Update Aware Replica Placement

Assaf Rappaport
Department of Computer Science
Technion, Israel Institute of Technology
Haifa 32000, Israel
Email: assafr@cs.technion.ac.il

Danny Raz
Department of Computer Science
Technion, Israel Institute of Technology
Haifa 32000, Israel
Email: danny@cs.technion.ac.il

***Abstract*—In recent years, companies such as eBay, Facebook, Google, Microsoft, and Yahoo! have made large investments in massive data centers supporting cloud services. These data centers are becoming the hosting platform for a wide spectrum of composite applications with an increasing trend towards more communication intensive applications. As a result, the bandwidth requirements within and between data centers is rapidly growing, and the efficient management of these networking resources is becoming a key ingredient in the ability to offer cost effective cloud services.**

Replica placement is a specific aspect of cloud management where the goal is to optimally place the applications and their related data over the available cloud infrastructure. The problem is inherently complex since data is continuously updated, and the cost associated with this update increases with the number of data replica and the network distance between them. We model this problem as a soft-capacitated connected facility location problem, which is NP-Hard in the general case. We present the first deterministic constant approximation algorithm for this problem and show, using extensive simulations and realistic data center and network topology, that our algorithm provides practically good placement decisions.

I. Introduction

Data centers and cloud services are continuing to grow rapidly, with ever more functionality and ever more users around the globe [1]. Because of this growth, major cloud service providers now use tens of geographically dispersed data centers, and they keep on building more. These data centers are becoming the hosting platform for a wide spectrum of composite applications. Companies such as eBay, Facebook, Google, Microsoft, and Yahoo! are reported to have made large investments in building massive data centers supporting cloud services. In particular, there is an increasing trend towards more communication intensive applications in data centers, and thus bandwidth usage is rapidly growing both inside and between data centers [2].

Replica placement across data centers is a very common approach for improving performance and availability of content services. Content replication algorithms deploy a set of servers, distributed throughout the data center's network, and replicate the relevant data across these servers. Both the time required to access the data and the traffic in the network are reduced by redirecting the application's requests to a nearby replica. The deployment of multiple replica servers also achieves content servers' redundancy and improves the availability of the system. In the past, replica placement algorithms were mainly considered in the context of web pages and CDNs (Content Distribution Networks) [20], [21], [3].

One can divide the research problems associated with replica placement across data centers into the how, what and where aspects. How to direct the client's requests to the proper replica server having the desired content (the request routing system), what content should be distributed to the replica servers across the network (the content selection aspect) and where to place the replica servers throughout the network while keeping them up to date.

In this work, we consider the problem of placing replicas of an object at multiple locations in the network. Consider an email application which depends on an authentication service. We focus on the problem of placing replicas of an object (e.g., the authentication service) at multiple locations. Replica placement deals with the actual number and network location of the replicas. Clearly, we would like to minimize the network distance between an email application and the closest replica containing the desired content (in this example the authentication server) and thus having more replicas helps. On the other hand, having more replicas is more expensive so we need to model the cost and the benefit in a way that can allow to make the appropriate decisions regarding the number and the network locations of the replicas. This problem is strongly related to a family of optimization problems generally referred to as *facility location* problems [14].

Most of the existing algorithms neglect the cost of keeping the replicas across the network up to date, and in cases where this cost is non-neglectable this may lead to suboptimal realistic solutions. A replica must be synchronized with the original content server in order to supply reliable and precise service to the client requests. The amount of synchronization traffic across the network depends on the number of replicas deployed in the network, the topology of the distributed update and the rate of updates in the content of the server.

Our work considers a network design problem that combines a facility location and connectivity problem. In a vigorous content world of dynamic and interactive services, the replica's content must be synchronized and up to date, and the update process of the replica hosting servers may be a significant factor of the network traffic load. We assume that the hosting servers are updated simultaneously using multicast

This research was supported in part by the Israel Science Foundation (grant no. 1129/10).

over a minimal Steiner tree, in such a case the update cost is the update rate multiply by the tree cost. The problem is to choose the best locations for the replicas (or hosting servers) among the potential sites. We model the scenario above as a Soft-Capacitated Connected Facility Location Problem which is NP-Hard in the general case. We assume that each client uses a single replica (of course, multiple clients can use the same replica). In other words, a client gets all of its content from the same replica. In this paper, we present a deterministic constant approximation algorithm to the soft capacitated connected facility location problem. To the best of our knowledge, this work is the first to present a deterministic constant approximation algorithm for this problem.

We evaluate the performance of our proposed algorithm through extensive simulation experiments using a realistic data center network topology. We compare our scheme to existing alternatives (a greedy algorithm and a local-search facility location algorithm), on publicly available data regarding the Google data center network. The results indicate that our new algorithm preforms better than the currently available algorithms, and this improvement can be significant at certain settings.

The rest of the paper is organized as follows: In Section 2 we survey previous work. In Section 3 we present the problem formulation and the necessary definitions and notations. In Section 4 we present our deterministic constant approximation algorithm for the soft capacitated connected facility location problem and in Section 5 we evaluate the performance of this algorithm through extensive simulation experiments on realistic scenarios.

II. Related Work

One of the most popular services that can be offered through the Cloud is Software as a Service or SaaS [10]. A SaaS deployed in a Cloud is usually composed of several components, where each of the components represents a business function of the SaaS that is being delivered [11]. The problem of placing the components of a SaaS and their related data in the Cloud is referred to as SaaS Placement Problem (SPP). Kwok and Mohindra [12] consider a placement problem for SaaS components in a multi-tenant architecture. The placement of the components is made within a set of available servers, and the main objective is to optimize the resource usage in each server. Although concerned with SaaS placement, that work is more focused on the multi-tenant resource model and does not take into consideration the SaaS's data placement in the network. Yusoh and Maolin [13] investigate the placement of applications and their related data within cloud environments. A penalty based genetic algorithm is proposed as a solution to the placement problem. The solution aims at placing the processing algorithm on a compute node that has a better bandwidth value with respect to the storage node.

In this work we consider a network design problem that combines both facility location and connectivity problems. The Connected Facility Location Problem has a wide range of applications and has recently received considerable attention both in the theoretical computer science literature and in the operations research literature.

Karger and Minkoff [5] introduced the so-called maybecast problem which is a probabilistic version of the Steiner tree problem. The name connected facility location has been introduced by Gupta et al. [7] in their work on virtual private networks. They prooved ConFL is NP-hard and improved the previous result by introducing a 10.66 factor approximation algorithm. Their algorithm was based on a linear programming (LP) rounding technique. Since then several authors proposed approximation algorithms for diverse variants of ConFL. Recently, Eisenbrand et al. [8] combined approximation algorithms for the basic facility location problem and the connectivity problem of the opened facilities by running a randomized approximation algorithm with an expected approximation ratio of 4 for ConFL.

The capacitated facility location problem and variations of it have been well-studied in the literature [14] [15] [16]. Leitner and Raidl [17] introduced the first variant of capacitated connected facility location problem. The authors considered a connected facility location problem where only clients which are reasonable from an economic point of view need to be served (prize collecting variant) with capacity constraints on possible facilities (PCConFL). They presented two Variable Neighborhood Search (VNS) approaches for a variant of PC-ConFL without assignment and opening costs. Subsequently, the same authors [18] [19] proposed a Lagrangian relaxation based approach which has been hybridized with local search and very large scale neighborhood search as well as two mixed integer programming models based on multi-commodity flows.

Recently, Eisenbrand et al. [9] presented the connected soft-capacitated facility location problem (soft-ConFL). Using the same techniques as in [8], the authors gave a randomized approximation algorithm with an expected approximation ratio of 6.27. In this paper we provide a deterministic algorithm for this problem with a guaranteed approximation ratio, to the best of our knowledge, this work is the first such deterministic algorithm for this problem. Moreover, we show that unlike the somewhat complex theoretical nature of [9], our approach is very practical and can be easily implemented on realistic data.

III. Problem Model

In this section we describe the *Soft-Capacitated Connected Facility Location Problem*, introduce several notations and definitions that are useful for the analysis of our problem, and define the r-gathering problem which will be used to prove our main result.

We are given an undirected graph $G = (V, E)$ with non-negative symmetric costs $c_{i,j}$ $(i, j \in V)$ on the edges. Let $F \subseteq V$ be a set of locations where facilities may be placed and $C \subseteq V$ be a set of demand nodes or clients that must be assigned to an open facility. Client $j \in C$ requires a non negative d_j units of demand, and facility $i \in F$ has a non negative opening cost f_i and can serve at most u_i units of demand. We focus on the soft capacitated problem in which multiple facilities can be built at a single location. The goal is to find a set $S \subseteq F$ of open facilities, a feasible assignment $\sigma : C \rightarrow S$ of clients in C to open facilities in S and a connected subgraph $R = (V_R, E_R)$ spanning S, so as to minimize the total cost:

$$Cost\,(S, \sigma, R) = M \cdot ST\,(R) + T\,(\sigma) + C_f\,(S)$$

where:

1) M is a rate update parameter,
2) $ST\,(R) = \sum_{i,j \in E_R} c_{i,j}$ is the edge cost of the graph R[1],
3) $T\,(\sigma) = \sum_{j \in C} d_j \cdot c_{j,\sigma(j)}$ is the cost of serving each client j from its assigned open facility $i \in S$, and
4) $C_f\,(S) = \sum_{i \in S} y_i \cdot f_i$ is the opening cost of the facilities, where y_i the number of facilities opened at location $i \in S$.

The capacity constraint implies that $\sum_{j \in C | \sigma(j) = i} d_j \leq u_i \cdot y_i$ for each i. Note that in the text we abuse notations and use the term facility f_i to describe the set of facilities opened in location i.

Informally, we aims at reducing the total cost which contains the update cost, the cost of opening facilities, and the cost of serving data from the open facilities. Note that like in real life, the model allow each facility (server) to serve only a fixed number u_i of clients.

We start with introducing several notations and definitions that are useful for the analysis. Given a set $S \subseteq F$ of open facilities and an assignment σ:

- $L_i(\sigma)$ is the set of clients assigned to facility $i \in S$ under assignment σ:
$$L_i(\sigma) = \{j \in C | \sigma(j) = i\}.$$
- $D_i(\sigma)$ is the total amount of demand assigned to facility $i \in S$ under assignment σ:
$$D_i(\sigma) = \sum_{j \in L_i(\sigma)} d_j.$$
- D is the total demand of clients in the network:
$$D = \sum_{j \in C} d_j.$$

We say that σ is λ-loaded if it satisfies:

$$D_i(\sigma) \geq \lambda \text{ for all } i \in S.$$

The λ-loaded facility location problem is a special type of the demand-weighted facility location problems in which we require that each facility will be assigned at least λ units of demand. This variant captures the idea that opening a facility is economically justified only when it serves at least a certain amount of demand (and this constraint may even be more natural than facility costs in some settings). This problem was introduced simultaneously by Karger and Minkoff [5] and by Guha et al. [4]. Both papers presented a $(\frac{1+\alpha}{1-\alpha}\beta, \alpha)$ bicriteria approximation, for any $\alpha < 1$, where β is the approximation-ratio of the metric facility location problem. Throughout the analysis of our problem, we use their bicriteria approximation. Note that despite using this bicriteria approximation algorithm, we present a deterministic constant approximation to the problem.

[1] The minimum cost connected subgraph spanning S is a minimum Steiner tree.

IV. A Constant Approximation Algorithm

In this section we present our deterministic constant approximation algorithm for the soft capacitated connected facility location problem. Basically, it is an algorithm with a guaranteed performance which is not too far from the best possible solution for the problem. In the next section we show that the actual performance of this algorithm over realistic data is indeed better that existing algorithms.

Throughout this section we assume that the total demand of clients is larger than the update rate parameter:

$$M \leq D.$$

If this is not the case, the optimal solution is trivial: Open only one facility (with minimum cost for satisfying total clients demands). It is straight forward to show that this yields a 2 approximation algorithm.

A. Soft-ConFL Algorithm

In this section, we present the Soft-ConFL algorithm. Given any ρ-approximation algorithm for the uncapacitated facility location problem and any φ-approximation algorithm for the minimum Steiner tree problem, our Soft-ConFL algorithm yields a $(24\rho + 2\varphi + 48\rho\varphi)$-approximation algorithm for the soft capacitated connected facility location problem.

Given an original soft capacitated connected facility location problem, we construct a related facility location problem by modifying both the opening cost of facilities and the distance function. First, we add a cost λ_i to each facility $i \in F$. This cost is defined as twice the minimum cost of satisfying M units of demand from facility i. More formally, let $j_1, j_2, \cdots j_n$ be the clients, ordered in increasing distance from facility i. Let k be the minimum number such that

$$d_{j_1} + d_{j_2} + \cdots + d_{j_k} \geq M.$$

For simplicity let us assume this sum is exactly equal to M (we can always split client k into two smaller demands). The opening cost of facility i is set to $f'_i = f_i + \lambda_i$, where λ_i is defined to be

$$\lambda_i = 2(c_{ij_1} d_{j_1} + c_{ij_2} d_{j_2} + \cdots + c_{ij_k} d_{j_k}).$$

Second, We modify the distance function to be:

$$c'_{i,j} = c_{i,j} + max(\frac{f'_i}{u_i}, \frac{f'_j}{u_j}),$$

where for any $i \notin F$, we set $\dfrac{f_i}{u_i}$ to be 0.

Now we run any given ρ-approximation algorithm for the uncapacitated facility location problem on the modified problem. We modify the output of this approximation algorithm by closing any facility with $D_i(\sigma) \leq \frac{M}{2}$ and assigning every client to the nearest open facility (this may require opening additional copies in this site). We run any given φ-approximation algorithm for the minimum Steiner tree problem on the set of opened facilities S, and this is the output of the Soft-ConFL algorithm.

Our main result is the following theorem that states that *Soft-ConFL Algorithm* indeed yields a constant approximation solutions.

Theorem 1 *Given a ρ-approximation algorithm for the uncapacitated facility location problem and a φ-approximation algorithm for the minimum Steiner tree problem, one can find a solution to the soft capacitated connected facility location problem, with a cost of at most $(24\rho + 2\varphi + 48\rho\varphi)$ times the optimum cost.*

In order to prove the theorem we prove the following four lemmas. The first lemma is a technical lemma stating that any solution can be converted into an M-loaded solution without increasing the cost too much.

Lemma 1 *For any given feasible solution to the soft capacitated connected facility location problem with update parameter $M \geq D$ $SOL = (S, \sigma, R)$, one can find a feasible $M - loaded$ solution, $SOL' = (S', \sigma', R')$, such that*

$$Cost(SOL') \leq 4 \cdot Cost(SOL).$$

Proof

We are given a feasible solution to the soft capacitated connected facility location problem $SOL = (S, \sigma, R)$ with update parameter $M \geq D$. We can assume that $R = \{V_R, E_R\}$ is a tree, otherwise one can find a lower cost sub-tree (with no cycles) of R which spans V_R. The following procedure iteratively converts the solution SOL into an $M - loaded$ solution SOL'.

Throughout the execution of the procedure, we differentiate between two types of facilities: *M-loaded facilities* which are open and have at least M units of demand assigned to them, and *Unloaded facilities* which are open but have less than M units of demand assigned to them. $S_{M-loaded}$ will denote the set of opened facilities which are M-loaded for the current solution; that is,

$$S_{M-loaded} = \{i \in S; D_i(\sigma) \geq M\}.$$

We also differentiate between two types of clients: *Co-op clients* which are assigned to *M-loaded facilities* and *Free clients* which are assigned to *Unloaded facilities*. We define $TotalFree(n, \sigma)$ to be the total demand assigned to unloaded facilities under assignment σ in sub-tree n; that is,

$$TotalFree(n, \sigma) = \sum_{i \in n \cap S_{M-loaded}} D_i.$$

Claim: Given a tree rooted at n with $M \leq TotalFree(n, \sigma)$, one can find a sub-tree n' of n with $M \leq TotalFree(n', \sigma) \leq 3M$, such that n' has no subtree with M or more free demand.

The proof of the Claim is omitted due to lack of space, it can be found in [22]. The main procedure, $FindMLoadedSolution(S, \sigma, R)$ is described below. It gets as an input a feasible solution $SOL = (S, \sigma, R)$ to the soft capacitated connected facility and returns an M-loaded solution. As mentioned before, we assume $M \leq D$, otherwise there exist no feasible M-loaded solution. The procedure first convert R into a binary tree rooted at r (the general tree is converted into a binary tree by introducing dummy nodes). We set S' to hold the set of opened M-loaded facilities. Throughout the execution we add new M-loaded facilities to S'. In each iteration of the algorithm we look for a sub-tree with $M \leq TotalFree(n', \sigma) \leq 3M$, according to our claim.

Algorithm IV.1: FINDMLOADEDSOLUTION(S, σ, R)

comment: Returns an M-loaded solution
$T \leftarrow Binary\ spanning\ tree\ of\ R\ rooted\ at\ r$
$S' \leftarrow S_{M-loaded}$
$\sigma \leftarrow \sigma'$
while FINDMINIMALFREETREE$(r, \sigma') \neq NULL$
do $\begin{cases} n \leftarrow \text{FINDMINIMALFREETREE}(r, \sigma') \\ l \leftarrow the\ minimal\ cost\ facility\ in\ n\ which\ is\ not \\ M_l oaded \\ S' = S' \cup l \\ assign\ free\ demands\ to\ l \\ update\ \sigma'\ with\ the\ new\ assignment \end{cases}$
$assign\ remaining\ free\ demands\ to\ the\ minimal\ cost$
$facility\ in\ S'$
return (S', σ', R')

The algorithm iteratively converts any given solution into an *M-loaded* solution without increasing the total cost by more than a factor of four. Throughout the execution of the algorithm, it maintains a feasible solution where the demand is assigned to open facilities. The following properties hold:

(P1) $ST(R') \leq ST(R)$
(P2) $C_f(S') \leq C_f(S)$
(P3) $T(\sigma') \leq T(\sigma) + 3M \cdot ST(R)$

These properties certainly hold when the algorithm starts. Furthermore, the fact that they hold when the algorithm stops, proves Lemma 1. Property (P1) clearly holds since $S' \subseteq S$. Property (P2) is maintained by the algorithm in each iteration, since we assign clients' demand to the minimal cost facility in each subtree. To show that property (P3) is maintained, first observe that in each iterations any *M-loaded* subtree (with no free demands) has no change in its facilities' assignments. Each iteration reassign free demand units to an open facility which is not *M-loaded* or part of an *M-loaded* subtree. Thus, the procedure will modify any subtree at most one time. Second, in each iteration (including the final iteration) we reassign maximum $3M$ demand units in subtree n therefore, we increase the cost of assigning each client by at most $3M$ times the cost of subtree n. Since we modify each subtree only once, we increase the total assignment cost by at most $3M \cdot ST(R)$. □

The second lemma is used to show that the *Soft-ConFL Algorithm* yields a constant approximation $\frac{M}{2}$-loaded solution using any ρ-approximation algorithm for the uncapacitated facility location problem.

Lemma 2 *Given a ρ-approximation algorithm for the uncapacitated facility location problem and and a parameter λ, one can find a $\frac{\lambda}{2} - loaded$ solution of cost at most 3ρ times the optimum $\lambda - loaded$ facility location cost.*

The proof is omitted due to lack of space, it can be found in [22]. The next lemma is used to show that the cost of connecting a set of facilities which are M-loaded is bounded.

Lemma 3 *Let $SOL' = (S', \sigma', R')$ be any solution for the soft capacitated connected facility location problem and let $SOL = (S, \sigma, R)$ be any $\lambda - loaded$ solution, then*

$$ST(R) \leq \frac{1}{\lambda}(T(\sigma) + T(\sigma')) + ST(R').$$

Proof

The proof of this lemma is quite simple. We show that for each facility i in S we can find a path of cost p_i to a facility in S', such that:

$$\sum_{i \in S} p_i \leq \frac{1}{\lambda}(T(\sigma) + T(\sigma')).$$

We choose p_i to be

$$p_i = min_{j \in L_i(\sigma)}(c_{i,j} + c_{j,\sigma'(j)}).$$

By definition:

$$T(\sigma) = \sum_{i \in S} \sum_{j \in L_i(\sigma)} d_j \cdot c_{i,j}$$

and

$$T(\sigma') = \sum_{j \in C} d_j \cdot c_{j,\sigma'(j)} = \sum_{i \in S} \sum_{j \in L_i(\sigma)} d_j \cdot c_{j,\sigma'(j)}$$

Therefore

$$\begin{aligned} T(\sigma) + T(\sigma') = \sum_{i \in S} \sum_{j \in L_i(\sigma)} d_j \cdot (c_{i,j} + c_{j,\sigma'(j)}) \geq \\ \sum_{i \in S} \sum_{j \in L_i(\sigma)} d_j \cdot p_i \end{aligned}$$

since SOL is $\lambda - loaded$, we have

$$\begin{gathered} T(\sigma) + T(\sigma') \geq \sum_{i \in S} \lambda \cdot p_i \\ \Downarrow \\ \sum_{i \in S} p_i \leq \frac{1}{\lambda}(T(\sigma) + T(\sigma')) \end{gathered}$$

The union of the Steiner tree which spans S' and all paths between S to S' spans S. Thus:

$$\begin{gathered} ST(R') + \sum_{i \in S} p_i \geq ST(R) \\ \Downarrow \\ ST(R') + \frac{1}{\lambda}(T(\sigma) + T(\sigma')) \geq ST(R) \end{gathered}$$

□

The last lemma is used to show that *Soft-ConFL Algorithm* yields a soft-capacitated solution with constant approximation.

Lemma 4 *Given a ρ-approximation algorithm for the $\lambda - loaded$ uncapacitated facility location problem, one can get a 2ρ-approximation for the $\lambda - loaded$ soft capacitated facility location version.*

Proof

Given an instance I of the $\lambda - loaded$ soft capacitated facility location, we construct an instance I' of the $\lambda - loaded$ uncapacitated facility location with the following modified distance function

$$c'_{i,j} = c_{i,j} + max(\frac{f_i}{u_i}, \frac{f_j}{u_j}),$$

where for any $i \notin F$, we set $\frac{f_i}{u_i}$ to be 0. It can be easily verified that the modified distance function satisfy the metric conditions. The facility costs and client demands in the new instance are the same as the facility costs f_i and client demands d_j in the soft capacitated instance.

First we show that given a solution $SOL = (S, \sigma)$ to I with assignment cost $T(\sigma)$ and facility cost $C_f(S)$, there exist a solution for I' with an assignment cost of at most $T(\sigma) + C_f(S)$ and a facility cost of at most $C_f(S)$. Let y_i be the number of facilities built at i in SOL. Let $x_{i,j}$ be an indicator variable that is 1 iff client j is assigned to facility i under assignment σ. $T(\sigma) = \sum_{i,j} x_{i,j} \cdot c_{i,j} \cdot d_j$ and $C_f(S) = \sum_i y_i \cdot f_i$. The capacity constraint implies that $\sum_j x_{i,j} \cdot d_j \leq u_i \cdot y_i$. We construct a feasible solution $SOL' = (S', \sigma')$ of I' as follows: The assignments of nodes to facilities is the same as in σ (which assures that it would satisfy the $\lambda - loaded$ condition). A facility is built at i iff at least one facility is built at i in S. Clearly the opening cost of the facilities $C_f(S')$ is at most $C_f(S)$ and the assignment cost $T(S')$

$$\begin{aligned} \sum_{i,j} x_{i,j} d_j c'_{i,j} = \sum_{i,j} x_{i,j} d_j (c_{i,j} + \frac{f_i}{u_i}) = \\ = \sum_{i,j} x_{i,j} d_j c_{i,j} + \sum_i f_i \frac{\sum_j x_{i,j} d_j}{u_i} \leq \\ \leq \sum_{i,j} x_{i,j} d_j c_{i,j} + \sum_i f_i y_i \\ = T(\sigma) + C_f(S). \end{aligned}$$

Note that this implies that $OPT(I') \leq 2 \cdot OPT(I)$.

Second we show that given a feasible solution $SOL' = (S', \sigma')$ to I', there exists a feasible solution $SOL = (S, \sigma)$ to I such that $cost(SOL) \leq cost(SOL')$. Let Y_i be an indicator variable that is 1 iff a facility is built at i in SOL'. Let $x_{i,j}$ be an indicator variable that is 1 iff client j is assigned to facility i in SOL'. The solution SOL for instance I is obtained as follows: The assignments of nodes to facilities is the same as in σ' (which assures that it would satisfy the $\lambda - loaded$ condition). The number of facilities built at i is

$$y_i = \lceil \frac{\sum_j x_{i,j} \cdot d_j}{u_i} \rceil.$$

Thus, $y_i \leq \frac{\sum_j x_{i,j} \cdot d_j}{u_i} + Y_i$. The cost of SOL' is

$$\sum_{i,j} x_{i,j} d_j c'_{i,j} + \sum_i f_i Y_i = \sum_{i,j} x_{i,j} d_j (c_{i,j} + \frac{f_i}{u_i}) + \sum_i f_i Y_i$$
$$= \sum_{i,j} x_{i,j} d_j c_{i,j} + \sum_i f_i (\frac{\sum_j x_{i,j} \cdot d_j}{u_i} + Y_i)$$
$$\geq \sum_{i,j} x_{i,j} d_j c_{i,j} + \sum_i f_i y_i = cost(SOL).$$

Since we have a ρ approximation algorithm for the $\lambda - loaded$ uncapacitated facility location problem, we can obtain a solution for I' with a cost of at most $\rho \cdot OPT(I')$, and since we proved $OPT(I') \leq 2 \cdot OPT(I)$ we get $\rho OPT(I') \leq 2\rho OPT(I)$. Now, we can obtain a solution to I of cost at most $2\rho OPT(I)$, yielding a 2ρ approximation for the $\lambda - loaded$ soft capacitated facility location problem. □

Proof of Theorem 1

Let $SOL^* = (S^*, \sigma^*, R^*)$ be the optimal solution of the soft capacitated connected facility location problem. Lemma 1 implies that there exists a M-loaded solution $SOL' = (S', \sigma', R')$ such that:

$$Cost(SOL') \leq 4 \cdot Cost(SOL^*).$$

Let $\widehat{SOL} = (\widehat{S}, \widehat{\sigma}, \widehat{R})$ be the optimal solution of the equivalent $M - loaded$ soft capacitated facility location problem which minimizes the total assignment cost and opening cost of facilities. Since $\widehat{SOL}$ is optimal solution of all feasible $M - loaded$ solution (including SOL') we get:

$$T(\widehat{\sigma}) + C_f(\widehat{S}) \leq T(\sigma') + C_f(S') \leq Cost(SOL') \leq 4 \cdot Cost(SOL^*).$$

By Lemma 2 and Lemma 4 we get that given a $\rho - approximation$ algorithm for the uncapacitated facility location problem we can find a $\frac{M}{2} - loaded$ solution $SOL = (S, \sigma, R)$ that satisfies:

$$T(\sigma) + C_f(S) \leq 6\rho(T(\widehat{\sigma}) + C_f(\widehat{S})) \leq 24\rho(Cost(SOL^*)).$$

By Lemma 3 and since SOL is a $\frac{M}{2} - loaded$ solution, we get:

$$ST(R) \leq \frac{2}{M}(T(\sigma) + T(sigma^*)) + ST(R^*)$$
$$\Downarrow$$
$$M \cdot ST(R) \leq 2(T(\sigma) + T(\sigma^*)) + M \cdot ST(R^*) \leq 2 \cdot Cost(SOL^*) + 2 \cdot T(\sigma) \leq (2 + 48\rho)Cost(SOL^*)$$

Since the Minimum Steiner Tree Problem is APX-complete (Bern and Plassmann [3]), we use a $\varphi - approximation$ algorithm (Robins and Zelikovsky [4]), thus:

$$M \cdot ST(R) \leq (2 + 48\rho)\varphi \cdot Cost(S^*)$$

Combining the above, we obtain the following inequality:

$$Cost(SOL) = T(\sigma) + C_f(S) + M \cdot ST(R) \leq (24\rho + 2\varphi + 48\rho\varphi)Cost(SOL^*),$$

Fig. 1. Google data centers world wide

which proves Theorem 1. □

V. Experimental Results

In this section we evaluate the performance of our algorithm through extensive simulation experiments on realistic scenarios. Unless stated otherwise, we assume unified demand for each node and unified cost for each data center. In our simulations, we assume that the network distances between the data centers are relative to the geographic distance between them.

Google's data infrastructure is massive and spread across the world, according to Data Center Knowledge (DCK) in 2008, there were 36 data centers all together[2] (see Figure 1) - 19 in the U.S., 12 in Europe, 3 in Asia, one in Russia, and one in South America[3]. Not all of the locations are dedicated Google data centers, since it is claimed that they sometimes lease space in other companies' data centers[4]. In our simulations we used the map of locations which was constructed by Pingdom and Data Center Knowledge.

In order to evaluate the performance of the *Soft-ConFL Algorithm* we compared our results to two alternative algorithms:

- **Greedy Algorithm:** Iteratively the algorithm picks one facility f_i which minimizes the total cost and add it to set S. During the iterations, the algorithm holds the set with the minimum cost and returns it.
- **Soft Capacitated Facility Location Algorithm (Soft-FLP):** A local search iterative heuristic which starts with any feasible solution, and improves the quality of the solution iteratively. At each step, it considers only local operations to improve the cost of the solution. A solution is called a local minima if there is no local operations which improves the cost. The local search procedure that we consider permits *adding*, *dropping* and *swapping* a facility. Arya et al. [15] showed that such a procedure gives a 3 approximation algorithm in the case where update costs are not considered.

For a proper comparison of our algorithm performance, we have used the same local search algorithm from [15] as the uncapacitated facility location algorithm which is used in Soft-ConFL.

[2]http://techcrunch.com/2008/04/11/where-are-all-the-google-data-centers/
[3]http://www.datacenterknowledge.com/archives/2008/03/27/google-data-center-faq/
[4]http://royal.pingdom.com/2008/04/11/map-of-all-google-data-center-locations/

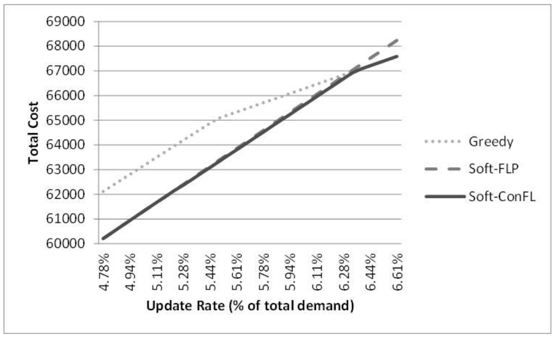

Fig. 2. Soft-ConFL algorithm vs. Greedy algorithm and Soft-FLP (facility cost of 5,000)

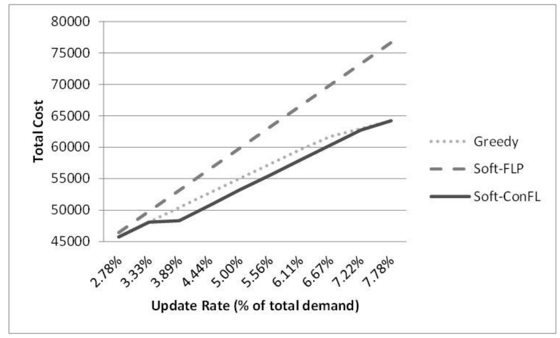

Fig. 3. Soft-ConFL algorithm vs. Greedy algorithm and Soft-FLP (facility cost of 3,000)

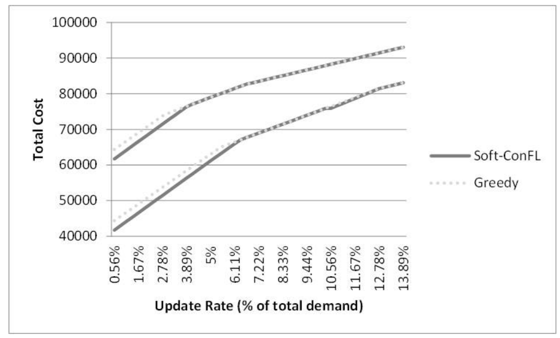

Fig. 4. Soft-ConFL algorithm vs. Greedy algorithm (facility cost of 5,000 and 10,000)

Figure 2 depicts the results of the Soft-ConFL algorithm, the Greedy algorithm and the Soft-FLP. We set a unified cost of 5,000 units for each location. Note that our cost function combines facility and network cost and the total cost of the minimum spanning tree of the data centers network is 22,000 units. We compared the algorithm performance at different update rates. The update rate variable is presented as percentage of the total demand[5]. One can observe that the higher the update rate, the performance of the Soft-FLP algorithm is worse, because it ignores the update cost of the facilities. The greedy algorithm yields inferior results for low update rates ($< 6\%$) because it opens a large set of facilities. However, for larger update rates ($> 7\%$) where it opens a small set of facilities it yields the same results as Soft-ConFL. Note that update rate of 6.4% is a turning point where the three algorithm yield similar results, the greedy and Soft-ConFL unify, and Soft-FLP starts to yields significant inferior results.

Figure 3 is similar to Figure 2 but the unified facility cost was set to a lower value of 3,000. As in the previous case, the local search algorithm (dashed line), which neglects the cost of keeping the replicas across the network up to date, yields sub-optimal results. We can see that the algorithms that considers this factor (the greedy algorithm and Soft-ConFL) can lead to better solutions. Note that for low update rates ($< 3.5\%$) both the greedy algorithm and Soft-ConFL yield the same results, for a certain range of update rates ($3.5\% < M < 7.5\%$) the Soft-ConFL yields better results and for higher update rates the graphs of both algorithm reunites.

The Soft-FLP algorithm yields suboptimal results for high update rates, thus in Figure 4 we focus on comparing the greedy algorithm and Soft-ConFL. We present two experiments with different facility costs (5,000 and 10,000). Note that in both cases we get almost identical pattern where the Soft-ConFL yields better results for lower update rates and for higher update rates they yield identical result. Interestingly Soft-ConFL yields better results for $9\% < M < 12\%$ and facility cost of 5,000.

To get a better understanding of the impact of the facility opening cost, we evaluated the performance of the algorithm with different facility opening costs (1,000-25,000 units of cost per facility). Figure 5 depicts the total cost of the Soft-ConFL algorithm and the Greedy algorithm as a function of facility opening cost. We compared the algorithm performance at different update rate variables which are presented as percentage of the total demand ($1\% - 4\%$). The greedy algorithm yields inferior results which cost up to 7% more than the Soft-ConFL Algorithm for all range of facility opening cost.

In Figure 6, we study in more details the results of Soft-ConFL for different update rates. As expected, the higher the cost to update the facilities the less facilities have been chosen (Starting with 10 facilities for 2.8% and ending with 3 facilities for 13.9%). Note that the decrease in the set of

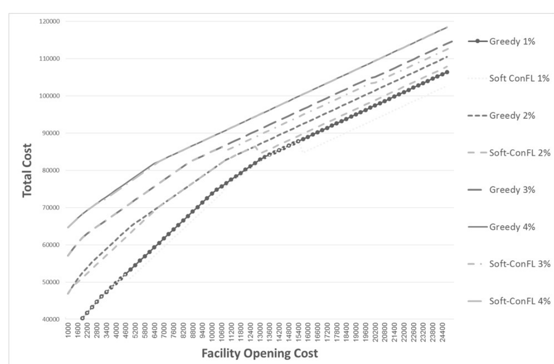

Fig. 5. Total Cost as a function of Facility Opening Cost

[5]Note that the demand in our problem is in fact demand per time unit, and the ratio is a dimensionless quantity.

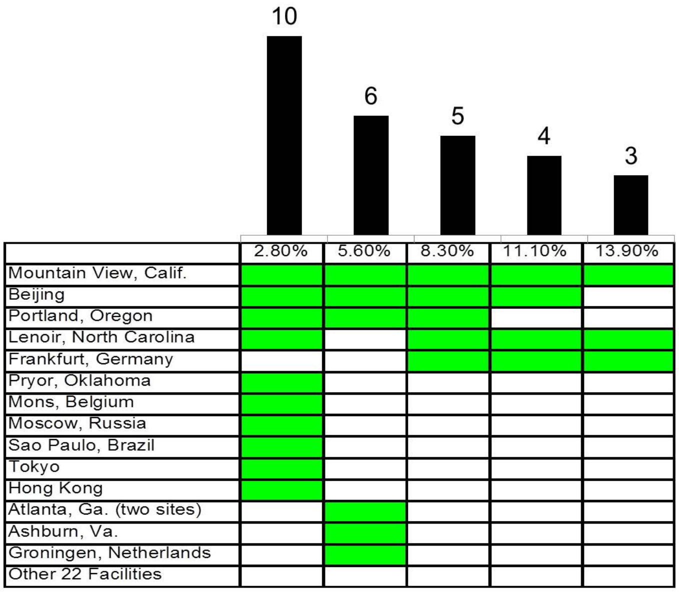

Fig. 6. Soft-ConFL results for different update rates (facility cost of 1,000)

opened facilities was not done by picking a subset of an existing set, but the algorithm introduced new facilities and closed existing facilities. Some interesting facilities are: the Mountain View facility that was chosen in any iteration, the Lenoir facility that was chosen at the first iteration (2.8%), was not chosen in the second iteration but was re-chosen again in the other iterations and the Frankfurt facility that was added to the open facility sets only from the third (8.3%) iteration.

VI. Conclusion

In this paper we studied optimal placement of of applications and their data over available cloud infrastructure where the data update cost is also considered. This is an important aspect of cloud management and one of the enabling factors that allows providing global service at a low cost. We presented the first deterministic constant approximation algorithm for the problem, and showed that in addition to a guarantee bound on the worst case performance of the algorithm, it also performs better than any exiting algorithm over a large set of realistic cloud data. Our new algorithm is robust and performs well in various scenarios. The main reason behind this is that unlike the Greedy approach, our algorithm can select completely different locations for the data when the parameters (for example the update rate) change.

Future research in this field may consider hard capacity constraints where a facility can be opened at most once in each location.Another important aspect is the online version in which data can be moved and the number of replicas can be dynamically adjusted according to the actual update rate or changes in the demand for the service.

References

[1] S. Agarwal, J. Dunagan, N. Jain, S. Saroiu, A. Wolman, and H. Bhogan, *Volley: Automated Data Placement for Geo-Distributed Cloud Sevices*, In Proceedings of the 7th USENIX conference on Networked systems design and implementation (NSDI), pages 2-2, Berkeley, CA, USA, 2010.

[2] X. Meng, Y. Pappas, and L. Zhang, *Improving the scalability of data center networks with traffic-aware virtual machine placement*, In Proceedings of the 29th conference on Information communications (INFOCOM), 2010.

[3] S. Borst, V. Gupta, and A. Walid, *Distributed Caching Algorithms for Content Distribution Networks*, in Proc. of IEEE INFOCOM, 2010.

[4] S. Guha, A. Meyerson, K. Munagala, *Hierarchical placement and network design problems*, In Proceeding of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS), pages 603-612, 2000.

[5] D. R. Karger and M. Minkoff, *Building Steiner trees with incomplete global knowledge*, In Proceeding of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS), pages 613-623, 2000.

[6] M. Sviridenko, *An improved approximation algorithm for the metric uncapacitated facility location problem*, In Cook, W.J., Schulz, A.S. (eds.) IPCO 2002. LNCS, vol. 2337, pp. 240-257. Springer, Heidelberg (2002).

[7] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi and B. Yener, *Provisioning a virtual private networks: A network design problem for multicommodity flow*, In Proceeding of the 33rd Annual ACM Symposium on Theory of Computing (STOC), pages 389-398, 2001.

[8] F. Eisenbrand, F. Grandoni, T. RothvoB and G. Schafer, *Approximating connected facility location problems via random facility sampling and core detouring*, In Proceeding of the nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 1174-1183, 2008.

[9] F. Eisenbrand, F. Grandoni, T. Rothvoβ and G. Schafer, *Connected facility location via random facility sampling and core detouring*, Journal of Computer and System Sciences 76: 709-726, 2010.

[10] L.M. Vaquero, L. Rodero-Merino, J. Caceres and M. Lindner, *A Break in the Clouds: Towards a Cloud Definition*, SIGCOMM Computer Communication Review, 2009. 39(1): p. 50-55.

[11] P.A. Laplante, Z. Jia and J. Voas, *What's in a name? Distinguishing between SaaS and SOA*, IT Professional, 2008. 10(3): p. 46-50.

[12] T. Kwok and A. Mohindra, *Resource calculations with constraints, and placement of tenants and instances for multi-tenant SaaS applications*, in Sixth International Conference on Service-Oriented Computing. 2008, Springer: Sydney, Australia. p. 633-648.

[13] Z.I.M. Yusoh and T. Maolin, *A penalty-based genetic algorithm for the composite SaaS placement problem in the Cloud*, Evolutionary Computation (CEC), 2010 IEEE Congress on , vol., no., p.1-8

[14] D. Shmoys, E. Tardos and K. Aardal, *Approximation algorithms for facility location problems*, In 29th ACM Symposium on Theory of Computing, pages 265-274, 1997.

[15] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala and V. Pandit, *Local search heuristics for k-median and facility location problems*, In Proceedings of 33rd ACM Symposium on Theory of Computing, 2001.

[16] M. Mahdian, Y. Ye and J. Zhang, *A 2-approximation algorithm for the soft-capacitated facility location problem*, In Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization, pages 129-140, 2003.

[17] M. Leitner, G. R. Raidl, *Variable neighborhood search for a prize collecting capacity constrained connected facility location problem*, In Proceedings of the 2008 International Symposium on Applications and the Internet, IEEE Computer Society, pages 233-236, 2008.

[18] M. Leitner, G. R. Raidl, *A Lagrangian decomposition based heuristic for capacitated connected facility location* In Proceedings of the 8th Metaheuristic International Conference (MIC), Hamburg, Germany, 2009.

[19] M. Leitner, G. R. Raidl, *Combining Lagrangian decomposition with very large scale neighborhoood search for capacitated connected facility location*, In: Post-Conference Book of the Eight Metaheuristics International Conference (MIC), 2010.

[20] G. Peng, *CDN: Content distribution Network*, Technical Report TR-125, Experimental Computer Systems Lab, Department of Computer Science, State University of New York, Stony Brook, 2004.

[21] C. D. Cranor, M. Green, C. Kalmanek, D. Shur, S. Sibal, J. E. Van der Merwe, C. J. Sreenan, *Enhanced Streaming Services in a Content Distribution Network*, IEEE Internet Computing, vol. 5, no. 4, 2001.

[22] A. Rappaport, *Approximation Algorithms for Soft-Capacitated Connected Facility Location Problems*, M.Sc. Thesis, Computer Science Department, Technion Israel Institute of Technology, 2013, also avaialble at http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2013/MSC/MSC-2013-10.pdf .