Reallocating Multiple Facilities on the Line*

Dimitris Fotakis[1], Loukas Kavouras[1], Panagiotis Kostopanagiotis[1], Philip Lazos[2], Stratis Skoulakis[1], and Nikolas Zarifis[1]

[1]National Technical University of Athens
[2]Sapienza University of Rome

Abstract

We study the multistage *K-facility reallocation problem* on the real line, where we maintain K facility locations over T stages, based on the stage-dependent locations of n agents. Each agent is connected to the nearest facility at each stage, and the facilities may move from one stage to another, to accommodate different agent locations. The objective is to minimize the connection cost of the agents plus the total moving cost of the facilities, over all stages. *K-facility reallocation* was introduced by de Keijzer and Wojtczak [10], where they mostly focused on the special case of a single facility. Using an LP-based approach, we present a polynomial time algorithm that computes the optimal solution for any number of facilities. We also consider online *K-facility reallocation*, where the algorithm becomes aware of agent locations in a stage-by-stage fashion. By exploiting an interesting connection to the classical *K-server problem*, we present a constant-competitive algorithm for $K = 2$ facilities.

1 Introduction

Facility Location is a classical problem that has been widely studied in both combinatorial optimization and operations research, due to its many practical applications. It provides a simple and natural model for industrial planning, network design, machine learning, data clustering and computer vision Drezner and Hamacher [12], Lazic [19], Caragiannis et al. [7], Betzler et al. [5]. In its basic form, K-Facility Location instances are defined by the locations of n agents in a metric space. The goal is to find K facility locations so as to minimize the sum of distances of the agents to their nearest facility.

In many natural location and network design settings, agent locations are not known in advance. Motivated by this fact, Meyerson [21] introduced online facility location problems, where agents arrive one-by-one and must be *irrevocably* assigned to a facility upon arrival. Moreover, the fast increasing volume of available data and the requirement for responsive services has led to new, *online* clustering algorithms Liberty et al. [20], balancing the quality of the clusters with their rate of change over time. In practical settings related to online data clustering, new data points arrive, and the decision of clustering some data points together should not be regarded as irrevocable (see e.g., Fotakis [15] and the references therein).

More recently, understanding the dynamics of temporally evolving social or infrastructure networks has been the central question in many applied areas such as viral marketing, urban planning etc. *Dynamic facility location* proposed by Eisenstat et al. [13] has been a new tool to analyze temporal aspects of such networks. In this time dependent variant of facility location, agents may change their location over time and we look for the best tradeoff between the optimal connections of agents to facilities and the stability of solutions between

*Loukas Kavouras is partially supported by a scholarship from the State Scholarships Foundation, granted by the action "Scholarships Grant Programme for second cycle graduate studies", which is co-financed by Greece and the European Union (European Social Fund- ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning 2014- 2020". Stratis Skoulakis is partially supported by a scholarship of Onnasis Foundation. Philip Lazos is supported by the ERC Advanced Grant 788893 (AMDROMA). The names of the authors are in alphabetical order.

consecutive timesteps. The stability of the solutions is modeled by introducing an additional moving cost (or switching cost), which has a different definition depending on the particular setting.

Model and Motivation. In this work, we study the multistage *K-facility reallocation problem* on the real line, introduced by de Keijzer and Wojtczak [10]. In *K-facility reallocation*, K facilities are initially located at $(x_1^0, \ldots, x_K^0)$ on the real line. Facilities are meant to serve n agents for the next T days. At each day, each agent connects to the facility closest to its location and incurs a connection cost equal to this distance. The locations of the agents may change every day, thus we have to move facilities accordingly in order to reduce the connection cost. Naturally, moving a facility is not for free, but comes with the price of the distance that the facility was moved. Our goal is to specify the exact positions of the facilities at each day so that the total connection cost plus the total moving cost is minimized over all T days. In the online version of the problem, the positions of the agents at each stage t are revealed only after determining the locations of the facilities at stage $t-1$.

For a motivating example, consider a company willing to advertise its products. To this end, it organizes K advertising campaigns at different locations of a large city for the next T days. Based on planned events, weather forecasts, etc., the company estimates a population distribution over the locations of the city for each day. Then, the company decides to compute the best possible campaign reallocation with K campaigns over all days (see also [10] for more examples).

de Keijzer and Wojtczak [10] fully characterized the optimal offline and online algorithms for the special case of a single facility and presented a dynamic programming algorithm for $K \geq 1$ facilities with running time exponential in K. Despite the practical significance and the interesting theoretical properties of K-facility reallocation, its computational complexity and its competitive ratio (for the online variant) are hardly understood.

Contribution. In this work, we resolve the computational complexity of *K-facility reallocation* on the real line and take a first step towards a full understanding of the competitive ratio for the online variant. More specifically, in Section 3, we present an optimal algorithm with running time polynomial in the combinatorial parameters of *K-facility reallocation* (i.e., n, T and K). This substantially improves on the complexity of the algorithm, presented in [10], that is exponential in K. Our algorithm solves a Linear Programming relaxation and then *rounds* the *fractional solution* to determine the positions of the facilities. The main technical contribution is showing that a simple rounding scheme yields an integral solution that has the exact same cost as the *fractional one*.

Our second main result concerns the *online version* of the problem with $K = 2$ facilities. We start with the observation that online *K-facility reallocation problem* with $K \geq 2$ facilities is a natural and interesting generalization of the classical *K-server problem*, which has been a driving force in the development of online algorithms for decades. The key difference is that, in the *K-server problem*, there is a single agent that changes her location at each stage and a single facility has to be relocated to this new location at each stage. Therefore, the total connection cost is by definition 0, and we seek to minimize the total moving cost.

From a technical viewpoint, the *K-facility reallocation problem* poses a new challenge, since it is *much* harder to track the movements of the optimal algorithm as the agents keep coming. It is not evident at all whether techniques and ideas from the *K-server problem* can be applied to the *K-facility reallocation* problem, especially for more general metric spaces. As a first step towards this direction, we design a constant-competitive algorithm, when $K = 2$. Our algorithm appears in Section 4 and is inspired by the *double coverage algorithm* proposed for the *K-server problem* Koutsoupias [18].

Related Work. We can cast the *K-facility reallocation problem* as a clustering problem on a temporally evolving metric. From this point of view, *K-facility reallocation problem* is a dynamic *K-median* problem. A closely related problem is the *dynamic facility location problem*, Eisenstat et al. [13], An et al. [1]. Other examples in this setting are the *dynamic sum radii clustering* Blanchard and Schabanel [6] and multi-stage optimization problems on matroids and graphs Gupta et al. [17].

In Friggstad and Salavatipour [16], a mobile facility location problem was introduced, which can be seen as a one stage version of our problem. They showed that even this version of the problem is NP-hard in general metric spaces using an approximation preserving reduction to *K-median problem*.

Online facility location problems and variants have been extensively studied in the literature, see Fotakis

[15] for a survey. Divéki and Imreh [11] studied an online model, where facilities can be moved with zero cost. As we have mentioned before, the online variant of the *K-facility reallocation problem* is a generalization of the *K-server problem*, which is one of the most natural online problems. Koutsoupias [18] showed a $(2K-1)$-competitive algorithm for the *K-server problem* for every metric space, which is also K-competitive, in case the metric is the real line Bartal and Koutsoupias [4]. Other variants of the *K-server problem* include the *(H, K)-server problem* Bansal et al. [3, 2], the *infinite server problem* Coester et al. [9] and the *K-taxi problem*Fiat et al. [14], Coester and Koutsoupias [8].

$$(1) \qquad \min \sum_{t=1}^{T} \left[\sum_{i \in C} \sum_{j \in V} d(\mathrm{Loc}(i,t), j) x_{ij}^t + \sum_{k \in F} S_k^t \right]$$

$$\sum_{j \in V} x_{ij}^t = 1 \qquad \forall i \in C, t \in \{1, T\}$$

$$x_{ij}^t \leq c_j^t \qquad \forall i \in C, j \in V, t \in \{1, T\}$$

$$c_j^t = \sum_{k \in F} f_{kj}^t \qquad \forall j \in V, t \in \{1, T\}$$

$$\sum_{j \in V} f_{kj}^t = 1 \qquad \forall k \in F, t \in \{1, T\}$$

$$S_k^t = \sum_{j,l \in V} d(j,l) S_{kjl}^t \qquad \forall k \in F, t \in \{1, T\}$$

$$\sum_{j \in V} S_{kjl}^t = f_{kl}^t \qquad \forall k \in F, l \in V, t \in \{1, T\}$$

$$\sum_{l \in V} S_{kjl}^t = f_{kj}^{t-1} \qquad \forall k \in F, j \in V, t \in \{1, T\}$$

$$x_{ij}^t, f_{kj}^t, S_{klj}^t \in \{0, 1\} \qquad \forall k \in F, j \in V, t \in \{1, T\}$$

Figure 1: Formulation of *K-facility reallocation*

2 Problem Definition and Preliminaries

Definition 1 (*K-Facility Reallocation Problem*) *We are given a tuple (x^0, C) as input. The K dimensional vector $x^0 = (x_1^0, \ldots, x_K^0)$ describes the initial positions of the facilities. The positions of the agents over time are described by $C = (C_1, \ldots, C_T)$. The position of agent i at stage t is α_i^t and $C_t = (\alpha_1^t, \ldots, \alpha_n^t)$ describes the positions of the agents at stage t.*

Definition 2 *A solution of K-Facility Reallocation Problem is a sequence $x = (x^1, \ldots, x^T)$. Each $x^t = (x_1^t, \ldots, x_K^t)$ is a K dimensional vector that gives the positions of the facilities at stage t and x_k^t is the position of facility k at stage t. The cost of the solution x is*

$$Cost(x) = \sum_{t=1}^{T} \left[\sum_{k=1}^{K} |x_k^t - x_k^{t-1}| + \sum_{i=1}^{n} \min_{1 \leq k \leq K} |\alpha_i^t - x_k^t| \right]$$

Given an instance (x^0, C) of the problem, the goal is to find a solution x that minimizes the $Cost(x)$. The term $\sum_{t=1}^{T} \sum_{k=1}^{K} |x_k^t - x_k^{t-1}|$ describes the cost for moving the facilities from place to place and we refer to

it as *moving cost*, while the term $\sum_{t=1}^{T}\sum_{i=1}^{n}\min_{1\leq k\leq K}|\alpha_i^t - x_k^t|$ describes the connection cost of the agents and we refer to it as *connection cost*.

In the online setting, we study the special case of 2-*facility reallocation problem*. We evaluate the performance of our algorithm using competitive analysis; an algorithm is c-competitive if for every request sequence, its online performance is at most c times worse (up to a small additive constant) than the optimal *offline* algorithm, which knows the entire sequence in advance.

3 Polynomial Time Algorithm

Our approach is a typical LP based algorithm that consists of two basic steps.

- **Step 1:** Expressing the *K-Facility Reallocation Problem* as an Integer Linear Program.
- **Step 2:** Solving *fractionally* the Integer Linear Program and *rounding* the fractional solution to an integral one.

3.1 Formulating the Integer Linear Program

A first difficulty in expressing the *K-Facility Reallocation Problem* as an Integer Linear Program is that the positions on the real line are infinite. We remove this obstacle with help of the following lemma proved in [10].

Lemma 3.1 *Let (x_0, C) an instance of the K-facility reallocation problem. There exists an optimal solution x^* such that for all stages $t \in \{1, T\}$ and $k \in \{1, K\}$,*

$$x_k^{*t} \in C_1 \cup \ldots \cup C_T \cup x^0$$

According to Lemma 3.1, there exists an optimal solution that locates the facilities only at positions where either an agent has appeared or a facility was initially lying. Lemma 3.1 provides an exhaustive search algorithm for the problem and is also the basis for the *Dynamic Programming* approach in [10]. We use Lemma 3.1 to formulate our Integer Linear Program.

The set of positions $Pos = C_1 \cup \ldots \cup C_T \cup x^0$ can be represented equivalently by a path $P = (V, E)$. In this path, the j-th node corresponds to the j-th leftmost position of Pos and the distance between two consecutive nodes on the path equals the distance of the respective positions on the real line. Now, the *facility reallocation problem* takes the following *discretized form*: We have a path $P = (V, E)$ that is constructed by the specific instance (x^0, C). Each facility k is initially located at a node $j \in V$ and at each stage t, each agent i is also located at a node of P. The goal is to move the facilities from node to node such that the connection cost of the agents plus the moving cost of the facilities is minimized.

To formulate this discretized version as an Integer Linear Program, we introduce some additional notation. Let $d(j, l)$ be the distance of the nodes $j, l \in V$ in P, F be the set of facilities and C be the set of agents. For each $i \in C$, $\mathrm{Loc}(i, t)$ is the node where agent i is located at stage t. We also define the following $\{0, 1\}$-indicator variables for all $t \in \{1, T\}$: $x_{ij}^t = 1$ if, at stage t, agent i connects to a facility located at node j, $f_{kj}^t = 1$ if, at stage t, facility k is located at node j, $S_{kjl}^t = 1$ if facility k was at node j at stage $t-1$ and moved to node l at stage t. Now, the problem can be formulated as the Integer Linear Program depicted in Figure1.

The first three constraints correspond to the fact that at every stage t, each agent i must be connected to a node j where at least one facility k is located. The constraint $\sum_{j\in V} f_{kj}^t = 1$ enforces each facility k to be located at exactly one node j. The constraint $S_k^t = \sum_{j,l\in V} d(j, l)S_{kjl}^t$ describes the cost for moving facility k from node j to node l. The final two constraints ensure that facility k moved from node j to node l at stage t if and only if facility k was at node j at stage $t-1$ and was at node l at stage t ($S_{kjl}^t = 1$ iff $f_{kl}^t = 1$ and $f_{kj}^{t-1} = 1$).

Algorithm 1: Algorithm for the offline case

Data: Given the initial positions $x^0 = \{x_1^0, \ldots, x_K^0\}$ of the facilities and the positions of the agents $C = \{C_1, \ldots, C_T\}$.

- Construct the path P and the Integer Linear Program (1).
- Solve the relaxation of the Integer Linear Program (1).
- *Rounding* [1]: For each stage $t \geq 1$:
  - For $m = 1, \ldots, K$, find the node j_m^t such that

$$\sum_{\ell=1}^{j_m^t - 1} c_\ell^t \leq m - 1 \leq \sum_{\ell=1}^{j_m^t} c_\ell^t.$$

  - Locate facility m at the respective position of node j_m^t on the line

$$x_m^t \leftarrow d(j, 1) + \min_{p \in C_1 \cup \ldots \cup C_T \cup x^0} p.$$

We remark that the values of f_{kj}^0 are determined by the initial positions of the facilities, which are given by the instance of the problem. The notation x_{ij}^t should not be confused with x_k^t, which is the position of facility k at stage t on the real line .

3.2 Rounding the Fractional Solution

Our algorithm, described in Algorithm 1, is a simple rounding scheme of the *optimal fractional solution* of the Integer Linear Program of Figure 1. This simple scheme produces an integral solution that has the exact same cost with an optimal fractional solution.

Theorem 3.1 *Let x denote the solution produced by Algorithm 1. Then*

$$Cost(x) = \sum_{t=1}^{T} \left[\sum_{i \in C} \sum_{j \in V} d(Loc(i,t), j) x_{ij}^t + \sum_{k \in F} S_k^t \right]$$

where x_{ij}^t, S_k^t denote the values of these variables in the optimal fractional solution of the Integer Linear Program (1).

Theorem 3.1 is the main result of this section and it implies the optimality of our algorithm. We remind that by Lemma 3.1, there is an optimal solution that locates facilities only in positions $C_1 \cup \ldots \cup C_T \cup x^0$. This solution corresponds to an integral solution of our Integer Linear Program, meaning that $Cost(x^*)$ is greater than or equal to the cost of the *optimal fractional solution*, which by Lemma 3.1 equals $Cost(x)$. We dedicate the rest of the section to prove Theorem 3.1. The proof is conducted in two steps and each step is exhibited in sections 3.3 and 3.4 respectively.

In section 3.3, we present a very simple rounding scheme in the case, where the values of the variables of the optimal fractional solution satisfy the following assumption.

Assumption 1 *Let f_{jk}^t and c_j^t be either $1/N$ or 0, for some positive integer N.*

[1] the nodes j_m^t can be equivalent calculated with the simpler criterion, j_m^t *is the most left node with* $f_{kj}^t > 0$. See also Section 3.4.

Although Assumption 1 is very restrictive and its not generally satisfied, it is the key step for proving the optimality guarantee of the rounding scheme presented in Algorithm 1. Then, in section 3.4 we use the rounding scheme of section 3.3 to prove Theorem 3.1. In the upcoming sections, $c_j^t, x_{ij}^t, f_{kj}^t, S_{kjl}^t, S_k^t$ will denote the values of these variables in the *optimal fractional* solution of the ILP (1).

3.3 Rounding Semi-Integral solutions

Throughout this section, we suppose that Assumption 1 is satisfied; f_{kj}^t and c_j^t are either $1/N$ or 0 , for some positive integer N. If the optimal fractional solution meets these requirements, then the integral solution presented in Lemma 3.2 has the same overall cost. The goal of the section is to prove Lemma 3.2.

Definition 3 *V_t^+ denotes the set of nodes of P with a positive amount of facility (c_j^t) at stage t,*

$$j \in V_t^+ \text{ if and only if } c_j^t > 0$$

We remind that since $c_j^t = 1/N$ or 0, $|V_t^+| = K \cdot N$. We also consider the nodes in $V_t^+ = \{Y_1^t, \ldots, Y_{K \cdot N}\}$ to be ordered from left to right.

Lemma 3.2 *Let Sol be the integral solution that at each stage t places the m-th facility at the $(m-1)N+1$ node of V_t^+ i.e. $Y_{(m-1)N+1}^t$. Then, Sol has the same cost as the optimal fractional solution.*

The term **m-th facility** refers to the ordering of the facilities on the real line according to their initial positions $\{x_1^0, \ldots, x_K^0\}$. The proof of Lemma 3.2 is quite technically complicated, however it is based on two intuitive observations about the optimal fractional solution.

Observation 1 *The set of nodes at each **agent** i **connects** at stage t are **consecutive** nodes of V_t^+. More precisely, there exists a set $\{Y_\ell^t, \ldots, Y_{\ell+N-1}^t\} \subseteq V_t^+$ such that*

$$\sum_{j \in V} d(Loc(i,t), j) x_{ij}^t = \frac{1}{N} \sum_{h=\ell}^{\ell+N-1} d(Loc(i,t), Y_h^t)$$

Proof: Let an agent i that at some stage t has $x_{iY_j^t}^t > 0, x_{iY_\ell^t}^t < 1/N$ and $x_{iY_h^t}^t > 0$ for some $j < \ell < h$. Assume that $\text{Loc}(i,t) \leq Y_\ell^t$ and to simplify notation consider $x_\ell = x_{iY_\ell^t}^t, x_h = x_{iY_h^t}^t$. Now, increase x_ℓ by ϵ and decrease x_h by ϵ, where $\epsilon = \min(1/N - x_\ell, x_h)$. Then, the cost of the solution is decreased by $(d(\text{Loc}(i,t), h) - d(\text{Loc}(i,t), \ell))\epsilon > 0$, thus contradicting the optimality of the solution. The same argument holds if $\text{Loc}(i,t) \geq Y_\ell^t$. The proof follows since $\sum_{j \in V} x_{ij}^t = 1$. □

Observation 2 *Under Assumption 1, the m-th facility places amount of facility $f_{mj}^t = 1/N$ from the $(m-1)N+1$ to the mN node of V_t^+ i.e. to nodes $\{Y_{(m-1)N+1}^t, \ldots, Y_{mN}^t\}$.*

Observation 2 serves in understanding the structure of the optimal fractional solution under Assumption 1. However, it will be not used in this form in the rest of the section. We use Lemma 3.3 instead, which is roughly a different wording of Observation 2 and its proof can be found in subsection A.1 of the Appendix.

Lemma 3.3 *Let S_k^t the fractional moving cost of facility k at stage t. Then*

$$\sum_{t=1}^{T} \sum_{k \in F} S_k^t = \frac{1}{N} \sum_{t=1}^{T} \sum_{j=1}^{K \cdot N} d(Y_j^{t-1}, Y_j^t)$$

Observations 1, and Lemma 3.3 (Observation 2) are the key points in proving Lemma 3.2.

Definition 4 *Let Sol_p the integral solution that places at stage t the m-th facility at the $(m-1)N+p$ node of V_t^+ i.e. $Y_{(m-1)N+p}^t$.*

Notice that the integral solution *Sol* referred in Lemma 3.2 corresponds to Sol_1. The proof of Lemma 3.2 follows directly by Lemma 3.4 and Lemma 3.5 that conclude this section.

Lemma 3.4 *Let S_k^t be the moving cost of facility k at stage t in the optimal fractional solution and $MovingCost(Sol_p)$ the total moving cost of the facilities in the integral solution Sol_p. Then,*

$$\frac{1}{N}\sum_{p=1}^{N} MovingCost(Sol_p) = \sum_{t=1}^{T}\sum_{k\in F} S_k^t$$

Proof: By the definition of the solutions Sol_p we have that:

$$\begin{aligned}
\frac{1}{N}\sum_{p=1}^{N} MovingCost(Sol_p) &= \frac{1}{N}\sum_{p=1}^{N}\sum_{t=1}^{T}\sum_{m=1}^{K} d(Y^{t-1}_{(m-1)N+p}, Y^{t}_{(m-1)N+p}) \\
&= \frac{1}{N}\sum_{t=1}^{T}\sum_{m=1}^{K}\sum_{p=1}^{N} d(Y^{t-1}_{(m-1)N+p}, Y^{t}_{(m-1)N+p}) \\
&= \frac{1}{N}\sum_{t=1}^{T}\sum_{j=1}^{K\cdot N} d(Y_j^{t-1}, Y_j^t) \\
&= \sum_{t=1}^{T}\sum_{k\in F} S_k^t
\end{aligned}$$

The last equality comes from Lemma 3.3. □

Lemma 3.4 states that if we pick uniformly at random one of the N integral solutions $\{Sol_p\}_{p=1}^N$, then the expected moving cost that we will pay is equal to the moving cost paid by the optimal fractional solution. Interestingly, the same holds for the expected connection cost. This is formally stated in Lemma 3.5 and it is where Observation 1 comes into play.

Lemma 3.5 *Let $ConCost_i^t(Sol_p)$ denote the connection cost of agent i at stage t in Sol_p. Then,*

$$\frac{1}{N}\sum_{p=1}^{N} ConCost_i^t(Sol_p) = \sum_{j\in V} d(Loc(i,t),j)x_{ij}^t$$

As already mentioned, the proof of Lemma 3.5 crucially makes use of Observation 1 and is presented in the subsection A.1 of the Appendix. Combining Lemma 3.4 and 3.5 we get that if we pick an integral solution Sol_p uniformly at random, the average total cost that we pay is Z^*_{LP}, where Z^*_{LP} is the optimal fractional cost. More precisely,

$$\begin{aligned}
\frac{1}{N}\sum_{p=1}^{N} Cost(Sol_p) &= \frac{1}{N}\sum_{p=1}^{N}[MovingCost(Sol_p) + \sum_{t=1}^{T}\sum_{i\in C} ConCost_i^t(Solp)] \\
&= \sum_{t=1}^{T}[\sum_{k=1}^{K} S_k^t + \sum_{i\in C}\sum_{j\in V} d(\mathrm{Loc}(i,t),j)x_{ij}^t] \\
&= Z^*_{LP}
\end{aligned}$$

Since $Sol_p \geq Z^*_{LP}$, we have that $Sol_1 = \cdots = Sol_N = Z^*_{LP}$ and this proves Lemma 3.2.

3.4 Rounding the General Case

In this section, we use Lemma 3.2 to prove Theorem 3.1. As already discussed, Assumption 1 is not satisfied in general by the fractional solution of the linear program (1). Each $S^t_{kj\ell}$ will be either 0 or $A^t_{kj\ell}/N^t_{kj\ell}$, for positive some integers $A^t_{kj\ell}, N^t_{kj\ell}$. However, each positive f^t_{kj} will have the form B^t_{kj}/N, where $N = \Pi_{S^t_{kj\ell}>0} N^t_{kj\ell}$. This is due to the constraint $f^t_{kj} = \sum_{j\in V} S^t_{kj\ell}$.

Consider the path $P' = (V', E')$ constructed from path $P = (V, E)$ as follows: Each node $j \in V$ is split into KN *copies* $\{j_1, \ldots, j_{KN}\}$ with zero distance between them. Consider also the LP (1), when the underlying path is $P' = (V', E')$ and at each stage t, each agent i is located to a node of V' that is a *copy* of i's original location, $Loc'(i,t) = \ell \in V'$, where $\ell \in \text{Copies}(\text{Loc}(i,t))$. Although these are two different LP's, they are closely related since a solution for the one can be converted to a solution for the other with the exact same cost. This is due to the fact that for all $j, h \in V$, $d(j,h) = d(j',h')$, where $j' \in \text{Copies}(j)$ and $h' \in \text{Copies}(h)$.

The reason that we defined P' and the second LP is the following: Given an optimal fractional solution of the LP defined for P, we will construct a fractional solution for the LP defined for P' with the exact same cost, which additionally satisfies Assumption 1. Then, using Lemma 3.2 we can obtain an integral solution for P' with the same cost. This integral solution for P' can be easily converted to an integral solution for P. We finally show that these steps are done *all at once* by the rounding scheme of Algorithm 1 and this concludes the proof of Theorem 3.1.

Given the fractional positions $\{f^t_{kj}\}_{t\geq 1}$ of the optimal solution of the LP formulated for $P = (V, E)$, we construct the fractional positions of the facilities in $P' = (V', E')$ as follows: If $f^t_{kj} = B^t_{kj}/N$, then facility k puts a $1/N$ amount of facility in B^t_{kj} nodes of the set $\text{Copies}(j) = \{j_1, \ldots, j_{KN}\}$ that have a 0 amount of facility. The latter is possible since there are exactly KN *copies* of each $j \in V$ and $c^t_j \leq K$ (that is the reason we required KN copies of each node). The values of the rest of the variables are defined in the proof of Lemma 3.7 that is presented in the end of the section. The key point is that the produced solution $\{f^{'t}_{k\ell}, c^{'t}_j, S^{'t}_{kj\ell}, x^{'t}_{ij}, S^{'t}_k\}$ will satisfy the following properties (see Lemma 3.7):

- its cost equals Z^*_{LP}
- $f^{'t}_{k\ell} = 1/N$ or 0, for each $\ell \in V'$
- $c^{'t}_\ell = 1/N$ or 0, for each $\ell \in V'$
- $c^t_j = \sum_{\ell \in \text{Copies}(j)} c^{'t}_\ell$, for each $j \in V$

Clearly, this solution satisfies Assumption 1 and thus Lemma 3.2 can be applied. This implies that the integral solution for P' that places the m-th facility to the $(m-1)N+1$ node of $V^{'+}_t$ ($Y^{'t}_{(m-1)N+1} \in V'$) has cost Z^*_{LP}. So the integral solution for P that places the m-th facility to the node $j^t_m \in V$, such that $Y^{'t}_{(m-1)N+1} \in \text{Copies}(j^t_m)$, has again cost Z^*_{LP}.

A naive way to determine the nodes j^t_m is to calculate N, construct P' and its fractional solution, find the nodes $Y^{'t}_{(m-1)N+1}$ and determine the nodes j^t_m of P. Obviously, this rounding scheme requires exponential time. Fortunately, Lemma 3.6 provides a linear time rounding scheme to determine the node j^t_m given the optimal fractional solution of $P = (V, E)$. This concludes the proof of Theorem 3.1.

Lemma 3.6 *The $(m-1)N+1$ node of $V^{'+}_t$ is a* copy *of the node $j^t_m \in V$ if and only if*

$$\sum_{\ell=1}^{j^t_m - 1} c^t_\ell \leq m-1 < \sum_{\ell=1}^{j^t_m} c^t_\ell$$

.

Proof: Let $(m-1)N+1$ node of $V_t'^{+}$ be a *copy* of the node $j_m^t \in V_t^+$. Then

$$\sum_{\ell=1}^{j_m^t-1} c_\ell^t = \sum_{\ell=1}^{j_m^t-1} \sum_{\ell' \in \text{Copies}(\ell)} c_{\ell'}^{'t} \leq (m-1)N\frac{1}{N} = m-1$$

$$\sum_{\ell=1}^{j_m^t} c_\ell^t = \sum_{\ell=1}^{j_m^t} \sum_{\ell' \in \text{Copies}(\ell)} c_{\ell'}^{'t} = ((m-1)N+1)\frac{1}{N} > m-1$$

The above equations hold because of the property $c_\ell^t = \sum_{\ell' \in \text{Copies}(\ell)} c_{\ell'}^{'t}$ and that $c_{\ell'}^{'t}$ is either 0 or $1/N$. Now, let $\sum_{\ell=1}^{j_m^t-1} c_\ell^t \leq m-1 < \sum_{\ell=1}^{j_m^t} c_\ell^t$ and assume that $(m-1)N+1$-th node of V_t^+ is a copy of $j \in V$. If $j < j_m^t$, then $\sum_{\ell=1}^{j} c_\ell^t > m-1$ and if $j > j_m^t$, then $\sum_{\ell=1}^{j_m^t} < m-1$. As a result, $j = j_m^t$. □

Remark 1 *We remark that the nodes j_m^t can be determined with an even simpler way than that presented in Algorithm 1. That is j_m^t is the most left node such that $f_{mj}^t > 0$. However, this rounding strategy requires some additional analysis.*

Lemma 3.7 *Let $\{f_{kj}^t, c_j^t, S_{kjl}^t, x_{ij}^t\}_{t\geq 1}$ the optimal fractional solution for the LP (1) with underlying path P. Then, there exists a solution $\{f_{kj}^{'t}, c_j^{'t}, S_{kjl}^{'t}, x_{ij}^{'t}, S_k^{'t}\}_{t\geq 1}$ of the LP (1) with underlying path P' such that*

1. *Its cost is Z_{LP}^*.*
2. *$f_{k\ell}^{'t} = 1/N$ or 0, for each $\ell \in V'$*
3. *$c_\ell^{'t} = 1/N$ or 0, for each $\ell \in V'$*
4. *$c_j^t = \sum_{\ell \in Copies(j)} c_\ell^{'t}$, for each $j \in V$*

Proof: First, we set values to the variables $f_{kj}^{'t}$. Initially, all $f_{kj}^{'t} = 0$. We know that if $f_{kj}^t > 0$, then it equals B_{kj}^t/N, for some positive integer B_{kj}^t. For each such f_{kj}^t, we find $u_1, \ldots, u_{B_{kj}^t} \in \text{Copies}(j)$ with $f_{ku_h}^{'t} = 0$. Then, we set $f_{ku_h}^{'t} = 1/N$ for $h = \{1, B_{kj}^t\}$. Since there are KN copies of each node $j \in V$ and $\sum_{j\in V} f_{kj}^t \leq K$, we can always find sufficient copies of j with $f_{ku}^{'t} = 0$. When this step is terminated, we are sure that conditions $2, 3, 4$ are satisfied.

We continue with the variables $S_{kj\ell}^{'t}$. Initially, all $S_{kj\ell}^{'t} = 0$. Then, each positive $S_{kj\ell}^t$ has the form $B_{kj\ell}^t/N$. Let $B = B_{kj\ell}^t$ to simplify notation. We now find B copies of $u_1, \ldots, u_B$ of j and $v_1, \ldots, v_B$ of ℓ so that

- $f_{ku_1}^{'t} = \cdots = f_{ku_B}^{'t} = f_{kv_1}^{'t} = \cdots = f_{kv_B}^{'t} = 1/N$
- $S_{ku_1h}^{'t} = \cdots = S_{ku_Bh}^{'t} = S_{khv_1}^{'t} = \cdots = S_{khv_B}^{'t} = 0$ for all $h \in V'$

We then set $S_{ku_1v_1}^{'t} = \cdots = S_{ku_Bv_B}^{'t} = 1/N$. Again, since $\sum_{\ell\in V} S_{kj\ell}^t = f_{kj}^t$ and $\sum_{j\in V} S_{kj\ell}^t = f_{k\ell}^t$ we can always find $B_{kj\ell}^t$ pairs of copies of j and ℓ that satisfy the above requirements. We can now prove that the movement cost of each facility k is the same in both solutions.

$$\begin{aligned}
\sum_{j\in V}\sum_{\ell\in V} d(j,\ell) S_{kj\ell}^t &= \sum_{j\in V}\sum_{\ell\in V} d(j,\ell) B_{kj\ell}^t/N \\
&= \sum_{j\in V}\sum_{\ell\in V}\sum_{h\in \text{Copies}(j)}\sum_{h'\in \text{Copies}(\ell)} S_{khh'}^{'t} d(h,h') \\
&= \sum_{j'\in V'}\sum_{\ell'\in V'} S_{kj'\ell'}^{'t} d(j',\ell')
\end{aligned}$$

The second equality follows from the fact that h, h' are copies of j, ℓ respectively and thus $d(h, h') = d(j, \ell)$.

Finally, set values to the variables x'^t_{ij} for each $j \in V'$. Again, each positive x^t_{ij} equals B^t_{ij}/N, for some positive integer. We take B^t_{ij} copies of j, $u_1, \ldots, u_{B^t_{ij}}$ and set $x'^t_{iu_1} = \cdots = x'^t_{iu_{B^t_{ij}}} = 1/N$. The connection cost of each agent i remains the same since

$$\begin{aligned}
\sum_{j \in V} d(\mathrm{Loc}(i,t), j) x^t_{ij} &= \sum_{j \in V} d(\mathrm{Loc}(i,t), j) B^t_{ij}/N \\
&= \sum_{j \in V} d(\mathrm{Loc}(i,t), j) \sum_{j' \in \mathrm{Copies}(j)} x'^t_{ij'} \\
&= \sum_{j \in V} \sum_{j' \in \mathrm{Copies}(j)} d(\mathrm{Loc}'(i,t), j') x'^t_{ij'} \\
&= \sum_{h \in V'} d(\mathrm{Loc}'(i,t), h) x'^t_{ih}
\end{aligned}$$

The third equality holds since $\mathrm{Loc}'(i,t) \in \mathrm{Copies}(\mathrm{Loc}(i,t))$. □

4 A Constant-Competitive Algorithm for the Online 2-Facility Reallocation Problem

In this section, we present an algorithm for the online *2-facility reallocation problem* and we discuss the core ideas that prove its performance guarantee. The online algorithm, denoted as Algorithm 2, consists of two major steps.

In Step 1, facilities are initially moved towards the positions of the agents. We remark that in Step 1, the final positions of the facilities at stage t are not yet determined. The purpose of this step is to bring at least one facility close to the agents. This initial moving consists of three cases (see Figure 2), depending only on the relative positions of the facilities at stage $t-1$ and the agents at stage t.

In Step 2, our algorithm determines the final positions of the facilities x^t_1, x^t_2. Notice that after Step 1, at least one of the facilities is inside the interval $[\alpha^t_1, \alpha^t_n]$, meaning that at least one of the facilities is close to the agents. As a result, our algorithm may need to decide between moving the second facility close to the agents or just letting the agents connect to the facility that is already close to them. Obviously, the first choice may lead to small connection cost, but large moving cost, while the second has the exact opposite effect. Roughly speaking, Algorithm 2 does the following: If the connection cost of the agents, when placing just one facility optimally, is not much greater than the cost for moving the second facility inside $[\alpha^t_1, \alpha^t_n]$, then Algorithm 2 puts the first facility to the position that minimizes the connection cost, if one facility is used. Otherwise, it puts the facilities to the positions that minimize the connection cost, if two facilities are used. The above cases are depicted in Figure 3. We formalize how this choice is performed, introducing some additional notation.

Definition 5

- *$C_t = \{\alpha^t_1, \ldots, \alpha^t_n\}$ denotes the positions of the agents at stage t ordered from left to right.*
- *If C is a set of positions with $|C| = 2k, k \in \mathbb{N}_{>0}$, then M_C denotes the median interval of the set, which is the interval $[\alpha_{n/2}, \alpha_{n/2+1}]$. If $|C| = 2k+1, k \in \mathbb{N}_0$, then M_C is a single point.*
- *$H(C)$ denotes the optimal connection cost for the set C when all agents of C connect to just one facility. That is $H(C) = \sum_{\alpha \in C} |\alpha - M_C|$. We also define $H(\emptyset) = 0$.*

- *C^*_{1t} (resp. C^*_{2t}) denotes the positions of the agents that connect to facility 1 (resp. 2) at stage t in the optimal solution x^*. C_{1t} (resp. C_{2t}) denotes the positions of the agents that connect to facility 1 (resp. 2) at stage t in the solution produced by Algorithm 2.*

Algorithm 2: Selecting x_1^t and x_2^t

Data: At stage $t \geq 1$ the new positions of the agents $C_t = \{\alpha_1^t, \ldots, \alpha_n^t\}$, ordered from left to right, arrive

Step 1: Moving the facilities towards the agents

$z_1 \leftarrow x_1^{t-1}$, $z_2 \leftarrow x_2^{t-1}$

if $z_1 > \alpha_n^t$ **then**

move facility 1 to the left until it hits α_n^t

$z_1 \leftarrow \alpha_n^t$

end

if $z_2 < \alpha_1^t$ **then**

move facility 2 to the right until it hits α_1^t

$z_2 \leftarrow \alpha_1^t$

end

if $z_1 < \alpha_1^t$ ***and*** $z_2 > \alpha_n^t$ **then**

move facility 1 to the right and facility 2 to the left until a facility hits $[\alpha_1^t, \alpha_n^t]$

$z_1 \leftarrow z_1 + \min(|x_1^{t-1} - \alpha_1^t|, |x_2^{t-1} - \alpha_n^t|)$

$z_2 \leftarrow z_2 - \min(|x_1^{t-1} - \alpha_1^t|, |x_2^{t-1} - \alpha_n^t|)$

end

Step 2: Selecting the final position of the facilities

if $\alpha_1^t \leq z_1 \leq \alpha_n^t$ ***and*** $z_2 - \alpha_n^t \geq 3H(C_t)$ **then**

put facility 1 to the median of C_t and move facility 2 to the left by $3H(C_t)$

$x_1^t \leftarrow M_{C_t}$, $x_2^t \leftarrow z_2 - 3H(C_t)$

end

if $\alpha_1^t \leq z_2 \leq \alpha_n^t$ ***and*** $\alpha_1^t - z_1 \geq 3H(C_t)$ **then**

put facility 2 to the median of C_t and move facility 1 to the right by $3H(C_t)$

$x_1^t \leftarrow z_1 + 3H(C_t), x_2^t \leftarrow M_{C_t}$

end

else

Compute the optimal partition (O_1, O_2) of C_t that minimizes the connection cost at stage t.

Put facility 1 to the median of O_1 and facility 2 to the median of O_2.

$x_1^t \leftarrow M_{O_1}, x_2^t \leftarrow M_{O_2}$

end

We first mention that Algorithm 2 seems much more complicated than it really is (the first two cases are symmetric both in Step 1 and Step 2). In fact, only the last two cases are difficult to handle and we explain them subsequently. The performance guarantee of Algorithm 2 is formally stated in Theorem 4.1.

Theorem 4.1 *Let $x = \{x_1^t, x_2^t\}_{t \geq 1}$ the solution produced by Algorithm 2 and x^* the optimal solution. Then,*

$$Cost(x) \leq 63 \cdot Cost(x^*) + |x_1^0 - x_2^0|$$

where x_1^0, x_2^0 are the initial positions of the facilities.

The rest of the section is dedicated to provide a proof sketch (some proofs are included in subsection A.2 of the Appendix) of Theorem 4.1. Although it is possible to improve the competitive ratio of Algorithm 2 by a much more technically involved analysis, we stress here that it is not possible to turn the result into

any constant factor. The reason is that the *2-facility reallocation problem* on the line is a generalization of *2-server problem* on the line, which has a lower bound of 2 on the competitive ratio of any online algorithm. Before proceeding, we present Lemma 4.1 that is a key component in the subsequent analysis and that reveals the real difficulty of the online 2-*facility reallocation problem*.

Lemma 4.1 *Let the optimal solution x^* and C^*_{1t}, C^*_{2t} the set of agents that connect respectively to facility* 1 *and* 2 *at stage t. Let the solution $y^t = (y^t_1, y^t_2)$ defined as follows:*

$$y^t_k = \begin{cases} M_{C^*_{kt}} & \text{if } C^*_{kt} \neq \emptyset \\ x^{*t}_k & \text{if } C^*_{kt} = \emptyset \end{cases}$$

Then, the following inequality holds.

$$\sum_{t=1}^{T} \left[\sum_{k=1}^{2} [H(C^*_{kt}) + |y^t_k - y^{t-1}_k|] \right] \leq 3 \cdot Cost(x^*)$$

Proof: Since $\sum_{t=1}^{T} \sum_{k=1}^{2} H(C^*_{kt}) = \sum_{t=1}^{T} \sum_{k=1}^{2} \sum_{a \in C^*_{kt}} |x^{*t}_k - a|$, we only have to prove that

$$\sum_{t=1}^{T} \sum_{k=1}^{2} |y^t_k - y^{t-1}_k| \leq 2 \sum_{t=1}^{T} \sum_{k=1}^{2} \left[H(C^*_{kt}) + |x^{*t}_k - x^{*t-1}_k| \right]$$

From the triangle inequality, we have that

$$\sum_{t=1}^{T} \sum_{k=1}^{2} |y^t_k - y^{t-1}_k| \leq \sum_{t=1}^{T} \sum_{k=1}^{2} [|y^t_k - x^{*t}_k| + |y^{t-1}_k - x^{*t-1}_k| + |x^{*t}_k - x^{*t-1}_k|]$$

The right hand side of the inequality is maximized, when $y^t_k \neq x^{*t}_k$ and $y^{t-1}_k \neq x^{*t-1}_k$ for $k = 1, 2$, namely when $C^*_{kt}, C^*_{k(t-1)} \neq \emptyset$. Since y^t_k (resp. y^{t-1}_k) is the median agent (lies in the median interval of C^*_{kt} in the case $|C^*_{kt}| = 2k$) of C^*_{kt} (resp.$C^*_{k(t-1)}$) in this case, x^{*t}_k (resp. x^{*t-1}_k) has to connect y^t_k (resp. y^{t-1}_k). Thus, $\sum_{t=1}^{T} \sum_{k=1}^{2} |y^t_k - x^{*t}_k| + |y^{t-1}_k - x^{*t-1}_k|$ can be upper bounded by the optimal connection cost, which is

$$\sum_{t=1}^{T} \sum_{k=1}^{2} \sum_{a \in C^*_{kt}} |x^{*t}_k - a| + \sum_{t=1}^{T} \sum_{k=1}^{2} \sum_{a \in C^*_{k(t-1)}} |x^{*t-1}_k - a| \leq 2 \sum_{t=1}^{T} \sum_{k=1}^{2} H(C^*_{kt})$$

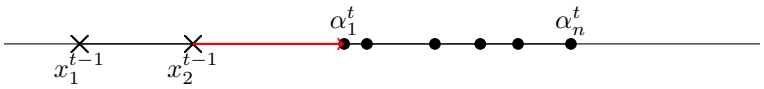
□

Lemma 4.1 indicates that the real difficulty of the problem is not determining the exact positions of the facilities in the optimal solution, but to determine the *service clusters* that the optimal solution forms. In fact, if we knew the clusters C^*_{1t}, C^*_{2t}, then Lemma 4.1 provides us with a 3-approximation algorithm. Obviously, this information cannot be acquired in the online setting, since C^*_{1t}, C^*_{2t} depend on the future positions of the agents that we do not know. We prove that Algorithm 2 has an approximation guarantee of 21 with respect to the solution y, that directly translates to an approximation guarantee of 63 with respect to $Cost(x^*)$. The latter is formally stated in Lemma 4.2 and is the main result of this section.

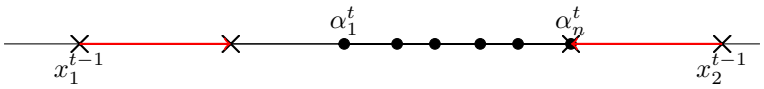
Lemma 4.2 *Let $x = \{x^t_1, x^t_2\}_{t \geq 1}$ be the solution produced by Algorithm 2. Then, the cost paid by solution x at stage t, $\sum_{k=1}^{2} |x^t_k - x^{t-1}_k| + \sum_{i=1}^{n} \min_{k=1,2} |x^t_k - \alpha^t_i|$, is at most*

$$21 \sum_{k=1}^{2} [H(C^*_{kt}) + |y^t_k - y^{t-1}_k|] + \Phi_t(x^t) - \Phi_{t-1}(x^{t-1})$$

where $\Phi_t(x_1, x_2) = 2(|x_1 - y^t_1| + |x_2 - y^t_2|) + |x_1 - x_2|$.



If both facilities 1 and 2 are on the left of the agents, then facility 2 is moved to the right until hitting the position of the leftmost agent (the case with facilities 1 and 2 on the right of agents is symmetric).



If facility 1 is on the left of the agents and facility 2 is on the right of the agents, then both facilities are moved with the same speed towards the interval $[\alpha_1^t, \alpha_n^t]$ until one of them hits the interval.

Figure 2: Step 1 of Algorithm 2 is depicted. After this step, the positions of the facilities are denoted by z_1, z_2 in Algorithm 2.

Lemma 4.2 directly implies Theorem 4.1 by applying a telescopic sum over all t and then applying Lemma 4.1. Notice that the additive term $|x_1^0 - x_2^0|$ in Theorem 4.1 depends only on the initial positions of the facilities and follows from the fact that $\Phi_0(x^0) = |x_1^0 - x_2^0|$. Since the additive term is a constant independent from the request sequence (the client positions C_t), it is common to define the competitive ratio of an online algorithm as in Section 2. In the rest of the section, we present the proof ideas of Lemma 4.2, which come together with explaining Steps 1 and 2 of our algorithm. Let us start with explaining Step 1. First, note that since $x_1^0 \leq x_2^0$, then $x_1^t \leq x_2^t$ by our algorithm construction. Now, assume that $x_2^{t-1} \leq \alpha_1^t$ (second case). Before deciding the exact positions of the facilities, we can *safely* move facility 2 to the right until reaching α_1^t. The term *safely* means that this moving cost is *roughly* upper bounded by the moving cost $\sum_{k=1}^{2} |y_k^t - y_k^{t-1}|$. This *safe moving* applies to all three cases of Step 1 in Algorithm 2 and is formally stated in Lemma 4.3.

Lemma 4.3 *Let $z = (z_1, z_2)$ denote the values of the variables z_1, z_2 after Step 1 of Algorithm 2. Then,*

$$\sum_{k=1}^{2} |z_k - x_k^{t-1}| \leq 2 \sum_{k=1}^{2} |y_k^t - y_k^{t-1}| - \Phi_t(z) + \Phi_{t-1}(x^{t-1})$$

The proof of Lemma 4.3 can be found in subsection A.2 of the Appendix. Lemma 4.3 reveals the basic idea of Step 1 performed by the online algorithm. We remind that Step 1 is performed by Algorithm 2 if both facilities are outside the interval C_t at the beginning of stage t. Therefore, it distinguishes between the three cases depicted in Figure 2 (We show 2 cases since the case with both facilities on the right of the agents is symmetric to the first). Moving with the same speed towards the interval $[a_1^t, a_n^t]$ results to the same moving cost for both facilities; both facilities will move the distance of the facility which is closest to its closest agent.

According to the *geometry* of the agents' positions, we can identify a *safe move* whose cost is also paid by solution y for moving the facilities. Moreover, the proof of Lemma 4.3 reveals why we compare our algorithm with the solution y and not directly with x^*. All these *safe moves* are based on the fact that either y_1^t or y_2^t lies in the $C_t = [\alpha_1^t, \alpha_n^t]$ (the latter does not necessarily hold for x^*). Finally, the *potential function* $\Phi_t(x_1, x_2)$ is crucial, since it permits *safe moves*, when all agents are on the right/left of the facilities (first/second case) as well as when they are contained in the interval $[x_1^{t-1}, x_2^{t-1}]$ (third case). This idea was first developed for the K-server problem Koutsoupias [18].

Up next, we analyze the ideas of Step 2. We now need to bound the connection cost plus some additional moving cost from the point where *the safe move stopped*. The following lemma formalizes the guarantees

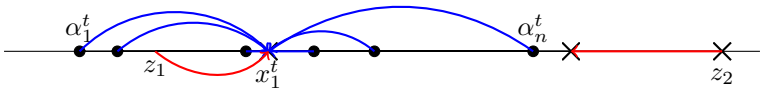
provided by Algorithm 2 after the execution of Step 2. The full proof of Lemma 4.4 can be found in subsection A.2 of the Appendix.

Lemma 4.4 Let $x^t = (x_1^t, x_2^t)$ denote the locations of facilities at stage t after the execution of Step 2. Then,

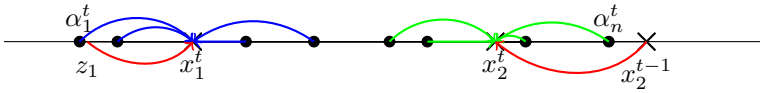
$$\sum_{k=1}^{2}[H(C_{kt}) + |x_k^t - z_k|] \leq 21 \sum_{k=1}^{2} H(C_{kt}^*) - \Phi_t(x^t) + \Phi_t(z)$$

When Algorithm 2 performs Step 2, we know that at least one facility lies inside the interval C_t. This facility will definitely connect some agents of C_t, since we can charge it a small moving cost even if it connects all agents. Thus, the algorithm needs only to decide whether it will connect agents by using only one facility or by using both facilities. The decision depends on the distance between the facility, which is outside of C_t (in case there is one), and the closest agent to this facility. If this distance is "small" (resp. if the facility is already inside the interval), Algorithm 2 will connect agents to both facilities minimizing the connection cost using two facilities. This will guarantee that the moving cost and connection cost incurred are relatively small compared to the cost of solution y.

Now, if the facility, which is outside the interval C_t, is "far" from its closest agent, Algorithm 2 moves this facility towards C_t by a distance, depending on the optimal connection cost using one facility, and serves all agents using the facility, which is inside C_t. Then, we can prove that this move is sufficient to bound the total cost of the algorithm compared to the cost of solution y, even if y has arbitrarily smaller connection cost (if it uses both facilities to serve the agents). The choices of Algorithm 2 are depicted in Figure 3. We provide more detailed Figures based on the analysis of Algorithm 2 in subsection A.2 of the Appendix.



The first choice of Step 2 is depicted. In this case, the facility initially lying inside the interval $[\alpha_1^t, \alpha_n^t]$ moves to the median of agents. In this position, the connection cost is minimized using one facility.



The second choice of Step 2 is depicted. Facilities are placed to the positions, where the connection cost of the agents is minimized using two facilities.

Figure 3: Step 2 of Algorithm 2 is depicted.

5 Open Problems

Regarding the offline variant of the *K-facility reallocation* problem, it would be interesting to consider the problem in general metric spaces. Since *K-facility reallocation* is essentially a dynamic *K-median* problem, a main open problem is to design approximation algorithms for this problem as well as to find lower bounds on the approximation ratio of any offline algorithm in general metric spaces. Turning to the online variant, the main question arising is to design an online algorithm for online *K-facility reallocation* problem on the line. This variant with any number of facilities seems much more intriguing. It would also be interesting to consider randomized algorithms for both the online and the offline variant.

References

[1] Hyung-Chan An, Ashkan Norouzi-Fard, and Ola Svensson. Dynamic facility location via exponential clocks. *ACM Transactions on Algorithms (TALG)*, 13(2):21, 2017.

[2] Nikhil Bansal, Marek Eliáš, Łukasz Jeż, Grigorios Koumoutsos, and Kirk Pruhs. Tight bounds for double coverage against weak adversaries. volume 62, pages 349–365. Springer, 2018.

[3] Nikhil Bansal, Marek Eliéš, Łukasz Jeż, and Grigorios Koumoutsos. The (h, k)-server problem on bounded depth trees. volume 15, page 28. ACM, 2019.

[4] Yair Bartal and Elias Koutsoupias. On the competitive ratio of the work function algorithm for the k-server problem. volume 324, pages 337–345. Elsevier, 2004.

[5] Nadja Betzler, Arkadii Slinko, and Johannes Uhlmann. On the computation of fully proportional representation. *Journal of Artificial Intelligence Research*, 47:475–519, 2013.

[6] Nicolas K. Blanchard and Nicolas Schabanel. Dynamic sum-radii clustering. In *International Workshop on Algorithms and Computation*, pages 30–41. Springer, 2017.

[7] Ioannis Caragiannis, Laurent Gourvès, and Jérôme Monnot. Achieving proportional representation in conference programs. In *Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*, pages 144–150, 2016.

[8] Christian Coester and Elias Koutsoupias. The online k-taxi problem. *To appear in STOC 2019*, 2019. URL `http://arxiv.org/abs/1807.06645`.

[9] Christian Coester, Elias Koutsoupias, and Philip Lazos. The infinite server problem. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[10] Bart de Keijzer and D. Wojtczak. Facility reallocation on the line. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 188–194, 2018.

[11] Gabriella Divéki and Csanád Imreh. Online facility location with facility movements. *Central European Journal of Operations Research*, 19(2):191–200, 2011.

[12] Zvi Drezner and Horst W. Hamacher. *Facility location - Applications and Theory*. Springer, 2002. ISBN 978-3-540-42172-6. URL `http://www.springer.com/computer/swe/book/978-3-540-42172-6`.

[13] David Eisenstat, Claire Mathieu, and Nicolas Schabanel. Facility location in evolving metrics. In *International Colloquium on Automata, Languages, and Programming*, pages 459–470. Springer, 2014.

[14] Amos Fiat, Yuval Rabani, and Yiftach Ravid. Competitive k-server algorithms (extended abstract). In *31st Annual Symposium on Foundations of Computer Science*, pages 454–463, 1990.

[15] Dimitris Fotakis. Online and incremental algorithms for facility location. *ACM SIGACT News*, 42(1): 97–131, 2011.

[16] Zachary Friggstad and Mohammad R Salavatipour. Minimizing movement in mobile facility location problems. *ACM Transactions on Algorithms (TALG)*, 7(3):28, 2011.

[17] Anupam Gupta, Kunal Talwar, and Udi Wieder. Changing bases: Multistage optimization for matroids and matchings. In *International Colloquium on Automata, Languages, and Programming*, pages 563–575. Springer, 2014.

[18] Elias Koutsoupias. The k-server problem. *Computer Science Review*, 3(2):105–118, 2009.

[19] Nevena Lazic. *Message Passing Algorithms for Facility Location Problems.* PhD thesis, University of Toronto, 2011.

[20] Edo Liberty, Ram Sriharsha, and Maxim Sviridenko. An algorithm for online k-means clustering. In *2016 Proceedings of the Eighteenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 81–89. SIAM, 2016.

[21] Adam Meyerson. Online facility location. In *Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*, page 426. IEEE Computer Society, 2001.

A Appendix

A.1 Omitted proofs of Section 3.4

Lemma 3.3 *Let S_k^t the fractional switching cost of facility k at stage t. Then,*

$$\sum_{t=1}^{T}\sum_{k\in F} S_k^t = \frac{1}{N}\sum_{t=1}^{T}\sum_{j=1}^{K\cdot N} d(Y_j^{t-1}, Y_j^t)$$

Proof: By Assumption 1, $c_j^t = 1/N$ if $j \in V_t^+ = \{Y_1^t, \ldots, Y_{KN}^t\}$ and 0 otherwise. Notice that the connection cost of the optimal fractional solution only depends on the variables c_j^t. As a result, $f_{kj}^t, S_k^t, S_{kjl}^t$ must be the optimal solution of the following linear program.

$$\begin{aligned}
\text{minimize} \quad & \sum_{t=1}^{T}\sum_{k=1}^{K} S_k^t \\
\text{s.t.} \quad & \sum_{k\in F} f_{kj}^t = \tfrac{1}{N} && \forall j \in V_t^+, t \in \{1,T\} \\
& \sum_{j\in V_t^+} f_{kj}^t = 1 && \forall k \in F, t \in \{1,T\} \\
& S_k^t = \sum_{j,l\in V} d(j,l) S_{kjl}^t && \forall k \in F, t \in \{1,T\} \\
& \sum_{j\in V_{t-1}^+} S_{kjl}^t = f_{kl}^t && \forall k \in F, l \in V_t^+, t \in \{1,T\} \\
& \sum_{l\in V_t^+} S_{kjl}^t = f_{kj}^{t-1} && \forall k \in F, j \in V_{t-1}^+, t \in \{1,T\}
\end{aligned}$$

Instead of proving that the minimum cost of the above linear program is $\frac{1}{N}\sum_{t=1}^{T}\sum_{j=1}^{K\cdot N} d(Y_j^{t-1}, Y_j^t)$, we prove this for the following more convenient relaxation of the above LP.

$$\begin{aligned}
\text{minimize} \quad & \sum_{t=1}^{T}\sum_{j\in V_{t-1}^+, l\in V_t^+} d(j,l) F_{jl}^t \\
\text{s.t.} \quad & \sum_{l\in V_t^+} F_{jl}^t = \tfrac{1}{N} && \forall j \in V_{t-1}^+, t \in \{1,T\} \\
& \sum_{j\in V_{t-1}^+} F_{jl}^t = \tfrac{1}{N} && \forall l \in V_t^+, t \in \{1,T\}
\end{aligned} \tag{1}$$

It is easy to prove that the LP (1) is a relaxation of the first by setting $F_{jl}^t = \sum_{k\in F} S_{kjl}^t$. Moreover, the above LP describes a flow problem between the nodes V_t^+, where F_{jl}^t is the amount of flow going from node $j \in V_{t-1}^+$ to node $l \in V_t^+$ (see Figure 4).

We are ready for the final step of our proof. First, observe that $F^t_{Y_j^{t-1}Y_j^t}$ is feasible solution for the above LP since $|V^+_{t-1}| = |V^+_t| = K \cdot N$. If we prove that this assignment minimizes the objective, then we are done. Assume that in the optimal solution, $F^t_{Y_1^{t-1}Y_1^t} < 1/N$. Since $\sum_{l \in V^+_t} F^t_{Y_1^{t-1}l} = \frac{1}{N}$, there exists Y_j^t such that $F^t_{Y_1^{t-1}Y_j^t} > 0$. Similarly, by using the second constraint we obtain that $F^t_{Y_{j'}^{t-1}Y_1^t} > 0$. Let $\epsilon = \min(F^t_{Y_1^{t-1}Y_j^t}, F^t_{Y_{j'}^{t-1}Y_1^t})$. Observe that if we increase $F^t_{Y_1^{t-1}Y_1^t}$, $F^t_{Y_{j'}^{t-1}Y_j^t}$ by ϵ and decrease $F^t_{Y_1^{t-1}Y_j^t}$, $F^t_{Y_{j'}^{t-1}Y_1^t}$ by ϵ, we obtain another feasible solution. The cost difference of the two solutions is $D = \epsilon(d(Y_1^{t-1}, Y_j^t) + d(Y_{j'}^{t-1}, Y_1^t) - d(Y_1^{t-1}, Y_1^t) - d(Y_{j'}^{t-1}, Y_j^t))$. If we prove that D is no negative, we are done. We show the latter using the fact that $Y_1^{t-1} \leq Y_{j'}^{t-1}$ and $Y_1^t \leq Y_j^t$. More precisely,

- If $Y_1^{t-1} \leq Y_1^t$ then $D \geq 0$ since $Y_1^t \leq Y_j^t$.
- If $Y_1^{t-1} \geq Y_1^t$ then $D \geq 0$ since $Y_1^{t-1} \leq Y_{j'}^{t-1}$.

Until now, we have shown that in the optimal solution, the node Y_1^{t-1} sends all of her flow to the node Y_1^t. Meaning that Y_1^t does not receive flow by any other node apart from Y_1^{t-1}. By repeating the same argument, it follows that in the optimal solution each node Y_j^{t-1} sends all of her flow to Y_j^t. □

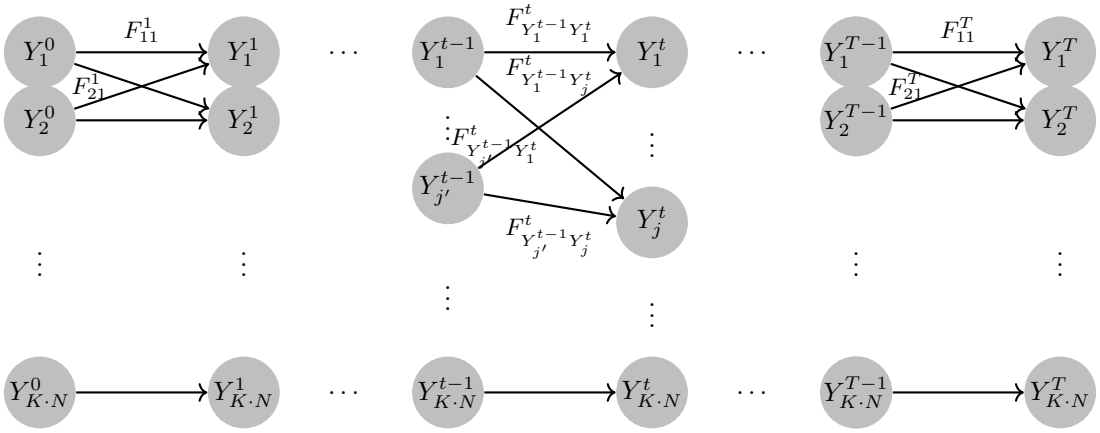


Figure 4: The flow described by LP (1).

Lemma 3.5 *Let $ConCost_i^t(Sol_p)$ denote the connection cost of agent i at stage t in Sol_p of Definition 4. Then*

$$\frac{1}{N}\sum_{p=1}^{N} ConCost_i^t(Sol_p) = \sum_{i \in C}\sum_{j \in V} d(\mathrm{Loc}(i,t), j)x_{ij}^t$$

Proof: We will prove that $\frac{1}{N}\sum_{p=1}^{N} \mathrm{ConCost}_i^t(Sol_p)$ equals $\sum_{j \in V} d(\mathrm{Loc}(i,t), j)x_{ij}^t$. We remind that by Assumption 1, c_j is $1/N$ if $j \in V_t^+$ and 0 otherwise. As a result, in the optimal fractional solution, each agent i finds the N closest to $\mathrm{Loc}(i,t)$ nodes of V_t^+ and receives a $1/N$ amount of service from each one of them. Let us call this set N_i^t. By Observation 1, the nodes in N_i^t must be consecutive nodes of V_t^+ i.e. $N_i^t = \{Y_l^t, \ldots, Y_{l+N-1}^t\}$ and

$$\sum_{j \in V} d(\mathrm{Loc}(i,t), j)x_{ij}^t = \sum_{j=l}^{l+N-1} d(\mathrm{Loc}(i,t), Y_j^t)/N$$

Since Sol_p puts facilities in the positions $\{Y^t_{(m-1)\cdot N+p}\}_{m=1}^K$, there exists a unique node $Y^t_{l(p)} \in N^t_i$ in which Sol_p puts a facility. $Y^t_{l(p)}$ is the closest node to $\text{Loc}(i,t)$ from all the nodes in which Sol_p puts a facility. As a result, $ConCost^t_i(Sol_p) = d(\text{Loc}(i), Y^t_{l(p)})$. Now, summing over p we get,

$$\begin{aligned}\frac{1}{N}\sum_{p=1}^{N} ConCost^t_i(Sol_p) &= \frac{1}{N}\sum_{p=1}^{N} d(\text{Loc}(i), Y^t_{l(p)}) \\ &= \sum_{j=l}^{l+N-1} d(\text{Loc}(i), Y^t_j)/N \\ &= \sum_{j\in V} d(\text{Loc}(i,t), j)x^t_{ij}\end{aligned}$$

□

A.2 Omitted proofs of Section 4

Lemma 4.3 *Let $z = (z_1, z_2)$ denote the values of the variables z_1, z_2 after Step 1 of Algorithm 2. Then,*

$$\sum_{k=1}^{2} |z_k - x^{t-1}_k| \leq 2\sum_{k=1}^{2} |y^t_k - y^{t-1}_k| - \Phi_t(z) + \Phi_{t-1}(x^{t-1})$$

Proof: Assume that $x^{t-1}_2 \leq \alpha^t_1$, then Algorithm 2 will first move facility 2 to α^t_1 ($z_1 = x^{t-1}_1, z_2 = \alpha^t_1$), paying moving cost equal to $|\alpha^t_1 - x^{t-1}_2|$. This moving cost can be bounded with the use of the potential function Φ. More specifically, we have that $\Phi_t(z) - \Phi_t(x^{t-1}) + \Phi_t(x^{t-1}) - \Phi_{t-1}(x^{t-1})$

$$\begin{aligned}&= \Phi_t(z) - \Phi_t(x^{t-1}) + 2\sum_{k=1}^{2}(|y^t_k - x^{t-1}_k| - |y^{t-1}_k - x^{t-1}_k|) \\ &\leq \Phi_t(z) - \Phi_t(x^{t-1}) + 2\sum_{k=1}^{2}|y^t_k - y^{t-1}_k| \qquad (2)\end{aligned}$$

In the considered case $z_1 = x^{t-1}_1, z_2 = \alpha^t_1$, the difference $\Phi_t(z) - \Phi_t(x^{t-1})$ in the potential function equals the quantity $2(|y^t_2 - \alpha^t_1| - |y^t_2 - x^{t-1}_2|) + |x^{t-1}_1 - \alpha^t_1| - |x^{t-1}_1 - x^{t-1}_2|$. By the definition of solution y in Lemma 4.2, either y^t_1 or y^t_2 lies in the interval $[\alpha^t_1, \alpha^t_n]$. Since either y^t_1 or y^t_2 lies in the interval $[a^t_1, a^t_2]$ and $y^t_1 \leq y^t_2$, we have that $a^t_1 \leq y^t_2$. Meaning that z_2 is closer to y^t_2 than x^{t-1}_2 and consequently $2(|y^t_2 - a^t_1| - |y^t_2 - x^{t-1}_2|) = -2|x^{t-1}_2 - a^t_1|$. Therefore, $\Phi_t(z) - \Phi_t(x^{t-1}) = -2|x^{t-1}_2 - a^t_1| + |x^{t-1}_1 - \alpha^t_1| - |x^{t-1}_1 - x^{t-1}_2| = -|a^t_1 - x^{t-1}_2| = -|z_2 - x^{t-1}_2|$, which completes the proof of Lemma 4.3 for this case of Step 1.

Notice that inequality (2) holds for all three cases of Step 1. Thus, one just need to prove that $\Phi_t(z) - \Phi_t(x^{t-1}) \leq -\sum_{k=1}^{2} |z_k - x^{t-1}_k|$ for the other two cases. We prove it for the third case of Step 1, since the second case ($x^{t-1}_1 \geq \alpha^t_n$) is just symmetric to the first case.

In the third case of Step 1, we have that $x^{t-1}_1 < a^t_1$, $x^{t-1}_2 > a^t_n$, $z_1 = x^{t-1}_1 + \min(|x^{t-1}_1 - a^t_1|, |x^{t-1}_2 - a^t_n|)$ and $z_2 = x^{t-1}_2 - \min(|x^{t-1}_1 - a^t_1|, |x^{t-1}_2 - a^t_n|)$. The difference $\Phi_t(z) - \Phi_t(x^{t-1})$ in the potential function equals the quantity $2(|z_1 - y^t_1| - |x^{t-1}_1 - y^t_1| + |z_2 - y^t_2| - |x^{t-1}_2 - y^t_2|) + |z_1 - z_2| - |x^{t-1}_1 - x^{t-1}_2|$. Now, $|z_1 - z_2|$ - $|x^{t-1}_1 - x^{t-1}_2| = -2\min(|x^{t-1}_1 - \alpha^t_1|, |x^{t-1}_2 - \alpha^t_n|) = -\sum_{k=1}^{2} |z_k - x^{t-1}_k|$. Assume that $y^t_1 \in [a^t_1, a^t_n]$, then $\sum_{k=1}^{2}(|z_k - y^t_k| - |x^{t-1}_k - y^t_k|) \leq 0$ since $|z_1 - y^t_1| - |x^{t-1}_1 - y^t_1| = -\min(|x^{t-1}_1 - \alpha^t_1|, |x^{t-1}_2 - \alpha^t_n|)$ and $|z_2 - y^t_2| - |x^{t-1}_2 - y^t_2| \leq \min(|x^{t-1}_1 - \alpha^t_1|, |x^{t-1}_2 - \alpha^t_n|)$. As a result, inequality (2) holds. Using the same argument in case $y^t_2 \in [a^t_1, a^t_n]$ completes the proof. □

Lemma 4.4 *Let $x^t = (x_1^t, x_2^t)$ denote the locations of facilities at stage t after the execution of Step 2. Then,*

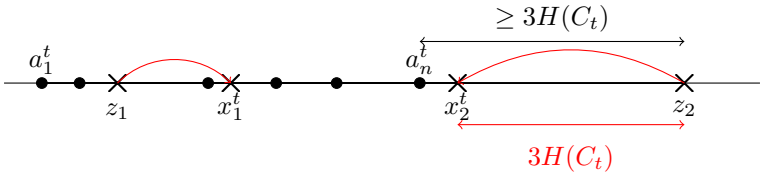
$$\sum_{k=1}^{2}[H(C_{kt}) + |x_k^t - z_k|] \leq 21 \sum_{k=1}^{2} H(C_{kt}^*) - \Phi_t(x^t) + \Phi_t(z)$$

Proof: Observe that by Algorithm 2, either $a_1^t \leq z_1 \leq a_n^t$ or $a_1^t \leq z_2 \leq a_n^t$. As a result, we need to prove the claim for the following 4 cases:

- $a_1 \leq z_1 \leq a_n$ **and** $z_2 - a_n \geq 3H(C_t)$
- $a_1 \leq z_1 \leq a_n$ **and** $z_2 - a_n < 3H(C_t)$
- $a_1 \leq z_2 \leq a_n$ **and** $a_1 - z_1 \geq 3H(C_t)$
- $a_1 \leq z_2 \leq a_n$ **and** $a_1 - z_1 < 3H(C_t)$

We will prove just the first and the second case since the third is symmetric to the first and the forth is symmetric to the second.

In case $a_1 \leq z_1 \leq a_n$ and $z_2 - a_n \geq 3H(C_t)$, Algorithm 2 puts facility 1 in the median of C_t, namely $x_1^t = M_{C_t}$ (or $x_1^t \in M_{C_t}$ in case the number of agents is even), and moves facility 2 to the left by a distance of $3H(C_t)$.



First note that $\sum_{k=1}^{2} H(C_{kt}) \leq H(C_t)$ since $x_1^t \in M_{C_t}$. Then $|x_1^t - z_1| \leq |a_1^t - a_n^t| \leq H(C_t)$ because both x_1^t and z_1 lie in the interval $[a_1^t, a_n^t]$ and $|x_2^t - z_2| = 3H(C_t)$ by Algorithm 2. Therefore, we have that $\sum_{k=1}^{2} H(C_{kt}) + |x_k^t - z_k| \leq 5H(C_t)$.

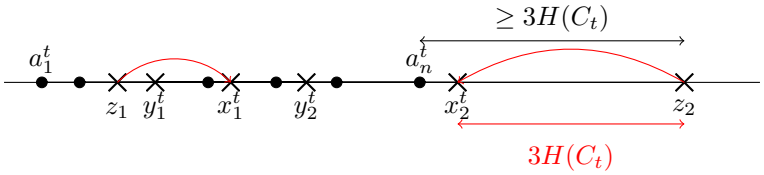
By the geometry of this case and the aforementioned bounds,

$$\begin{aligned}\Phi_t(x^t) - \Phi_t(z) &= 2\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right) + |x_1^t - x_2^t| - |z_1 - z_2| \\ &\leq 2\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right) - 2H(C_t)\end{aligned}$$

Since $\sum_{k=1}^{2}[H(C_{kt}) + |x_k^t - z_k|] \leq 5H(C_t)$, the challenge is bounding $\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right)$ by the connection cost $\sum_{k=1}^{2} H(C_{kt}^*)$. In case $\sum_{k=1}^{2} H(C_{kt}^*) = H(C_t)$ meaning that y^t connects all agents to just one facility things are quite easy, at least intuitively. The real difficulty arises when $C_{1t} \neq \emptyset$ and $C_{2t} \neq \emptyset$, where $\sum_{k=1}^{2} H(C_{kt}^*)$ can be arbitrarily smaller than $H(C_t)$. As we will see in this case x^t gets closer to y^t and the term $\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right)$ becomes negative.

We start with the most challenging case, where $C_{1t}^* \neq \emptyset$ and $C_{2t}^* \neq \emptyset$. We remind that our goal is showing that x^t gets closer to y^t. Since $C_{2t}^* \neq \emptyset$ and $y_2^t \in M_{C_{2t}^*}$ we get that $y_2^t \leq a_n^t$ and as a result $|x_2^t - y_2^t| - |z_2 - y_2^t| = |x_2^t - z_2| - 3H(C_t)$.

$$\Phi_t(x^t) - \Phi_t(z) \leq 2\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right) - 2H(C_t)$$



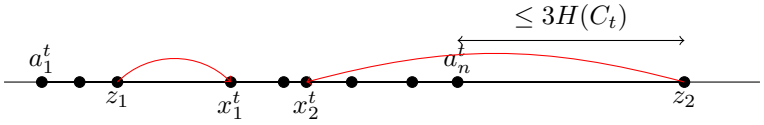
$$\begin{aligned}
&= 2\left(|x_1^t - y_1^t| - |z_1 - y_1^t|\right) + 2\left(|x_2^t - z_2| - |z_2 - y_2^t|\right) - 2H(C_t) \\
&\leq 2|x_1^t - z_1| - 8H(C_t) \\
&\leq 2H(C_t) - 8H(C_t) \\
&\leq -6H(C_t) \\
&\leq \sum_{k=1}^{2} H(C_{kt}^*) - \sum_{k=1}^{2}[H(C_{kt}) + |x_k^t - z_k|]
\end{aligned}$$

Now, assume that $C_{1t}^* = \emptyset$ or $C_{2t}^* = \emptyset$ meaning that $\sum_{k=1}^{2} H(C_{kt}^*) = H(C_t)$. As a result, bounding *everything* by $H(C_t)$ serves our purpose. More formally,

$$\begin{aligned}
\Phi_t(x^t) - \Phi_t(z) &\leq 2\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right) - 2H(C_t) \\
&\leq 2\sum_{k=1}^{2}|x_k^t - z_k| - 2H(C_t) \\
&\leq 6H(C_t) \\
&\leq 11H(C_t) - \sum_{k=1}^{2}\left[H(C_{kt}) + |x_k^t - z_k|\right] \\
&= 11\sum_{k=1}^{2} H(C_{kt}^*) - \sum_{k=1}^{2}\left[H(C_{kt}) + |x_k^t - z_k|\right]
\end{aligned}$$

The forth inequality follows from the fact that $\sum_{k=1}^{2} H(C_{kt}) + |x_k^t - z_k| \leq 5H(C_t)$.

We now need to treat the second case where $a_1 \leq z_1 \leq a_n$ and $z_2 - a_n < 3H(C_t)$. Since Algorithm 2 computes the optimal clustering (C_{1t}, C_{2t}) and puts x_1^t in the interval $M_{C_{1t}}$ and x_2^t in the interval $M_{C_{2t}}$, we are ensured that the connection cost of our solution is less than the connection cost of y^t, $\sum_{k=1}^{2} H(C_{kt}) \leq \sum_{k=1}^{2} H(C_{kt}^*)$, so we are mostly concerned in bounding $\sum_{k=1}^{2} |x_k^t - z_k|$.



The easy case is when $\sum_{k=1}^{2} H(C_{kt}^*) = H(C_t)$. A small difference with the previous case is that we don't know how $|x_2^t - z_2|$ is. However, $z_1, x_1^t, x_2^t \in [a_1^t, \ldots a_n^t]$ and $|x_2^t - z_2| = |x_2^t - a_n^t| + |a_n^t - z_2|$. Thus, $|x_1^t - z_1| + |x_2^t - a_n^t| \leq H(C_t)$, $|a_n^t - z_2| \leq 3H(C_t)$ and therefore $\sum_{k=1}^{2}[H(C_{kt}) + |x_k^t - z_k|] \leq 5H(C_t)$. So we can again bound *everything* by $H(C_t)$.

$$\Phi_t(x^t) - \Phi_t(z) = 2\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right) + |x_1^t - x_2^t| - |z_1 - z_2|$$

$$
\begin{aligned}
&\leq 3\sum_{k=1}^{2}|x_k^t - z_k| \\
&\leq 4\sum_{k=1}^{2}|x_k^t - z_k| - \sum_{k=1}^{2}|x_k^t - z_k| \\
&\leq 20H(C_t) - \sum_{k=1}^{2}\left[|x_k^t - z_k|\right] \\
&\leq 21\sum_{k=1}^{2}H(C_{kt}^*) - \sum_{k=1}^{2}\left[H(C_{kt}) + |x_k^t - z_k|\right]
\end{aligned}
$$

Things become more complicated, when the connection cost $\sum_{k=1}^{2} H(C_{kt}^*)$ is relatively small ($C_{1t}^* \neq \emptyset$ and $C_{2t}^* \neq \emptyset$), where bounding everything by $H(C_t)$ does not work. However, the solutions x^t and y^t will be relatively close in this case. More formally,

$$
\begin{aligned}
\Phi_t(x^t) - \Phi_t(z) &= 2\sum_{k=1}^{2}\left(|x_k^t - y_k^t| - |z_k - y_k^t|\right) + |x_1^t - x_2^t| - |z_1 - z_2| \\
&= 2\sum_{k=1}^{2}|x_k^t - y_k^t| - 2\sum_{k=1}^{2}|z_k - y_k^t| + \sum_{k=1}^{2}|x_k^t - z_k| \\
&= 2\sum_{k=1}^{2}|x_k^t - y_k^t| + 2\sum_{k=1}^{2}\left(|x_k^t - z_k| - |z_k - y_k^t|\right) - \sum_{k=1}^{2}|x_k^t - z_k| \\
&\leq 4\sum_{k=1}^{2}|x_k^t - y_k^t| - \sum_{k=1}^{2}|x_k^t - z_k|
\end{aligned}
$$

We need to upper bound the distance $\sum_{k=1}^{2}|x_k^t - y_k^t|$. Observe that in the solution x^t, the agent at position a_1^t connects to the left facility (facility 1) and the agent at position a_n^t connects to the right facility (facility 2), $|x_1^t - a_1^t| + |x_2^t - a_n^t| \leq \sum_{k=1}^{2} H(C_{kt})$. Since $C_{1t}^* \neq \emptyset$ and $C_{2t}^* \neq \emptyset$, the same holds for the solution y^t. As a result,

$$
\begin{aligned}
\Phi_t(x^t) - \Phi_t(z) &\leq 4\sum_{k=1}^{2}|x_k^t - y_k^t| - \sum_{k=1}^{2}|x_k^t - z_k| \\
&\leq 4\left(|x_1^t - a_1^t| + |y_1^t - a_1^t| + |x_2^t - a_n^t| + |y_2^t - a_n^t|\right) - \sum_{k=1}^{2}|x_k^t - z_k| \\
&\leq 4\sum_{k=1}^{2}\left[H(C_{kt}) + H(C_{kt}^*)\right] - \sum_{k=1}^{2}|x_k^t - z_k| \\
&\leq 9\sum_{k=1}^{2}H(C_{kt}^*) - \sum_{k=1}^{2}\left[H(C_{kt}) + |x_k^t - z_k|\right]
\end{aligned}
$$

□