Your Paper

Yijing Kong

September 27, 2018

Abstract

Your abstract.

1 Introduction

This document summarizes the heuristic algorithm and the algorithm of offline model.

2 Mathematics

2.1 Network model

The subsection describes the network model following the P2MP transfer problem.

2.2 Heuristic Algorithm

Algorithm 1 Semi-flexible Transfer Source, compared to Completed-flexible Transfer Source

Input: $req = (s, d^1, ..., d^m, f, t^{start}, t^{end})$;
$\mathcal{P}_{u,v} = \{P_1, P_2, \ldots, P_k\}$: k-shortest paths between the datacenter u and v;
$c_{e,t}$: residual link bandwidth on link $e \in E$ at time $t \in [t^{start}, t^{end}]$.
Output: $r = \{0, 1\}$: the current request whether be accepted or not.

```
src ← {s}
dst ← {d^1, ..., d^m}
r ← 0
while |dst| > 0 do
    for s = 0 → |src| − 1 do
        for d = 0 → |dst| − 1 do
            computing transfer time from src[s] to dst[d] and then geting a matrix T[s,d];
        end for
    end for
    finding the minimum time from T and with the corresponding longtitude and latitude,
    [s_min, d_min]

end while
return r
```

2.3 Offline Model and XXX Algorithm

The first paragraph: REVIEW.
Determining transfer sources: We use a binary $h_{i,u}^t$ to denote whether any one of datacenters u can serve as the transfer source at time t $\in$ [0, T] for $req_i \in \mathcal{R}$. For one of requests, we certainly have

$$\forall i, u, t : h_{i,u}^t src_{i,u} = (survival)_i^t \tag{1}$$

$$\forall i, u, t: h_{i,u}^t dst_{i,u} \le (survival)_i^t \tag{2}$$

$$\forall i, u, t: h_{i,u}^t (1 - src_{i,u} - dst_{i,u}) = 0 \tag{3}$$

where contant $(survival)_i^t$, $src_{i,u}$ and $dst_{i,u}$ as binary respectively denote the real lifetime, the possible source datacenters at time t ∈ [0, T], the required destination datacenters for a request. While for any datacenters, it can become an avalible transfer source at time t if it has received the complete package of data f by the end of timeslot $t-1$ for any requests:

$$\forall i, t, v: \sum_{\tau=0}^{t-1} \sum_{u, u \ne v} x_{i,u,v}^{\tau} \ge f_i h_{i,v}^t \tag{4}$$

where $x_{i,u,v}^{\tau} \ge 0$ represents the data volume received at destination v from datacenter u at time $\tau \in [0, t-1]$ for i^{th} request. Meanwhlie, a datacenter u is able to transfer data to datacenter $v, v \ne u$ only if it is an available transfer source:

$$\forall i, t, u, v, u \ne v: x_{i,u,v}^t \le f_i h_{i,u}^t \tag{5}$$

To be practical, each destination v is restricted to set up transfer connection with a single source datacenter in the process of data delivery:

$$\forall i, v, u \ne v: \sum_{u, u \ne v} z_{i,u,v} = 1 \tag{6}$$

where binary $z_{i,u,v}$ denote whether datacenter u is the transfer source of destination v for i^{th} request. Clearly, there should be

$$\forall i, t, u, v, u \ne v: x_{i,u,v}^t \le f_i z_{i,u,v} \tag{7}$$

Allocating available bandwidth resources: Let $y_{i,u,v,p}^t$ denote the bandwidth resources that allocated along the routing path P from datacenter u to datacenter v at time t for i^{th} request. The allocations are feasible if:

$$\forall i, t, u, v, u \ne v: \sum_{p \in \mathcal{P}_{u,v}} \alpha y_{i,u,v,p}^t \ge x_{i,u,v}^t \tag{8}$$

$$\forall e, t: \sum_i \sum_u \sum_v \sum_{p \in \mathcal{P}_{u,v}} y_{i,u,v,p}^t I(e \in P) \le c_{e,t} \tag{9}$$

where contant α denotes the length of each time slot and expression $\sum_i \sum_u \sum_v \sum_{p \in \mathcal{P}_{u,v}} y_{i,u,v,p}^t I(e \in P)$ denotes the amount of traffic on link e at time $t \in [0, T]$ for all of requests. Equation (8) states that the allocated routing and bandwidth resources should at least be capable of transferring the damanding data size for i^{th} request; Equation (9) expresses the link load is restricted to not exceed the capacity to avoid link congestion. Meanwhlie, if the real numbers of path between u and v less than the default path number, there should be:

$$\forall i, t, u, v, u \ne v: y_{i,u,v,p}^t \le plen_{u,v,p} f_i \tag{10}$$

where binary contant $plen_{u,v,p}$ denotes whether the p^{th} path between u and v exists or not.
Guaranteeing deadline for each request: This needs us to ensure the data transfer can be completed before the deadline for every destination datacenter. Let binary $\omega_{i,d}$ denotes whether the destination v has received all the data before deadline for the i^{th} request. Guaranteeing the dealine for destination v requires the transfer to be completed no later than time T. So we have

$$\forall i, v: \sum_{t=0}^{T} \sum_{u, u \ne v} x_{i,u,v}^t \ge f_i \omega_{i,v} \tag{11}$$

Let binary r_i denotes the i^{th} request whether be accepted. If the number of completed destinations equals to the number of required destinations for a request, we accept the request and transfer the required size data during the scheduled transfer period. Clearly, there should be

$$\forall i : \sum_{v} \omega_{i,v} = r_i M_i \tag{12}$$

where contant M_i denotes the number of destinations for the i^{th} request.

Maximizing the accepted requests: Our offline model for verifying the basic algorithm is straightforward and is summarized in Algorithm 4. Given G and the set R including all of deadline-constrained P2MP transfer requests at time $t \in [0, T]$, our goal is to maximize the number of accepted requests before time T through solving the corresponding MIP problem in Algorithm 4.

XXX Algorithm

Algorithm 4

1: Input:
$\mathcal{R} = \{req_1, req_2, \ldots, req_i, \ldots, req_n\}$: all of the requests arrived at time t $\in$ [0, T];
$req_i = (s_i, d_i^1, \ldots, d_i^m, f_i, t_i^{start}, t_i^{end})$: a transfer request;
$\mathcal{P}_{u,v} = \{P_1, P_2, \ldots, P_k\}$: k-shortest paths between the datacenter u and v;
$c_{e,t}$: residual link bandwidth on link $e \in E$ at time t $\in$ [0, T];

2: output:
Return the completion status r_i in the solution of the following problem:

$$max \sum_{r_i \in \mathcal{R}} r_i$$

$$s.t. \text{ constrains } (1)\sim(12)$$