and B. Christian, "Optimizing Cost and Performance in Online Service Provider Networks," in *Proc. 7th Conference on Networked Systems Design and Implementation (NSDI)*, 2010, pp. 1–15.

[4] V. Valancius, C. Lumezanu, N. Feamster, R. Johari, and V. V. Vazirani, "How Many Tiers? Pricing in the Internet Transit Market," in *Proc. ACM SIGCOMM*, 2011, pp. 194–205.

[5] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-Datecenter Bulk Transfers with NetStitcher," in *Proc. ACM SIGCOMM*, 2011.

[6] D. K. Goldberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang, "Optimizing Cost and Performance for Multihoming," in *Proc. ACM SIGCOMM*, 2004, pp. 79–92.

[7] D. Karger and S. Plotkin, "Adding Multiple Cost Constraints to Combinatorial Optimization Problems, with Applications to Multicommodity Flows," in *Proc. 27th ACM Symposium on Theory of Computing (SOTC)*, 1995, pp. 18–25.

[8] A. Goldberg and R. Tarjan, "Solving Minimum-Cost Flow Problems by Successive Approximation," in *Proc. 9th ACM Symposium on Theory of Computing (SOTC)*, 1987, pp. 7–18.

[9] T. Benson, A. Anand, A. Akella, and M. Zhang, "The Case for Fine-Grained Traffic Engineering in Data Centers," in *Proc. ACM SIGCOMM Workshop: Research on Enterprise Networking (WREN)*, 2010.

[10] "SevenL Networks Inc.: Network and Data Center Infrastructure." [Online]. Available: https://www.sevenl.net/infrastructure

[11] J. L. R. Ford and D. R. Fulkerson, "Constructing Maximal Dynamic Flows from Static Flows," *Operations Research*, vol. 6, no. 3, pp. 419–433, 1958.

[12] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[13] R. Krishnan, H. V. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao, "Moving Beyond End-to-End Path Information to Optimize CDN Performance," in *Proc. 9th Internet Measurement Conference (IMC)*, 2009, pp. 190–201.

[14] V. K. Adhikari, S. Jain, and Z.-L. Zhang, "YouTube Traffic Dynamics and Its Interplay with a Tier-1 ISP: An ISP Perspective," in *Proc. 10th Internet Measurement Conference (IMC)*, 2010, pp. 431–443.

[15] L. Rao, X. Liu, L. Xie, and W. Liu, "Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[16] P. Wendell, J. W. Jiang, M. J. Freedman, and J. Rexford, "DONAR: Decentralized Server Selection for Cloud Services," in *Proc. ACM SIGCOMM*, 2010, pp. 231–242.

[17] A. Mahimkar, A. Chiu, R. Doverspike, M. D. Feuer, P. Magill, E. Mavrogiorgis, J. Pastor, S. L. Woodward, and J. Yates, "Bandwidth on Demand for Inter-Data Center Communication," in *Proc. 10th ACM Workshop on Hot Topics in Networks (HotNets)*, 2011, pp. 241–246.