ADMM-based Distributed Algorithm for Emergency Backup in Time-Variant Inter-DC Networks

Xiaokang Xie, Qing Ling, Ping Lu, Zuqing Zhu†
School of Information Science and Technology, University of Science and Technology of China, Hefei, China
†Email: {zqzhu}@ieee.org

***Abstract*—This paper considers the emergency backup in an inter-datacenter (inter-DC) network whose topology is time-variant due to the progress of a disaster. We first transform the dynamic backup into a static flow problem through building a variable time-expanded network (V-TEN). Then, by considering both data utility and resource cost, we formulate an optimization to maximize the backup profit and leverage the alternating direction method of multipliers (ADMM) to design a time-efficient and distributed algorithm. Simulation results show that our ADMM-based algorithm outperforms several existing ones.**

***Index Terms*—Emergency backup, Inter-datacenter network, Time-expanded network (TEN), Time-variant network, Alternating direction method of multipliers (ADMM).**

I. Introduction

It is known that to adapt to the exponential growth of data-intensive applications, enterprises such as Google, Microsoft and Amazon have built inter-datacenter (inter-DC) networks to connect geographically distributed DCs [1]. Since a DC usually stores huge amounts of data and provides services to numerous users, its breakdown can cause massive data losses and service outages to thousands or even millions of people. However, a natural disaster can easily wipe multiple DCs out. For instance, the Sichuan earthquake in 2008 had destroyed tens of DCs in China and caused severe data losses [2], while Hurricane Sandy had brought down several DCs in the east coast of the United States for days. Hence, inter-DC networks are especially vulnerable to natural disasters and thus they have to use data backups. Such data backups are mainly in two types, *i.e.*, the regular backup that is performed during normal operation for improving data redundancy [3], and the emergency backup that is conducted in response to upcoming disasters for evacuating important data out [4].

For regular backup, previous studies have considered how to optimize the resource allocation for minimizing costs in [5] and tried to shorten the overall backup duration (*i.e.*, the backup time window) in [6]. However, since regular backup is performed when an inter-DC network is in normal state, it does not need to consider the time constraint and hence involves a simpler optimization than emergency backup. As an emergency backup is triggered to evacuate important data out before an upcoming disaster can impact the DC(s), it is time-constrained. Moreover, it is known that data itself may have different values [7] and different disasters can impose different levels of damages to DCs [4]. Hence, emergency backup should not simply treat all the data in endangered DCs equally as in the inter-DC data transfers of a specific cloud computing application [8, 9]. With this consideration, our previous work [10] prioritized the endangered data to maximize the data owners' utilities and leveraged the time-expanded network (TEN) approach to develop a distributed algorithm for optimizing emergency backup.

However, the approach in [10] still bears three issues. Firstly, The network model therein ignores the costs of resources used for buffering and transferring the data in inter-DC networks. Secondly, it uses the dual decomposition based sub-gradient method [11] to solve the optimization, which could have a relatively slow convergence speed. Lastly and most importantly, it does not consider the change of network topology during an emergency backup and assumes that the upcoming disaster would impact all the endangered DCs simultaneously. Apparently, this assumption is not true for all the disasters, for example, a hurricane or a typhoon would impact the inter-DC network in a time-variant manner [4].

In this paper, we extend our work in [10] to address the aforementioned issues. Specifically, we consider the emergency backup in an inter-DC network whose topology is time-variant and construct a variable TEN (V-TEN) to transform the dynamic backup into a static flow problem. Then, with the considerations of both data utility and resource cost, we formulate an optimization problem that maximizes the backup profit. Finally, we leverage the alternating direction method of multipliers (ADMM) [12] to develop a time-efficient and distributed algorithm to solve the problem. The major contributions of this work are as follow.

- To the best of our knowledge, this work is the first to study emergency backup in an inter-DC network, whose topology is time-variant.
- We propose an ADMM-based distributed algorithm for emergency backup, which is robust and time-efficient.

The rest of the paper is organized as follows. Section II describes the network model. In Section III, we formulate the optimization problem and propose the ADMM-based algorithm. The performance evaluation is discussed in Section IV. Finally, Section V summarizes the paper.

II. Network Model for Emergency Backup

A. Time-Variant Inter-DC Network

Fig. 1 shows an example of time-variant inter-DC networks during disasters. Basically, when a disaster happens, it would damage certain DC(s) at first (*e.g.*, DC 1 in Fig. 1), and then

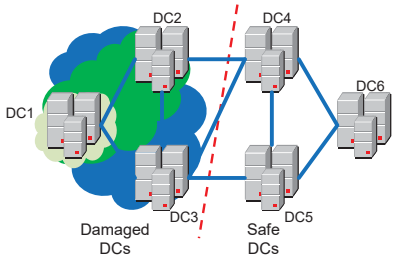
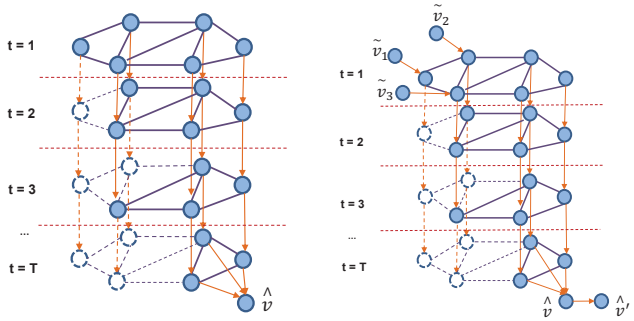


Fig. 1. Time-variant inter-DC network during emergency backup.



(a) Normal V-TEN (b) Modified V-TEN for ADMM

Fig. 2. Modeling with variable TEN (V-TEN) approaches.

with time going on, other DCs might become new victims (*e.g.*, DCs 2 and 3 in Fig. 1). Hence, if we take a snapshot of the inter-DC network at any instant, the DCs can be classified into two categories: damaged and safe ones. Consequently, the emergency backup should try to evacuate important data from the DCs that are predicted to be damaged to the safe ones at each instant, and during the process, the evacuated data can be buffered at certain DCs that would be safe for the time being.

We can denote the original topology of the inter-DC network as $G(V, E)$, where $V = \{1, \cdots, |V|\}$ is the set of DCs and E represents the set of links that connect the DCs. Then, we define a DC set V^d to include the DCs that would become damaged eventually due to the disaster, while the remaining DCs (*i.e.*, $V \setminus V^d$) would be safe for the whole process. For a DC $i \in V^d$, we assume that it would be safe for a period of T_i, which is its warning time. Hence, for the inter-DC network, the time window for emergency backup is $\max_{i \in V^d}(T_i)$. Note that, the warning time of each DC can be estimated according to environmental conditions [4], and thus we assume that it is known at the beginning of emergency backup. The emergency backup can be considered as a discrete-time system whose operation status only changes on time interval $t = \Delta t, 2\Delta t, \cdots$. Normalizing the time with Δt, we get the operation time as $t = 1, 2, \cdots, T$, where $T = \max_{i \in V^d}(T_i)/\Delta t$.

B. Modeling with Variable TEN (V-TEN)

Note that, to directly optimize the emergency backup in a time-variant inter-DC network, we need to solve a problem of fractional multi-commodity flows over time, which is known to be NP-hard [13]. Therefore, we introduce the preprocessing that leverages the time-expanded network (TEN) approach [14] to simplify the model. As the topology of the inter-DC network is time-variant, we modify the TEN approach to variable TEN (V-TEN) and use it to transform the dynamic emergency backup into a static flow problem.

Fig. 2(a) shows an example of V-TEN. Specifically, we first replicate the network topology for T times, where the t-th replica represents the topology at time t and is denoted as $G^t(V^t, E^t)$. The bandwidth of a link $e^t \in E^t$ is denoted as B_{e^t}, which represents the link's available bandwidth at time t. In order to represent the available DC storage space that can be used to buffer data at time t, we insert a directed link e_i^t from each DC i in G^t to the same DC in G^{t+1} and set its bandwidth $B_{e_i^t}$ as the available storage space of DC i at time t. Then, we define a link set $E^s = \{e_i^t,\ \forall t, i\}$ to include all these storage links. Note that, as the topology of the inter-DC network is time-variant due to the disaster, we then need to remove the DCs and links that would have become damaged at time t in G^t. The V-TEN $\mathcal{G}(\mathcal{V}, \mathcal{E})$ can be obtained by including the modified replicas and all the feasible storage links.

Apparently, when $t = 1$, all the DCs are up and running, *i.e.*, $V^1 = V$, and the DCs that would become damaged in the future are the sources of the emergency backup. While each of the DCs in $V \setminus V^d$ in G^T can be a feasible destination of the emergency backup. Next, we insert a super sink $\hat{v}$ in $\mathcal{G}$ to simply the flow routing. Specifically, $\hat{v}$ ends a directed virtual link from each safe DC in $V \setminus V^d$ in G^T. The bandwidth of each virtual link is the storage capacity of the DC from which it origins. Finally, we finish the preprocessing to construct the V-TEN $\mathcal{G}(\mathcal{V}, \mathcal{E})$, and then the dynamic emergency backup problem is transformed into a static flow problem to find the profit-maximized multi-commodity flow (PM-MCF) in $\mathcal{G}$, by considering both data utility and resource cost.

C. Models of Data Utility and Resource Cost

Similar to our work in [10], we still assume that the utility of emergency backup depends on the size of the backed-up data. Specifically, the utility function of a damaged DC $i \in V^d$ is defined as $f_i(s_i)$, where s_i is the total backed-up data that is successfully evacuated to safe DC(s) in $V \setminus V^d$. Here, since each endangered DC would try to pump data out in descending order of its value, we use a logarithmic function for $f_i(\cdot)$ [15]. Meanwhile, the costs of the storage and bandwidth resources used in the emergency backup should also be considered in the optimization, and thus we assign a unit bandwidth cost to each link in $\mathcal{G}$. Note that, the unit bandwidth cost of the virtual links to $\hat{v}$ is 0. Then, the emergency backup's profit is total utility minus total cost.

III. Problem Formulation and ADMM-based Distributed Algorithm

A. Optimization Model

For simplicity, we use link set $\mathcal{E}$ to include all the links in $\{E^t,\ \forall t\}$ and E^s. Hence, we can unify the representations of $e^t \in E^t$ and $e_i^t \in E^s$ as $e \in \mathcal{E}$. Then, the available bandwidth

of a link $e \in \mathcal{E}$ is denoted as B_e, and we use $b_{i,e}$ and $c_{i,e}$ to represent the bandwidth allocated on link e for backing up the data from DC $i \in V^d$ and its unit cost, respectively. Therefore, the optimization formulation of emergency backup is

Objective:

$$Maximize \quad \sum_{i \in V^d} \left[f_i(s_i) - \sum_{e \in \mathcal{E}} c_{i,e} \cdot b_{i,e} \right]. \tag{1}$$

Constraints:

1) *Data Size Constraint:*

$$s_i \leq C_i, \quad \forall i \in V^d, \tag{2}$$

where C_i represents the total amount of data on DC $i \in V^d$. Eq. (2) ensures that the amount of backed-up data would not exceed the originally stored data in each DC $i \in V^d$.

2) *Link Capacity Constraint:*

$$\sum_{i \in V^d} b_{i,e} \leq B_e, \quad \forall e \in \mathcal{E}. \tag{3}$$

Eq. (3) ensures that the aggregated bandwidth on each link would not exceed its capacity.

3) *Flow Conservation Constraint:*

$$\sum_{e \in v^+} b_{i,e} - \sum_{e \in v^-} b_{i,e} = \begin{cases} s_i, & v = i, \\ -s_i, & v = \hat{v}, \\ 0, & \text{Otherwise}, \end{cases} \quad \forall i \in V^d, v \in \mathcal{V}, \tag{4}$$

where $\mathcal{V}$ includes all the nodes in the V-TEN $\mathcal{G}$, and v^+ and v^- denote the sets of outgoing and incoming links of node v, respectively. Eq. (4) ensures that for any intermediate DC, the total of in- and out-flows for the data backup of DC i is 0, while the total amount of the data sent from DC i is equal to that of the data received by the super sink $\hat{v}$.

Note that, the V-TEN approach simplifies the optimization at the cost of an increased network size. Namely the numbers of variables and constraints *i.e.*, $|V^d| \cdot (1 + |\mathcal{E}|)$ and $|\mathcal{E}| + |V^d| \cdot (|\mathcal{V}| + 1)$, increase exponentially with the backup window T. In the following, we will leverage the alternating direction method of multipliers (ADMM) [12] to develop a time-efficient algorithm for the emergency backup.

B. Introduction of ADMM

ADMM is a powerful tool to solve structured optimizations, and especially fits for the design of distributed algorithms [12]. More specifically, ADMM solves problems in the form

$$\begin{aligned} Minimize \quad & f(\vec{x}) + g(\vec{z}) \\ s.t. \quad & A \cdot \vec{x} + B \cdot \vec{z} = \vec{c}, \end{aligned} \tag{5}$$

with variables $\vec{x}$ and $\vec{z}$, where A and B are constant matrices, $\vec{c}$ is a constant vector and $f(\cdot)$ and $g(\cdot)$ are convex functions. Thus, the variables are separated into two independent vectors, not only in the objective but also in the constraints. To solve the optimization in Eq. (5), ADMM tries to find the saddle point of the following augmented Lagrangian function.

$$\begin{aligned} L_\rho(\vec{x}, \vec{y}, \vec{z}) = & f(\vec{x}) + g(\vec{z}) + \vec{y}^T \cdot [A \cdot \vec{x} + B \cdot \vec{z} - \vec{c}] \\ & + \frac{\rho}{2} \cdot \|A \cdot \vec{x} + B \cdot \vec{z} - \vec{c}\|_2^2, \end{aligned} \tag{6}$$

where $\vec{y}$ is the vector of dual variables, $\|A \cdot \vec{x} + B \cdot \vec{z} - \vec{c}\|_2^2$ is the quadratic penalty term for accelerating the convergence and improving the robustness of the dual update, and $\rho > 0$ is the penalty parameter. Since the separation of variables $\vec{x}$ and $\vec{z}$ enables them to be optimized independently in an alternating manner, ADMM can divide a large-scale optimization into smaller subproblems that can be easily solved in parallel.

C. ADMM-based Distributed Algorithm

1) Transforming Optimization into ADMM Form: By comparing the optimization defined by Eqs. (1)-(4) to the standard form of ADMM in Eq. (5), we find that it cannot be solved with ADMM directly for two issues. Firstly, variables $\{s_i\}$ and $\{b_{i,e}\}$ are coupled in Eq. (4), making the variables inseparable in ADMM. Secondly, Eqs. (3) and (4) mix $\{b_{i,e}\}$ across DC nodes $\{i\}$ and $\{e\}$, respectively, which makes the subproblems extremely difficult to be solved.

In order to address the first issue, we add $|V^d| + 1$ virtual nodes to the V-TEN $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Specifically, for each damaged DC $i \in V^d$, we insert a virtual node $\tilde{v}_i$ and connect it to DC i with a virtual link $\tilde{e}_i$. While the available bandwidth on virtual link $\tilde{e}_i$ is set as the total amount of data on DC i (*i.e.*, $B_{\tilde{e}_i} = C_i$). Similarly, we also add a new virtual node $\hat{v}'$ to connect to the super sink $\hat{v}$. Then, we obtain the modified V-TEN $\mathcal{G}'(\mathcal{V}', \mathcal{E}')$ for ADMM as shown in Fig. 2(b). It can be seen that in the modified V-TEN $\mathcal{G}'$, the original sources and destination for the emergency backup become intermediate nodes. Hence, the flow conservation constraint in Eq. (4) becomes 0 on the right side for all the original nodes in $\mathcal{V}$. Moreover, variables $\{s_i\}$ become $\{b_{i,\tilde{e}_i}\}$ and thus Eqs. (2) and (3) can be unified.

For the second issue, we introduce auxiliary variables $\{z_{i,e}\}$ to reformulate the optimization such that the link capacity and flow conservation constraints can be decoupled.

$$z_{i,e} = b_{i,e}, \quad \forall i \in V^d, \ e \in \mathcal{E}'. \tag{7}$$

Hence, Eqs. (2) and (3) are unified as

$$\sum_{i \in V^d} z_{i,e} \leq B_e, \quad \forall e \in \mathcal{E}'. \tag{8}$$

Then, we introduce a set of adjacent parameters $\{a_{v,e}\}$ to indicate the relation among links and nodes. Specifically, if $e \in v^+$, we have $a_{v,e} = 1$, if $e \in v^-$, we have $a_{v,e} = -1$, and $a_{v,e} = 0$ otherwise. Note that, as we only care about the flows in the original V-TEN $\mathcal{G}$, we set all the adjacent parameters that relate to the newly-added virtual nodes/links as 0, and thus the flow conservation constraints in Eq. (4) become

$$\sum_{e \in \mathcal{E}'} a_{v,e} \cdot b_{i,e} = 0, \quad \forall i \in V^d, \ v \in \mathcal{V}'. \tag{9}$$

Next, we introduce another set of parameters to indicate whether a link $e \in \mathcal{E}'$ is a newly-added virtual link $\tilde{e}_i$, *i.e.*, if $e = \tilde{e}_i, i \in V^d$, we have $h_{i,e} = 1$, and $h_{i,e} = 0$ otherwise. Finally, the optimization defined by Eqs. (1)-(4) gets transformed into

$$\begin{aligned} Minimize \quad & \sum_{i \in V^d} \sum_{e \in \mathcal{E}'} [c_{i,e} \cdot z_{i,e} - f_i(h_{i,e} \cdot b_{i,e})] \\ s.t. \quad & \text{Eqs. (7)-(9).} \end{aligned} \tag{10}$$

It can be seen that the optimization in Eq. (10) takes the standard ADMM form, and hence could be solved time-efficiently with a distributed algorithm [12].

2) Solving Optimization with ADMM Approach: We design an augmented Lagrangian function to solve the optimization in Eq. (10) with ADMM

$$\begin{aligned} L_\rho = &\sum_{i\in V^d}\sum_{e\in\mathcal{E}'}[-f_i(h_{i,e}\cdot b_{i,e})+c_{i,e}\cdot z_{i,e}] \\ &+\sum_{i\in V^d}\sum_{e\in\mathcal{E}'}\left[\varphi_{i,e}\cdot(b_{i,e}-z_{i,e})+\frac{\rho}{2}\|b_{i,e}-z_{i,e}\|_2^2\right], \end{aligned} \tag{11}$$

where $\{\varphi_{i,e}\}$ are the dual variables. Then, we obtain the subproblems for the variables and solve them as follows.

Step 1 ($z_{i,e}$-*minimization*): We take variables $\{z_{i,e}\}$ from the Lagrangian function in Eq. (11) and solve the following subproblem for each link $e\in\mathcal{E}'$ in iterations.

$$\begin{aligned} Minimize \quad &\sum_{i\in V^d}\left[\frac{\rho}{2}\cdot z_{i,e}^2-\left(\varphi_{i,e}^{(k)}+\rho\cdot b_{i,e}^{(k)}-c_{i,e}\right)\cdot z_{i,e}\right] \\ s.t. \quad &\text{Eq. (8)}, \end{aligned} \tag{12}$$

where the variables with superscript "(k)" mean that they are obtained in the k-th iteration. In each iteration, the optimization is a quadratic program with a series of inequality constraints, which can be solved exactly by using the Karush-Kuhn-Tucker (KKT) conditions [11]. Specifically, in the $(k+1)$-th iteration, variables $\{z_{i,e}\}$ are updated as follows

$$z_{i,e}^{(k+1)}= \begin{cases} \max\left(b_{i,e}^{(k)}+\dfrac{\varphi_{i,e}^{(k)}-c_{i,e}}{\rho},0\right), \sum\limits_{j\in V^d}\left(b_{j,e}^{(k)}+\dfrac{\varphi_{j,e}^{(k)}-c_{j,e}}{\rho}\right)\le B_e, \\ \max\left(b_{i,e}^{(k)}+\dfrac{\varphi_{i,e}^{(k)}-\gamma_e^{(k+1)}-c_{i,e}}{\rho},0\right), \text{Otherwise}, \end{cases} \tag{13}$$

where the parameter $\gamma_e^{k+1}>0$ is introduced when for a specific link e, the constraint in Eq. (8) cannot be satisfied. Hence, we use γ_e^{k+1} to decrease the value of the corresponding $z_{i,e}^{(k+1)}$ and ensure that in each iteration, variables $\{z_{i,e}\}$ represent a feasible solution to the optimization in Eq. (12).

Step 2 ($b_{i,e}$-*minimization*): With the variables $z_{i,e}^{(k+1)}$ obtained in Step 1, the subproblem to get $\{b_{i,e}\}$ is as follows.

$$\begin{aligned} Minimize \quad &\sum_{i\in V^d}\sum_{e\in\mathcal{E}'}\Big[b_{i,e}\cdot(\frac{\rho}{2}\cdot b_{i,e}+\varphi_{i,e}^{(k)}-\rho\cdot z_{i,e}^{(k+1)}) \\ &-f_i(h_{i,e}\cdot b_{i,e})\Big] \\ s.t. \quad &\text{Eq. (9)}. \end{aligned} \tag{14}$$

Then, to address the optimization of network flow with strict flow conservation, we introduce a Lagrangian multiplier $\delta_{i,v}$ to relax the constraints in the subproblem [13]. Hence, the optimization in Eq. (14) is reformulated as

$$\begin{aligned} Minimize \quad &\sum_{i\in V^d}\sum_{e\in\mathcal{E}'}\left(\varphi_{i,e}^{(k)}-\rho\cdot z_{i,e}^{(k+1)}+\sum_{v\in\mathcal{V}'}\delta_{i,v}^{(k)}\cdot a_{v,e}\right)\cdot b_{i,e} \\ &+\sum_{i\in V^d}\sum_{e\in\mathcal{E}'}\left[\frac{\rho}{2}\cdot b_{i,e}^2-f_i(h_{i,e}\cdot b_{i,e})\right], \end{aligned} \tag{15}$$

and in the iterations, $\{\delta_{i,v}\}$ are updated as

$$\delta_{i,v}^{(k+1)}=\delta_{i,v}^{(k)}+\lambda\cdot\sum_{v\in\mathcal{V}'}a_{v,e}\cdot b_{i,e}^{(k+1)}, \tag{16}$$

where λ is the step size. Note that, depending on the form of $f_i(\cdot)$, the subproblem on each link in Eq. (15) can be solved with a standard mathematical approach. For instance, if we adopt a logarithmic function as $f_i(s_i)=\alpha_i\cdot\log(1+s_i)$, where α_i is the utility weight of DC i [10], we can apply the Taylor series expansion to $f_i(\cdot)$, convert the optimization to a quadratic problem, and then obtain the analytical solution as

$$b_{i,e}^{(k+1)}=\max\left(\frac{\rho\cdot z_{i,e}^{(k+1)}-\varphi_{i,e}^{(k)}-g(b_{i,e}^{(k)})+\delta_{i,e^+}^{(k)}-\delta_{i,e^-}^{(k)}}{\rho+L(b_{i,e}^{(k)})},0\right), \tag{17}$$

where we have $g(b_{i,e}^{(k)})$ and $L(b_{i,e}^{(k)})$ as

$$g(b_{i,e}^{(k)})=\frac{\alpha_i\cdot h_{i,e}}{\log(10)}\cdot\frac{1+2\cdot b_{i,e}^{(k)}}{(1+b_{i,e}^{(k)})^2}, L(b_{i,e}^{(k)})=\frac{\alpha_i\cdot h_{i,e}}{\log(10)\cdot(1+b_{i,e}^{(k)})^2}, \tag{18}$$

and define e^- and e^+ as the two end-nodes of a directed link e, *i.e.*, $e=(e^-,e^+)$ [10].

Step 3 ($\varphi_{i,e}$-*update*): With the variables $\{z_{i,e}^{(k+1)}\}$ and $\{b_{i,e}^{(k+1)}\}$, we can update the dual variables $\{\varphi_{i,e}\}$ as

$$\varphi_{i,e}^{(k+1)}=\varphi_{i,e}^{(k)}+\rho\cdot(b_{i,e}^{(k+1)}-z_{i,e}^{(k+1)}). \tag{19}$$

Finally, the ADMM-based distributed algorithm for emergency backup is presented in *Algorithm* 1, if we assume that the utility function $f_i(\cdot)$ takes the form of a logarithmic function, *i.e.*, $f_i(s_i)=\alpha_i\cdot\log(1+s_i)$. Note that, with *Algorithm* 1, we solve the subproblems for each link independently since in each iteration, all the variables related to a link can be updated with local information, *i.e.*, the values of variables related to the link or its end-nodes in the previous iteration. Hence, the ADMM-based algorithm is a distributed one and can be implemented in a parallel manner for high time-efficiency.

IV. Performance Evaluation

A. Simulation Setup

We use an inter-DC network with the NSFNET topology in Fig. 3 to evaluate the performance of our ADMM-based emergency backup algorithm (ADMM). Basically, we assume that each node in the topology hosts a DC and at most five of them (*i.e.*, DCs 10-14 in Fig. 3) could be impacted by an upcoming disaster within a time window of T. In each simulation, the actual number of damaged DCs (*i.e.*, $|V^d|$) is selected within $[2,5]$ to emulate different disaster severities. Specifically, there are four disaster scenarios with DC-down sequences as $V^d=\{14,13\}$, $V^d=\{14,13,12\}$, $V^d=\{14,13,12,11\}$, and $V^d=\{14,13,12,11,10\}$, which can happen within different time windows. In each simulation, the damaged DCs will be impacted at random time according to a sequence. The amount of data on each damaged DC is set randomly, while the total data for backup is fixed as 400 TBytes. Similarly, the storage capacity of each safe DC is also randomly chosen under the constraint that the total capacity is

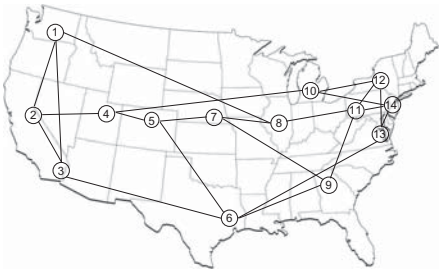


Fig. 3. Inter-DC network with NSFNET topology.

500 TBytes. The utility function takes the form of $f_i(x_i) = \alpha_i \cdot \log(1 + x_i)$, where for DCs $i = \{14, 13, 12, 11, 10\}$, $\alpha_i = \{100, 120, 150, 200, 160\}$, respectively. The bandwidth capacity of each link is chosen from $[30, 80]$ Gbps, while the unit bandwidth cost ranges within $[0.01, 0.015]$ units per TByte. The simulations run on a computer with 3.1 GHz Intel Core i3-2100 CPU and 8 GB RAM.

Algorithm 1: ADMM-based Distributed Algorithm

$k = 0, \{b_{i,e}^{(0)}\} = \mathbf{0}, \{\varphi_{i,e}^{(0)}\} = \mathbf{0}, \{\delta_{i,v}^{(0)}\} = \mathbf{0}$;
while *the solution has not converged* **do**
  get values of $\{z_{i,e}^{(k+1)}\}$ with Eq. (13);
  get values of $\{b_{i,e}^{(k+1)}\}$ with Eq. (17);
  get values of $\{\delta_{i,v}^{(k+1)}\}$ with Eq. (16);
  get values of $\{\varphi_{i,e}^{(k+1)}\}$ with Eq. (19);
  $k = k + 1$;
end

We compare ADMM with three benchmark algorithms. The first one is the highest-utility-data-first algorithm (HUDF), which sorts all the data for backup in descending order of the utility at each backup time, and then it calculates the corresponding maximum flows to transmit the data in sequence in the best-effort manner. The procedure is repeated until the time window T ends or all the data for backup is evacuated out. The second one is the V-TEN based HUDF algorithm (VTEN-HUDF). Specifically, we first construct a V-TEN as described in Section II-B and then apply the HUDF algorithm in the V-TEN. For the last one, we leverage the sub-gradient approach that we proposed in [10] and modify it to fit to V-TEN. Specifically, we use the dual decomposition based on the sub-gradient method [11] to solve the original optimization described in Section III-A. Hence, in the following discussions, we refer to this benchmark algorithm as Sub-Grad.

B. Convergence Performance

We first obtain the duality gap of ADMM, which is the difference between the prime solution and that from ADMM. The prime solution is obtained by solving the original optimization in Section III-A directly. Firstly, we set the time window $T = 6$ minutes[1], select $|V^d| = 4$, and plot the duality gap of ADMM

[1]In all the simulations, we have the time interval as $\Delta t = 1$ minute.

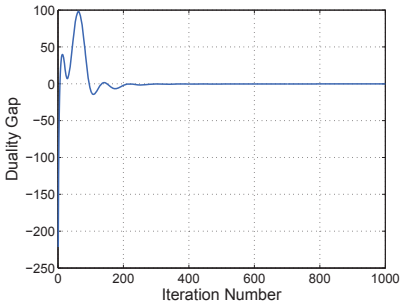


Fig. 4. Duality gap of ADMM ($T = 6$ and $|V^d| = 4$).

TABLE I
ITERATION NUMBERS FOR ADMM TO CONVERGE UNDER DIFFERENT ACCURACY REQUIREMENTS

$\lvert V^d\rvert$	2		3		4	5
T / Accuracy	3	6	9	12	15	18
10	70	150	160	310	510	550
1	160	340	830	1700	2190	5310
0.1	1570	2850	2820	4670	5910	15950

in Fig. 4. It can be seen that the duality gap converges within 300 iterations. Then, we try more combinations of T and $|V^d|$ and obtain the iteration numbers for ADMM to converge under different accuracy requirements. Table I shows the results. Note that, the simulations have verified that ADMM takes less than 1.1 second to finish 15950 iterations, and thus the results in Fig. 4 and Table I confirm the time-efficiency of ADMM.

C. Robustness Analysis

We then investigate the robustness of ADMM. Specifically, since ADMM introduces Lagrangian multipliers to relax the flow conservation constraints in dual decomposition, we can change the value of the step size λ to evaluate its robustness. With $T = 6$, $|V^d| = 4$ and the accuracy requirement as 1, Table II shows the iteration numbers and running time for ADMM to converge using different λ. We observe that ADMM converges faster with a larger λ and it can always converge within 0.2 second when λ changes from 0.001 to 0.1. Note that, we also verify that ADMM performs similarly for other combinations of T and $|V^d|$. Therefore, we confirm that the convergence of ADMM is robust and would not be seriously affected by λ. On the contrary, the convergence performance of Sub-Grad depends on the choice of λ heavily. In other words, without a proper λ, Sub-Grad might not be able to converge. Moreover, even after optimizing λ, Sub-Grad converges much slower than ADMM. For example, Fig. 5 compares the duality gaps obtained by ADMM and Sub-Grad for $T = 6$ and $|V^d| = 4$. We find that ADMM converges within 300 iterations, while Sub-Grad needs around 20000 iterations to converge even with the optimal step size $\lambda = 0.000378$. Meanwhile, our simulations also verify that ADMM is not sensitive to parameter ρ either (*i.e.*, the conclusion of [12]), and the proper value of ρ can be obtained easily.

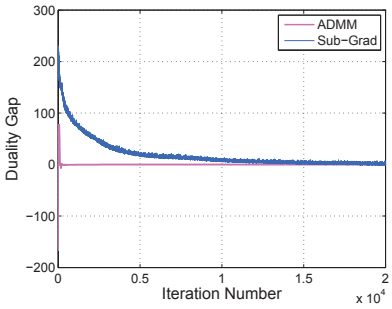


Fig. 5. Comparison on the convergence performance of ADMM and Sub-Grad ($T = 6$ and $|V^d| = 4$).

TABLE II
ITERATION NUMBERS AND RUNNING TIME FOR ADMM TO CONVERGE WITH DIFFERENT STEP SIZES ($T = 6$ AND $|V^d| = 4$)

λ	0.001	0.005	0.01	0.02	0.05	0.1
Iteration number	6300	2700	2600	1960	900	600
Time (seconds)	0.177	0.103	0.099	0.076	0.038	0.030

D. Comparisons with Benchmark Algorithms

Finally, we compare ADMM with the benchmarks in terms of the backup profit defined in Eq. (1) and the running time, with different combinations of T and $|V^d|$. Table III shows the backup profits from the algorithms. Here, "Optimal" refers to the scheme that solves the original optimization in Section III-A directly. It can be seen that the profits from ADMM are always higher than those from the benchmarks and they are almost the same as the optimal results (*i.e.*, satisfying the accuracy requirement of 1). Note that, for $T = 30$ and $|V^d| = 5$, the problem scale is too large and thus we have difficulty to obtain the optimal profit directly. The profits from Sub-Grad are higher than those from VTEN-HUDF and HUDF, and because the V-TEN approach helps to utilize the storage capacities of DCs for buffering data, VTEN-HUDF outperforms HUDF in terms of profit. Table III also lists the running time of the algorithms. As expected, solving the original optimization directly is much more time-consuming than the rest of the algorithms. Due to their time complexities, the running time of VTEN-HUDF and Sub-Grad is also relatively long. Although HUDF performs the worst in terms of profit, it is the most time-efficient algorithm. The running time of ADMM is comparable to that of HUDF, and it is interesting to notice that when the problem scale increases with larger T and $|V^d|$, the running time of ADMM increases much slower than that of HUDF. This is because ADMM is a distributed algorithm. Finally, we can conclude that among all the algorithms, ADMM achieves the best tradeoff between the backup profit and time-efficiency.

V. CONCLUSION

This paper studied the emergency backup in an inter-DC network whose topology is time-variant and constructed a variable TEN (V-TEN) to transform the dynamic backup into a static flow problem. Then, we leveraged the ADMM approach to develop a time-efficient and distributed algorithm to solve the problem. Simulation results showed that our ADMM-based algorithm outperformed several existing ones.

TABLE III
RESULTS ON BACKUP PROFIT AND RUNNING TIME

$\lvert V^d\rvert$	2	3		4	5
T	6	9	15	20	30
	Backup Profit (units)				
Optimal	292.90	552.14	615.58	975.82	–
HUDF	272.63	536.32	593.73	941.50	1261.77
VTEN-HUDF	289.09	542.96	606.75	968.89	1275.85
Sub-Grad	290.20	549.98	612.05	969.53	1279.1
ADMM	292.90	552.14	615.22	975.49	1283.31
	Running Time (seconds)				
Optimal	6.243	19.548	35.685	46.612	–
HUDF	0.072	0.051	0.031	0.213	0.202
VTEN-HUDF	0.068	0.302	1.0	3.347	6.892
Sub-Grad	0.535	1.305	2.195	4.656	3.405
ADMM	0.139	0.160	0.272	0.279	0.243

ACKNOWLEDGMENTS

This work was supported in part by the NSFC Project 61371117 and 61573331, Natural Science Research Project for Universities in Anhui (KJ2014ZD38), and the Strategic Priority Research Program of the CAS (XDA06011202).

REFERENCES

[1] P. Lu *et al.*, "Highly-efficient data migration and backup for big data applications in elastic optical inter-datacenter networks," *IEEE Netw.*, vol. 29, pp. 36–42, Sept./Oct. 2015.
[2] [Online]. Available: https://en.wikipedia.org/wiki/2008_Sichuan_earthquake.
[3] J. Yao, P. Lu, L. Gong, and Z. Zhu, "On fast and coordinated data backup in geo-distributed optical inter-datacenter networks," *J. Lightw. Technol.*, vol. 33, pp. 3005–3015, Jul. 2015.
[4] B. Mukherjee, M. Habib, and F. Dikbiyik, "Network adaptability from disaster disruptions and cascading failures," *IEEE Commun. Mag.*, vol. 52, pp. 230–238, May 2014.
[5] A. Bianco, L. Giraudo, and D. Hay, "Optimal resource allocation for disaster recovery," in *Proc. of GLOBECOM 2010*, pp. 1–5, Dec. 2010.
[6] J. Yao, P. Lu, and Z. Zhu, "Minimizing disaster backup window for geo-distributed multi-datacenter cloud systems," in *Proc. of ICC 2014*, pp. 3631–3635, Jun. 2014.
[7] [Online]. Available: http://en.wikipedia.org/wiki/Big_data.
[8] P. Lu, Q. Sun, K. Wu, and Z. Zhu, "Distributed online hybrid cloud management for profit-driven multimedia cloud computing," *IEEE Trans. Multimedia*, vol. 17, pp. 1297–1308, Aug. 2015.
[9] K. Wu, P. Lu, and Z. Zhu, "Distributed online scheduling and routing of multicast-oriented tasks for profit-driven cloud computing," *IEEE Commun. Lett.*, vol. 20, pp. 684–687, Apr. 2016.
[10] P. Lu, Q. Ling, and Z. Zhu, "Maximizing utility of time-constrained emergency backup in inter-datacenter networks," *IEEE Commun. Lett.*, vol. 20, pp. 890–893, May 2016.
[11] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, Apr. 2004.
[12] S. Boyd *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, pp. 1–122, Jan. 2011.
[13] M. Skutella, "An introduction to network flows over time," in *Research Trends in Combinatorial Optimization*, W. Cook, L. Lovasz, and J. Vygen, Eds. Springer, 2009, ch. 21, pp. 451–482.
[14] L. Ford and D. Fulkerson, "Constructing maximal dynamic flows from static flows," *Oper. Res*, vol. 6, pp. 419–433, Jun. 1958.
[15] P. Lu, K. Wu, Q. Sun, and Z. Zhu, "Toward online profit-driven scheduling of inter-DC data-transfers for cloud applications," in *Proc. of ICC 2015*, pp. 7186–7191, Jun. 2015.