A Multi-Exchange Local Search Algorithm for the Capacitated Facility Location Problem

Jiawei Zhang* Bo Chen† Yinyu Ye ‡

October 19, 2003

Abstract

We present a multi-exchange local search algorithm for approximating the capacitated facility location problem (CFLP), where a new local improvement operation is introduced that possibly exchanges multiple facilities simultaneously. We give a tight analysis for our algorithm and show that the performance guarantee of the algorithm is between $3+2\sqrt{2}-\epsilon$ and $3+2\sqrt{2}+\epsilon$ for any given constant $\epsilon > 0$. Previously known best approximation ratio for the CFLP is 7.88, due to Mahdian and Pál (2003), based on the operations proposed by Pál, Tardos and Wexler (2001). Our upper bound $3 + 2\sqrt{2} + \epsilon$ also matches the best known ratio, obtained by Chudak and Williamson (1999), for the CFLP with uniform capacities. In order to obtain the tight bound of our new algorithm, we make interesting use of techniques from the area of parallel machine scheduling.

Key words: capacitated facility location, approximation algorithm, local search algorithm.

*Department of Management Science and Engineering, Stanford University, Stanford, CA, 94305, USA. Email:jiazhang@stanford.edu.

†Warwick Business School, The University of Warwick, Coventry, England, UK. Email:B.Chen@warwick.ac.uk.

‡Department of Management Science and Engineering, Stanford University, Stanford, CA, 94305, USA. Email:yinyu-ye@stanford.edu.

1 Introduction

In the capacitated facility location problem (CFLP), we are given a set of clients $\mathcal{D}$ and a set of facilities $\mathcal{F}$. Each client $j \in \mathcal{D}$ has a demand d_j that must be served by one or more open facilities. Each facility $i \in \mathcal{F}$ is specified by a cost f_i, which is incurred when facility i is open, and by a capacity u_i, which is the maximum demand that facility i can serve. The cost for serving one unit demand of client j from facility i is c_{ij}. Here, we wish to open a subset of facilities such that the demands of all the clients are met by the open facilities and the total cost of facility opening and client service is minimized. We assume that unit service costs are non-negative, symmetric, and satisfy the triangle inequality, i.e., for each $i, j, k \in \mathcal{D} \cup \mathcal{F}, c_{ij} \geq 0$, $c_{ij} = c_{ji}$ and $c_{ij} \leq c_{ik} + c_{kj}$.

The CFLP is a well-known NP-hard problem in combinatorial optimization with a large variety of applications, such as location and distribution planning, telecommunication network design, etc. Numerous heuristics and exact algorithms have been proposed for solving CFLP. In this paper, we are concerned with approximation algorithms for the CFLP. Given a minimization problem, an algorithm is said to be a (polynomial) *ρ-approximation* algorithm, if for any instance of the problem, the algorithm runs in polynomial time and outputs a solution that has a cost at most $\rho \geq 1$ times the minimal cost, where ρ is called the performance guarantee or the *approximation ratio* of the algorithm.

Since the work of Shmoys, Tardos and Aardal (1997), designing approximation algorithms for facility location problems has received considerable attention during the past few years. In particular, almost all known approximation techniques, including linear programming relaxation, have been applied to the uncapacitated facility location problem (UFLP), which is a special case of the CFLP with all $u_i = \infty$. The best currently known approximation ratio for the UFLP is 1.517 due to Mahdian, Ye and Zhang (2002 and 2003). Furthermore, Guha and Khuller (1999) proved that the existence of a polynomial time 1.463-approximation algorithm for the UFLP would imply that $P = NP$.

On the other hand, because the (natural) linear programming relaxation for the CFLP is known to have an unbounded integrality gap, all known approximation algorithms for the CFLP with bounded performance guarantee are based on local search techniques. Korupolu, Plaxton and Rajaraman (1998) is the first to show that local search algorithm proposed by Kuehn and

Hamburger (1963) has an approximation ratio of 8 for the case where all the capacities are uniform, i.e., $u_i = u$ for all $i \in \mathcal{F}$. The ratio has been improved to 5.83 by Chudak and Williamson (1998) using a simplified analysis. When the capacities are not uniform, Pál et al. (2001) have recently presented a local search algorithm with performance guarantee between 4 and 8.53. Their algorithm consists of the following three types of local improvement operations: open one facility; open one facility and close one or more facilities; close one facility and open one or more facilities. Very recently, Mahdian and Pál (2003) reduced the upper bound to 7.88 by using operations that combines those in Pál et al. (2001).

The main contribution of this paper is the introduction of a new type of operations, called the multi-exchange operation, which possibly exchanges multiple facilities simultaneously, i.e., close many facilities and open many facilities. In general, the multi-exchange operation cannot be computed in polynomial time since it is NP-hard to determine if such an operation exists so that a solution can be improved. However, we will restrict our attention to a subset of such operations such that the aforementioned determination can be computed efficiently. We will show that the restricted multi-exchange operation generalizes those in Pál et al. (2001).

This new type of operations enables us to approximate the CFLP with a factor of $3 + 2\sqrt{2} + \epsilon$ for any given constant $\epsilon > 0$. The approximation ratio of our algorithm matches what has been proved by Chudak and Williamson (1999) for the uniform capacity case. We also show that our analysis is almost tight. In particular, we construct an instance of the CFLP such that the ratio of the cost resulted from the improved local search algorithm to the optimal cost is $3 + 2\sqrt{2} - \epsilon$ for any given constant $\epsilon > 0$.

The performance guarantee of our algorithm follows from a set of inequalities that is implied by the local optimality of the solution, i.e., none of the legitimate operations can improve the solution. The multi-exchange operations is meant to give stronger inequalities than those implied by simpler operations. The major task of the proof is to identify those inequalities, which involves partitioning the facilities into groups. Here, we make uses of proof techniques from the area of parallel machines scheduling. Our proof also uses the notion of exchange graph from Pál et al. (2001).

The remainder of this paper is organized as follows. In Section 2, we present our new local search operation together with some preliminary analyses. The performance guarantee of the algorithm is proved in Section 3. We present the lower bound ratio example in Section 4 and then

make some remarks in the final section.

2 Local search algorithm

Given a subset $S \subseteq \mathcal{F}$, denote by $C(S)$ the optimal value of the following linear program LP(S):

$$\begin{aligned} \text{LP}(S): \qquad \text{Minimize} & \sum_{i\in S} f_i + \sum_{i\in S}\sum_{j\in\mathcal{D}} c_{ij}x_{ij} \\ \text{subject to} & \sum_{i\in S} x_{ij} = d_j \quad \forall j \in \mathcal{D} \\ & \sum_{j\in\mathcal{D}} x_{ij} \le u_i \quad \forall i \in S \\ & x_{ij} \ge 0 \quad \forall i \in S,\ j \in \mathcal{D} \end{aligned}$$

where x_{ij} denotes the amount of demand of client j served by facility i. The above linear program LP(S) is feasible if and only if $\sum_{i\in S} u_i \ge \sum_{j\in\mathcal{D}} d_j$. If it is infeasible, we denote $C(S) = +\infty$. If it is feasible, then an optimal solution $x_S = (x_{ij})$ and the optimal value $C(S)$ can be found in polynomial time.

Therefore, the CFLP is reduced to finding a subset $S \subseteq \mathcal{F}$, which we simply call a *solution* with service allocation x_S, such that $C(S)$ is minimized. Let $C_f(S) = \sum_{i\in S} f_i$ and $C_s(S) = C(S) - C_f(S)$ denote the respective facility cost and service cost associated with solution S.

Given the current solution S, the algorithm proposed by Pál et al. (2001) performs the following three types of local improvement operations:

- **add**(s): Open a facility $s \in \mathcal{F}\backslash S$ and reassign all the demands to $S\cup\{s\}$ by computing $x_{S\cup\{s\}}$ and $C(S\cup\{s\})$ using linear program LP($S\cup\{s\}$). The cost of this operation is defined as $C(S\cup\{s\}) - C(S)$, which can be computed in polynomial time for each $s \in \mathcal{F}\backslash S$.

- **open**(s, T): Open a facility $s \in \mathcal{F}\backslash S$, close a set of facilities $T \subseteq S$ and reassign all the demands served by T to facility s. Demands served by facilities in $S\backslash T$ will not be affected. Let the *cost* of reassigning one unit of demand of any client j from facility $t \in T$ to facility s be c_{ts}, which is an upper bound on the change of service cost $c_{js} - c_{jt}$. The cost of operation **open**(s, T) is defined as the sum of $(f_s - \sum_{i\in T} f_i)$ and the total cost of reassigning demands

served by facilities in T to s. Pál et al. (2001) have shown that, for each $s \notin S$, if there exists a set T such that the cost of **open**(s, T) is $f_s - \alpha$ for some $\alpha \geq 0$, then one can find in polynomial time a set T' such that the cost of **open**(s, T') is at most $f_s - (1 - \epsilon)\alpha$ for any given $\epsilon > 0$.

- **close**(s, T): Close a facility $s \in S$, open a set of facilities $T \subseteq \mathcal{F} \backslash S$, and reassign all the demands served by s to facilities in T. Demands served by facilities in $S \backslash \{s\}$ will not be affected. Again let the cost of reassigning one unit of demand of any client j from facility s to facility $t \in T$ be c_{st}, which is an upper bound of $c_{jt} - c_{js}$. The cost of operation **close**(s, T) is defined as the sum of $(\sum_{i \in T} f_i - f_s)$ and the total cost of reassigning demands served by s to facilities in T. Similarly, for each $s \in S$, if there exists a set T such that the cost of **close**(s, T) is $\alpha - f_s$ for some $\alpha \geq 0$, then one can find in polynomial time a set T' such that the cost of **close**(s, T) is at most $(1 + \epsilon)\alpha - f_s$ for any given $\epsilon > 0$.

In our algorithm, we introduce an additional type of powerful restricted multi-exchange operations, which is defined below and consists of three parts.

- **multi**(r, R, t, T): (i) Close a set of facilities $\{r\} \cup R \subseteq S$ where $r \notin R$ and R may be empty; (ii) Open a set of facilities $\{t\} \cup T \subseteq \mathcal{F} \backslash S$ where $t \notin T$ and T may be empty; (iii) Reassign the demands served by facilities in R to facility t and the demands served by facility r to $\{t\} \cup T$. Recall that we let c_{st} be the cost of reassigning one unit of demand from facility $s \in \{r\} \cup R$ to facility t and $c_{r\hat{t}}$ be the cost of reassigning one unit of demand from facility r to facility $\hat{t} \in T$. Then, the cost of operation **multi**(r, R, t, T) is defined as the sum of $(f_t + \sum_{i \in T} f_i - f_r - \sum_{i \in R} f_i)$ and the total cost of reassigning (according to the given rules) demands served by facilities in $\{r\} \cup R$ to facilities in $\{t\} \cup T$.

It is clear that **multi**(r, R, t, T) generalizes **open**$(t, R \cup \{r\})$ by letting $T = \emptyset$. Operation **multi**(r, R, t, T) also generalizes **close**$(r, T \cup \{t\})$ by letting $R = \emptyset$. However, we will still use both notions **open**(t, R) and **close**(r, T) to distinguish them from the more general and powerful operation **multi**(r, R, t, T). In the following lemma, we show that the restricted multi-exchange operation can also be implemented efficiently and effectively.

Lemma 1. *For any $r \in S$ and $t \in \mathcal{F}\backslash S$, if there exist a set $R \subseteq S$ and a set $T \subseteq \mathcal{F}\backslash S$ such that the cost of* ***multi****(r, R, t, T) is $A - B$ for some $A \geq 0$ and $B = f_r + \Sigma_{s\in R} f_s$, then one can find in polynomial time sets R' and T' such that the cost of* ***multi****(r, R', t, T') is at most $(1+\epsilon)A-(1-\epsilon)B$ for any given $\epsilon > 0$.*

The ideas behind the proof of the lemma is as follows. Note that in the multi-exchange operation **multi**(r, R, t, T), all demands served by facilities in R are only reassigned to facility t. Let d be the total amount of demand served by facilities in R. For any $s \in S$, denote by $\bar{d}_s$ the total amount of demand served by facility s in the current solution S and let $w_s = f_s - \bar{d}_s c_{st}$, which is the *profit* of closing facility s and reassigning the demands served by s to t. We *split* facility t into two, t' and t'', such that they are the same as facility t except that t' has a reduced capacity d and t'' has a capacity $u_t - d$ and facility cost 0. Let the cost of **close**$(r, \{t''\} \cup T)$ be $\alpha - f_r$ and the cost of **open**(t', R) be $f_t - \Sigma_{s\in R} f_s + \beta$ for some $\alpha, \beta \geq 0$. Then The total cost of operation **multi**(r, R, t, T) is simply the sum of these two costs $(f_t + \alpha + \beta) - (f_r + \Sigma_{s\in R} f_s)$, i.e.,

$$A = f_t + \alpha + \beta. \tag{1}$$

We also have

$$\sum_{s\in R} w_s = \sum_{s\in R} f_s - \beta. \tag{2}$$

Suppose we know the value of d. Then we can perform the split of t into t' and t''. Since we can approximate both operations **close**$(r, \{t''\} \cup T)$ and **open**(t', R) efficiently and effectively, we can do so also for operation **multi**(r, R, t, T). Therefore, the main issue in proving Lemma 1 is to determine d efficiently. In the following proof, we show how to locate d in one of a small number of intervals and then use approximation.

Proof of Lemma ***1.*** Given the constant $\epsilon > 0$, let $k = \lceil 1/\epsilon \rceil$. Let R_0 be any set of facilities in S such that $|R_0| \leq k$. We can enumerate all possible R_0 in polynomial time since k is a constant. If $|R| \leq k$, then there must exist an R_0 such that $R_0 = R$, i.e., we can find R in polynomial time. Therefore, without loss of generality we assume $|R| > k$ and that we have found in polynomial time $R_0 \subseteq R$ such that R_0 contains k most *profitable* elements of R, i.e., those facilities s that have the largest w_s in R. For any $S' \subseteq S$, denote $\bar{d}(S') = \Sigma_{s\in S'} \bar{d}_s$ and $w(S') = \Sigma_{s\in S'} w_s$.

Suppose first that we know the value of $\bar{d}(R)$. Consider the following knapsack problem:

$$\begin{array}{ll} \text{Maximize} & \sum_{s\in S\backslash(R_0\cup\{r\})} w_s z_s \\ \text{subject to} & \sum_{s\in S\backslash(R_0\cup\{r\})} \bar{d}_s z_s \le \bar{d}(R) - \bar{d}(R_0) \\ & z_s \in \{0,1\},\ \forall s \in S\backslash(R_0\cup\{r\}) \end{array} \tag{3}$$

It is evident that $w(R) - w(R_0)$ is the optimal value of the above knapsack problem. Note that, in any optimal solution, $w_s \ge 0$ if $z_s = 1$. Let $\tilde{S} = \{s \in S\backslash(R_0 \cup \{r\} :\ w_s \ge 0\}$ and $\tilde{m} = |\tilde{S}|$. We index the facilities $s \in \tilde{S}$ in order of non-increasing $w_s/\bar{d}_s$ and obtain $\tilde{S} = \{s_1, \ldots, s_{\tilde{m}}\}$. Denote $R_\ell = R_0 \cup \{s_1, \ldots, s_\ell\}$ for any $\ell = 1, \ldots, \tilde{m}$.

Suppose $\bar{d}(R_{\tilde{m}}) > \bar{d}(R)$. The opposite case will be dealt with very easily. Then there exists a (unique) integer m such that $0 \le m < \tilde{m}$ and

$$\sum_{i=1}^{m} \bar{d}_{s_i} \le \bar{d}(R) - \bar{d}(R_0) < \sum_{i=1}^{m+1} \bar{d}_{s_i}. \tag{4}$$

Let S_0 be an optimal solution to the above knapsack problem: $w(S_0) = w(R) - w(R_0)$. Since according to our indexing of elements in $\tilde{S}$,

$$\bar{d}_{s_i} w_s \le \bar{d}_s w_{s_i},\ \forall s \in S_0\backslash\{s_1, \ldots, s_{m+1}\}, \forall s_i \in \{s_1, \ldots, s_{m+1}\}\backslash S_0,$$

summation of these inequalities, together with (4), leads to $w(R) - w(R_0) \le \Sigma_{i=1}^{m+1} w_{s_i}$, i.e.,

$$w(R) \le w(R_{m+1}). \tag{5}$$

Now we replace facility t with $\tilde{t}$, which is the same as t except that $\tilde{t}$ has capacity $u_t - \bar{d}(R_m)$ and facility cost 0. Recall that facility t'' has capacity $u_t - \bar{d}(R)$ and the cost of operation $\mathbf{close}(r, T \cup \{t''\})$ is $\alpha - f_r$. By performing operation $\mathbf{close}(r, \tilde{T} \cup \{\tilde{t}\})$ for $\tilde{T} \subseteq \mathcal{F}\backslash(S \cup \{\tilde{t}\})$, we can find in polynomial time T_m such that the cost of $\mathbf{close}(r, T_m \cup \{\tilde{t}\})$ is at most $(1+\epsilon)\alpha - f_r$, since facility $\tilde{t}$ has at least the same capacity as t'' to accommodate demands reassigned from facility r according to the first inequality of (4). Now we claim the pair $(R', T') = (R_m, T_m)$ has the desired approximation property as stated in the lemma.

In fact, with (5) we have

$$w(R_m) = w(R_{m+1}) - w_{s_{m+1}} \ge w(R) - w_{s_{m+1}}.$$

However, by definition of R_0, we have $w_{s_{m+1}} \leq \frac{1}{k} w(R)$. Hence

$$w(R_m) \geq (1 - 1/k) w(R) \geq (1 - \epsilon) w(R). \tag{6}$$

Consequently, the total cost of **multi**(r, R_m, t, T_m) is at most

$$(f_t - w(R_m)) + ((1 + \epsilon)\alpha - f_r) \leq f_t - (1 - \epsilon) w(R) + (1 + \epsilon)\alpha - f_r.$$

According to (1) and (2), the right-hand side of above is

$$f_t - (1 - \epsilon)\left(\Sigma_{s \in R} f_s - \beta\right) + (1 + \epsilon)\alpha - f_r \leq (1 + \epsilon)A - (1 - \epsilon)B.$$

Therefore, (R_m, T_m) is indeed a good qualified approximation of (R, T).

If $\bar{d}(R_{\tilde{m}}) \leq \bar{d}(R)$, then it is evident that $w(R_{\tilde{m}}) = w(R)$. Hence both (6) and the first inequality of (4) are satisfied for $m = \tilde{m}$. Therefore, in the same way we can quickly find a qualified pair $(R_{\tilde{m}}, T_{\tilde{m}})$.

Now let us drop our initial assumption for the knowledge of $d = \bar{d}(R)$, which is used in the formulation of the knapsack problem (3). Observe that we actually need not solve (3). To obtain the required pair (R_m, T_m), we can simple enumerate all $\tilde{m} + 1$ pairs (R_i, T_i) for $i = 0, 1, \ldots, \tilde{m}$, corresponding to all possibilities that $\bar{d}(R)$ is in one of the intervals $\bar{d}(R_i) \leq \bar{d}(R) < \bar{d}(R_{i+1})$ and $\bar{d}\,(R) \geq \bar{d}(R_{\tilde{m}})$. More specifically, we proceed as follows. For $i = 0, 1, \ldots, \tilde{m}$, let $\tilde{t}_i$ be a facility that is the same as t except that $\tilde{t}_i$ has capacity $u_t - \bar{d}(R_i)$ and facility cost 0. Find T_i in polynomial time by approximating the cheapest operation among all **close**$(r, \tilde{T} \cup \{\tilde{t}_i\})$ for $\tilde{T} \subseteq \mathcal{F} \backslash (S \cup \{\tilde{t}_i\})$. Among the $\tilde{m} + 1$ operations **multi**(r, R_i, t, T_i), $i = 0, 1, \ldots, \tilde{m}$, we choose the cheapest to be used as the required operation **multi**(r, R', t, T'). ∎

Starting with any feasible solution S, our local search algorithm repeatedly performs the legitimate four types of operations defined above until none of them is *admissible*, where an operation is said to be admissible if its cost is less than $-C(S)/p(\|\mathcal{D}\|, \epsilon)$ with $p(\|\mathcal{D}\|, \epsilon) = 3\|\mathcal{D}\|/\epsilon$. Then it is clear that the algorithm will terminate after at most $O\left(p(\|\mathcal{D}\|, \epsilon) \log(C(S)/C(S^*))\right)$ operations. A solution is said to be a *local optimum* if none of the four legitimate operations will have a negative cost. We summarize results in this section in the following theorem.

Theorem 1. *The multi-exchange local search algorithm can identify and implement any admissible operation in polynomial time for any given $\epsilon > 0$. The algorithm terminates in polynomial time and outputs a solution that approximates a local optimum to any pre-specified precision.*

3 Analysis of the algorithm

Now we are ready to analyze our algorithm. Let $S, S^* \subseteq \mathcal{F}$ be any local and global optimal solutions, respectively. As is done in Korupolu et al (1998), Chudak and Williamson (1999), Pál et al (2001), and Mahdian and Pál (2003), we will bound the facility cost $C_f(S)$ and service cost $C_s(S)$ separately. As stated in Mahdian and Pál (2003), any local search algorithm with **add** operation in its repertoire will guarantee a low service cost when it settles down to a local optimum, which is summarized in the following lemma, various versions of which have been proved in the aforementioned papers. We omit its proof here.

Lemma 2. *The service cost $C_s(S)$ is bounded above by the optimal total cost $C(S^*)$.*

Denote $x(i,\cdot) = \sum_{j\in\mathcal{D}} x_{ij}$. Using a simple argument of network flow, Pál, Tardos and Wexler (2001) have proved the following.

Lemma 3. *The optimal value of the following linear program is at most $C_s(S^*) + C_s(S)$:*

$$\begin{aligned} \textit{minimize} \sum_{s\in S\backslash S^*}\sum_{t\in S^*} c_{st}y(s,t) \qquad & (7)\\ \textit{subject to} \sum_{t\in S^*} y(s,t) = x(s,\cdot) \quad \forall s \in S\backslash S^* & \\ \sum_{s\in S\backslash S^*} y(s,t) \le u_t - x(t,\cdot) \quad \forall t \in S^* & \\ y(s,t) \ge 0 \quad \forall s \in S\backslash S^*, t \in S^*. & \end{aligned}$$

For notational clarity, we denote $c(s,t) = c_{st}$. Let $y(s,t)$ be the optimal solution of (7). Consider bipartite graph $G = (S\backslash S^*, S^*, E)$, where $S\backslash S^*$ and S^* are the two sets of nodes in the bipartite graph and $E = \{(s,t) : s \in S\backslash S^*,\ t \in S^* \text{ and } y(s,t) > 0\}$ is the set of edges. Due to the optimality of $y(s,t)$, we can assume without loss of generality that G is a forest.

Consider any tree $\mathcal{T}_r$ of G, which is rooted at node $r \in S^*$. It is evident that the tree has alternate layers of nodes in S^* and in $S \backslash S^*$. For each node t in this tree, let $K(t)$ be the set of children of t. For any $t \in S^*$ and any $s \in S \backslash S^*$, denote $y(\cdot, t) = \Sigma_{s \in S \backslash S^*} y(s,t)$ and $y(s, \cdot) = \Sigma_{t \in S^*} y(s,t)$. A facility node $t \in S^*$ is said to be *weak* if $\sum_{s \in K(t)} y(s,t) > \frac{1}{2} y(\cdot, t)$, and *strong* otherwise.

Let us concentrate on any given sub-tree of depth at most 2 rooted at $t \in S^*$, which consists of t, $K(t)$ and $G(t) = \bigcup_{s \in K(t)} K(s)$. A node $s \in K(t)$ is said to be *heavy* if $y(s,t) > \frac{1}{2} y(\cdot, t)$, and *light* otherwise. A light node $s \in K(t)$ is said to be *dominant* if $y(s,t) \geq \frac{1}{2} y(s, \cdot)$, and *non-dominant* otherwise. The set $K(t)$ is partitioned into three subsets: the set $H(t)$ of heavy nodes, the set $Dom(t)$ of light and dominant nodes and the set $NDom(t)$ of light and non-dominant nodes (see Figure 1 for an illustration). It is clear that if $t \in S^*$ is strong, then $H(t)$ must be empty.

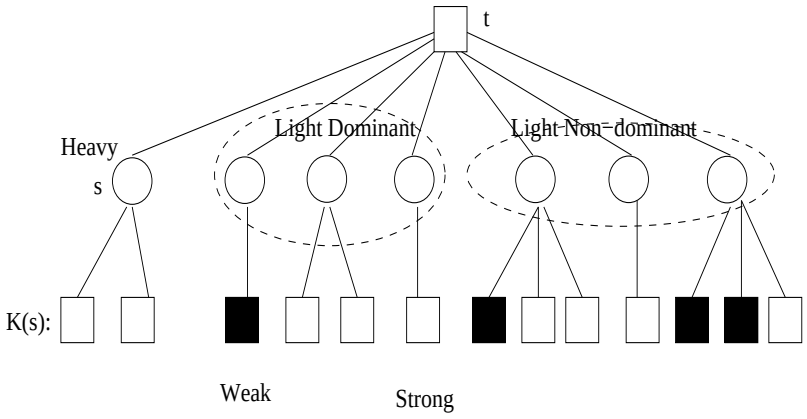

Figure 1: Facility classification

We are to present a list of legitimate operations such that (i) each facility in $S \backslash S^*$ is closed exactly once, (ii) every facility in S^* is opened a small number of times, and (iii) the total reassignment cost is not too high. Depending on their types, the nodes in $S \backslash S^*$ and S^* will be involved in different operations.

Let us first deal with the nodes in $NDom(t)$. For each node $s \in NDom(t)$, let $W(s) = \{t' \in K(s) : t' \text{ is week}\}$ and

$$Rem(s) = \max \left\{ y(s,t) - \sum_{t' \in W(s)} y(s,t'), \quad 0 \right\}.$$

Re-index the nodes in $NDom(t)$ as $s_1, s_2, \cdots, s_k$ for some $k \geq 0$ ($NDom(t) = \emptyset$ if $k = 0$), such that $Rem(s_i) \leq Rem(s_{i+1})$ for $i = 1, \ldots, k-1$. The operations given below will close s_i exactly once for $1 \leq i < k$. We will consider node s_k together with nodes in $Dom(t)$.

We perform operation $\mathbf{close}(s_i, K(s_i) \cup (K(s_{i+1}) \backslash W(s_{i+1})))$ for any $i = 1, \cdots, k-1$. The feasibility and costs of these operations are established in the following lemma.

Lemma 4. *For $i = 1, 2, \cdots, k-1$, the following hold:*

(i) Operation $\mathbf{close}(s_i, K(s_i) \cup (K(s_{i+1}) \backslash W(s_{i+1})))$ is feasible, i.e., the total capacity of facilities opened by the operation is at least the total capacity of those closed by the operation:

$$y(s_i, \cdot) \leq \sum_{t' \in K(s_i) \cup (K(s_{i+1}) \backslash W(s_{i+1}))} y(\cdot, t').$$

(ii) The cost of operation $\mathbf{close}(s_i, K(s_i) \cup (K(s_{i+1}) \backslash W(s_{i+1})))$ is at most

$$\begin{aligned} & c(s_i, t) y(s_i, t) + \sum_{t' \in W(s_i)} 2c(s_i, t') y(s_i, t') + \sum_{t' \in K(s_i) \backslash W(s_i)} c(s_i, t') y(s_i, t') \\ & + c(s_{i+1}, t) y(s_{i+1}, t) + \sum_{t' \in K(s_{i+1}) \backslash W(s_{i+1})} c(s_{i+1}, t') y(s_{i+1}, t'). \end{aligned}$$

Proof. As illustrated in Figure 2, we note that

$$y(s_i, \cdot) = y(s_i, t) + \sum_{t' \in W(s_i)} y(s_i, t') + \sum_{t' \in K(s_i) \backslash W(s_i)} y(s_i, t') = Y_1 + Y_2 + Y_3,$$

where $Y_1 = y(s_i, t) - \sum_{t' \in W(s_i)} y(s_i, t')$, $Y_2 = 2 \sum_{t' \in W(s_i)} y(s_i, t')$ and $Y_3 = \sum_{t' \in K(s_i) \backslash W(s_i)} y(s_i, t')$. We have $Y_2 \leq \sum_{t' \in W(s_i)} y(\cdot, t')$ since $W(s_i)$ is a set of weak nodes, and

$$Y_1 \leq Rem(s_i) \leq Rem(s_{i+1})) \leq \sum_{t' \in K(s_{i+1}) \backslash W(s_{i+1})} y(s_{i+1}, t'),$$

where the last inequality holds because s_{i+1} is non-dominant. Therefore,

$$Y_1 + Y_2 + Y_3 \leq \sum_{t' \in K(s_i) \cup (K(s_{i+1}) \backslash W(s_{i+1}))} y(\cdot, t')$$

The proof of (i) has suggested a way to reassign the remands from s_i to $K(s_i) \cup (K(s_{i+1}) \backslash W(s_{i+1}))$ as follows: Reassign $y(s_i, t')$ amount of demand to t' for any $t' \in K(s_i) \backslash W(s_i)$ and the cost of reassigning each unit is at most $c(s_i, t')$; reassign $Rem(s_i)$ to $t' \in K(s_{i+1}) \backslash W(s_{i+1})$ and the cost of

reassigning each unit is at most $c(s_i, t) + c(t, s_{i+1}) + c(s_{i+1}, t')$ by triangle inequality; and reassign to $W(s_i)$ the remaining demands, which is at most $2\sum_{t' \in W(s_i)} y(s_i, t')$, and the cost of reassigning one unit to t' is at most $c(s_i, t')$. Then (ii) follows easily. ∎

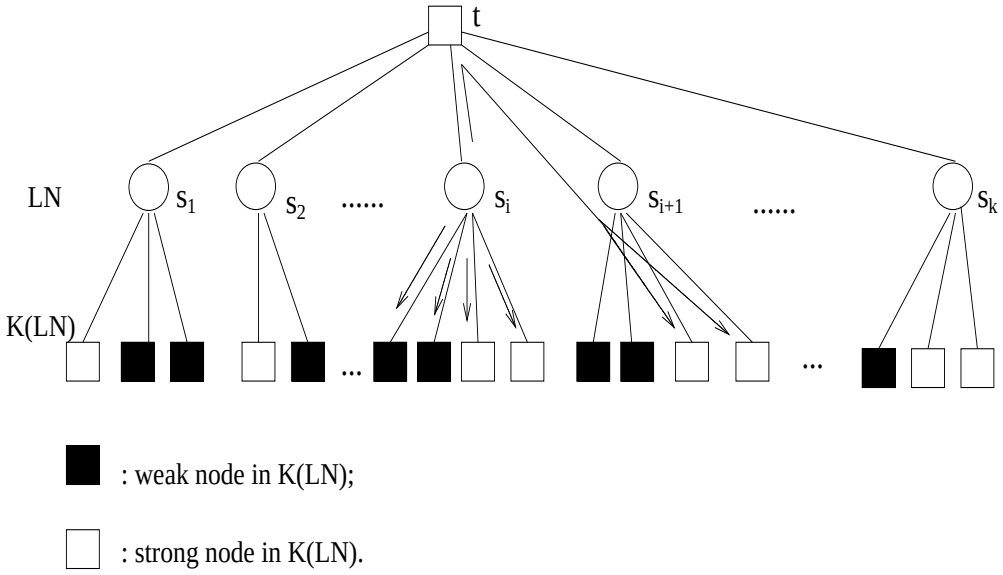

Figure 2: Illustration of the proof of Lemma 4

Now we move to considering nodes in $H(t)$, $Dom(t)$ and the node s_k. To do this, we need to distinguish several cases on t and $H(t)$.

Case 1. Node t is strong (see Figure 3 for illustration). This is an easy case. As we observed before, we have $H(t) = \emptyset$. Perform operation **multi**$(s_k, Dom(t), t, K(s_k))$. A feasible reassignment to t of demands served by s_k and facilities in $Dom(t)$ and $K(s_k)$ is as follows: for each $s \in Dom(t)$, reassign $y(s, \cdot)$ amount of demand to t; for s_k, reassign $y(s_k, t)$ amount of demand to t and $y(s_k, t')$ amount of demand to $t' \in K(s_k)$. To show that the operation is feasible, it suffices to show that $\sum_{s \in Dom(t)} y(s, \cdot) + y(s_k, t) \leq y(\cdot, t)$ and $y(s_k, t') \leq y(\cdot, t')$ for any $t' \in K(s_k)$. The second is trivial and the first holds since $s \in Dom(t)$ and hence

$$\sum_{s \in Dom(t)} y(s, \cdot) + y(s_k, t) \leq 2 \sum_{s \in Dom(t)} y(s, t) + y(s_k, t) \leq y(\cdot, t).$$

This leads to the following lemma.

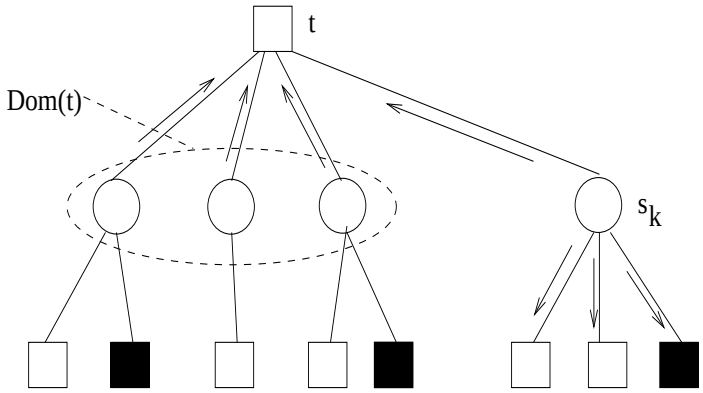

Figure 3: Illustration of Case 1

Lemma 5. *If t is strong, then operation* **multi**$(s_k, Dom(t), t, K(s_k))$ *is feasible, and the cost of this operation is at most*

$$\sum_{s\in \mathrm{Dom}(t)} 2c(s,t)y(s,t) + c(s_k,t)y(s_k,t) + \sum_{t'\in K(s_k)} c(s_k,t')y(s_k,t').$$

Case 2. Node t is weak and $H(t) \neq \emptyset$ (see Figure 4 for illustration). Since there can be no more than two heavy nodes, let $H(t) = \{s_0\}$. We perform operations **close**$(s_0, \{t\} \cup K(s_0))$ and **multi**$(s_k, Dom(t), t, K(s_k))$. For operation **multi**$(s_k, Dom(t), t, K(s_k))$, the reassignment of demands is done exactly as in Case 1. For operations **close**$(s_0, \{t\} \cup K(s_0))$, we reassign $y(s_0, t)$ amount of demand to t and $y(s_0, t')$ amount of demand to $t' \in K(s_0)$, which is obviously feasible. Then we have

Lemma 6. *If node t is weak and $H(t) = \{s_0\}$, then*

(i) Operation **multi**$(s_k, \mathrm{Dom}(t), t, K(s_k))$ *is feasible and the cost of this operation is at most*

$$\sum_{s\in \mathrm{Dom}(t)} 2c(s,t)y(s,t) + c(s_k,t)y(s_k,t) + \sum_{t'\in K(s_k)} c(s_k,t')y(s_k,t').$$

(ii) Operation **close**$(s_0, \{t\} \cup K(s_0))$ *is feasible and its cost is at most*

$$\sum_{t'\in K(s_0)} c(s_0,t')y(s_0,t') + c(s_0,t)y(s_0,t).$$

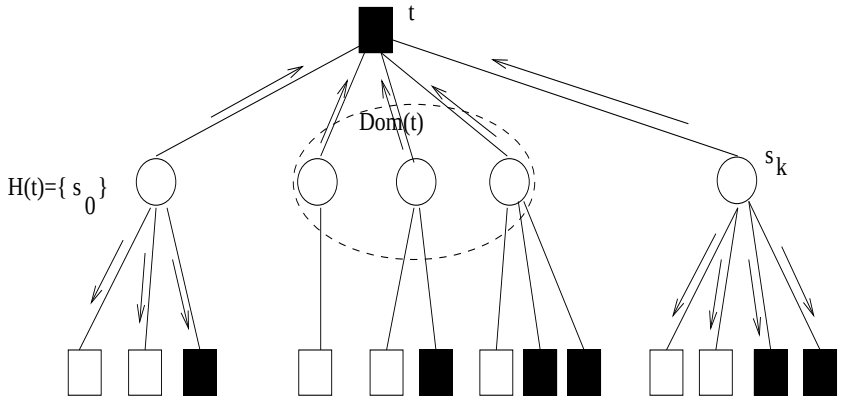

Figure 4: Illustration of Case 2

Case 3. Node t is weak and $H(t) = \emptyset$ (see Figure 5 for illustration). The analysis for this last case is more involved. In order to find feasible operations, we need the following lemma, where we make use of techniques from the area of parallel machine scheduling.

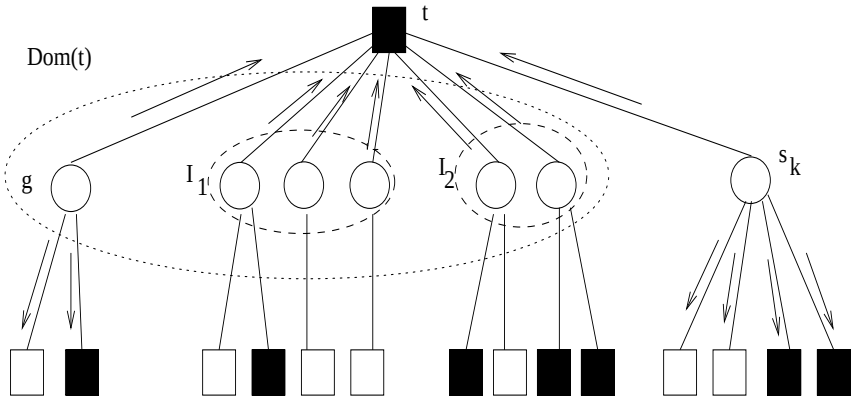

Figure 5: Illustration of Case 3 ($g = \gamma_1$ in the proof of Lemma 7).

Lemma 7. *If* $\mathrm{Dom}(t) \neq \emptyset$*, then there exists a node* $\gamma_1 \in \mathrm{Dom}(t)$ *such that set* $\mathrm{Dom}(t)\backslash\{\gamma_1\}$ *can be partitioned into two subsets* D_1 *and* D_2 *that satisfy the following:*

$$y(\gamma_i, t) + \sum_{s \in D_i} y(s, \cdot) \leq y(\cdot, t) \text{ for } i = 1, 2,$$

where $\gamma_2 = s_k$.

Proof. If $Dom(t)$ contains only one node, then we are done. We assume that $Dom(t)$ contains at least two nodes. First we note that

$$\sum_{s\in Dom(t)} 2y(s,t) + 2y(s_k,t) \le 2y(\cdot,t). \tag{8}$$

Initially set $I_1 := I_2 := \emptyset$. Let $v(I_i) = \sum_{s\in I_i} 2y(s,t)$. Arrange the nodes in $Dom(t)\cup\{s_k\}$ in a list in non-increasing order of $y(s,t)$. Iteratively add the first unassigned node in the list to I_1 or I_2 whichever has the smaller value of $v(I_i)$ until all the nodes in the list are assigned. Based on (8), we assume without loss of generality that $\Sigma_{s\in I_1} 2y(s,t) \ge y(\cdot,t)$ and $\sum_{s\in I_2} 2y(s,t) \le y(\cdot,t)$. Let ϕ be the node last assigned to I_1. It follows from (8) that

$$\min\left\{\sum_{s\in I_1\backslash\{\phi\}} 2y(s,t) + y(\phi,t), \sum_{s\in I_2} 2y(s,t) + y(\phi,t)\right\} \le y(\cdot,t).$$

However, according to the way the assignment has been conducted, we have $v(I_1\backslash\{\phi\}) \le v(I_2)$, i.e., $\sum_{s\in I_1\backslash\{\phi\}} 2y(s,t) \le \sum_{s\in I_2} 2y(s,t)$. Therefore, the above displayed inequality implies:

$$\sum_{s\in I_1\backslash\{\phi\}} 2y(s,t) + y(\phi,t) \le y(\cdot,t). \tag{9}$$

If $s_k \in I_2$, then we let $\gamma_1 = \phi$, $D_1 = I_1\backslash\{\phi\}$ and $D_2 = I_2\backslash\{s_k\}$. Then, since any node $s \in Dom(t)$ is dominant,

$$y(s_k,t) + \sum_{s\in D_2} y(s,\cdot) \le y(s_k,t) + \sum_{s\in D_2} 2y(s,t) \le \sum_{s\in I_2} 2y(s,t) \le y(\cdot,t).$$

On the other hand, it follows from (9) that

$$y(\gamma_1,t) + \sum_{s\in D_1} y(s,\cdot) \le y(\gamma_1,t) + \sum_{s\in D_1} 2y(s,t) \le y(\cdot,t).$$

If $s_k \in I_1$, then let γ_1 be any node in I_2, let $D_2 = I_1\backslash\{s_k\}$ and $D_1 = I_2\backslash\{\gamma_1\}$. Then,

$$y(\gamma_1,t) + \sum_{s\in D_1} y(s,\cdot) \le y(\gamma_1,t) + \sum_{s\in D_1} 2y(s,t) \le \sum_{s\in I_2} 2y(s,t) \le y(\cdot,t).$$

Since $s_k \in I_1$ and ϕ is the node last assigned to I_1, we have $y(s_k,t) \ge y(\phi,t)$. Hence,

$$\begin{aligned} y(s_k,t) + \sum_{s\in D_2} y(s,\cdot) &\le y(s_k,t) + \sum_{s\in D_2} 2y(s,t) \\ &= \sum_{s\in I_1} 2y(s,t) - y(s_k,t) \le \sum_{s\in I_1} 2y(s,t) - y(\phi,t). \end{aligned}$$

Since the last term above is equal to $\sum_{s\in I_1\setminus\{\phi\}} 2y(s,t)+y(\phi,t)$, the desired inequality follows from (9), which completes our proof of the lemma. ∎

Lemma 8. *Let γ_i and D_i $(i=1,2)$ be those defined in Lemma 7. If node t is weak and $H(t)=\emptyset$, then operations* **multi**$(\gamma_i, D_i, t, K(\gamma_i))$ *$(i=1,2)$ are feasible, and their costs are respectively at most*

$$\sum_{s\in D_i} 2c(s,t)y(s,t)+c(\gamma_i,t)y(\gamma_i,t)+\sum_{t'\in K(\gamma_i)} c(\gamma_i,t')y(\gamma_i,t')$$

for $i=1,2$.

Now we are ready to provide a global picture of the impact of all the operations and demand reassignment we have introduced, which will help bound the total facility cost of a local optimal solution.

Lemma 9. *Given any $r\in S^*$, the operations we defined on sub-tree $\mathcal{T}_r$ and the corresponding reassignment of demands satisfy the following two conditions: (i) Each facility $s\in S\setminus S^*$ is closed exactly once; (ii) Every facility $t\in S^*$ is opened at most 3 times; (iii) The total reassignment cost of all the operations is at most $2\sum_{s\in S\setminus S^*}\sum_{t\in S^*} c(s,t)y(s,t)$.*

Proof. The bounds on the total reassignment cost follow from Lemmas 4,5,6 and 8. We show that every facility $t\in S^*$ is opened at most 3 times. Note that when we take all sub-trees of $\mathcal{T}_r$ of depth 2 into account, any $t\in S^*$ can be the root or a leaf of such a sub-tree for at most once. If $t\in S^*$ is strong, it will be opened once as a root and twice as a leaf. If t is weak, it will be opened at most once as a leaf and at most twice as a root. ∎

Theorem 2. $C(S)\le 6C_f(S^*)+5C_s(S^*)$.

Proof. Since S is locally optimal, none of the above defined operations has a negative cost. Therefore, the total cost will also be non-negative. It follows from Lemma 9 that

$$3C_f(S^*\setminus S)+2\sum_{s\in S\setminus S^*}\sum_{t\in S^*} c(s,t)y(s,t)-C_f(S\setminus S^*)\ge 0.$$

The theorem follows from Lemmas 2 and 3. ∎

Scaling the service costs c_{ij} by a factor $\delta = (1+\sqrt{2})/2$ and with Lemma 2 and Theorem 2, we can show that $C(S) \leq (3+2\sqrt{2})C(S^*)$ which, together with Theorem 1, leads to the following corollary.

Corollary 1. *For any small constant $\epsilon > 0$, the multi-exchange local search algorithm runs in polynomial time with approximation guarantee of $3+2\sqrt{2}+\epsilon \leq 5.83$.*

4 A tight example

Figure 6 illustrates an example that gives a lower bound of our algorithm. In this example, we have a total of $4n$ clients that are represented by triangles, of which $2n$ have demands $n-1$ each and the other $2n$ have demands 1 each. There are a total of $4n+1$ facilities. In the current solution, the $2n$ open facilities are represented by squares, each of which has facility cost 4 and capacity n. The other $2n+1$ facilities are represented by circles, of which the one at the top of the figure has facility cost 4 and capacity $2n$, and each of the other $2n$ facilities at the bottom of the figure has facility cost 0 and capacity $n-1$. The numbers by the edges represent the unit service costs scaled by the factor of $\delta = (1+\sqrt{2})/2$. Other unit service costs are given by the shortest distances induced by the given edges. We will call a facility *circle-facility* or *square-facility* depending on how it is represented in the figure.

We show that the current solution is a local optimum, i.e., none of the four operations will have a negative cost. First of all, for **add** operation, it is easy to see that open any of the circle-facilities would not reduce the total service cost. Secondly, notice that the connection cost between a square-facility and a circle-facility is 0, 2, or 4. Therefore, closing any of the square-facilities would require opening the top circle-facility, and hence any **close** operation will not have a negative cost. Thirdly, we cannot open one bottom circle-facility while closing any of the square-facilities due to the capacity constraint. Therefore, in order to open one facility and close one or many facilities, we have to open the top circle-facility. In this case, we could possibly close any two of the square-facilities. However, routing one unit of demand from any square-facility to the top circle-facility will have a cost of 2. Therefore, any **open** operation will not be profitable. Finally, we show that the restricted multi-exchange operation **multi**(r, R, t, T) is not profitable either. We have already shown that this is the case $R = \emptyset$ or $T = \emptyset$. Therefore, we assume that $R \neq \emptyset$ and $T \neq \emptyset$. By

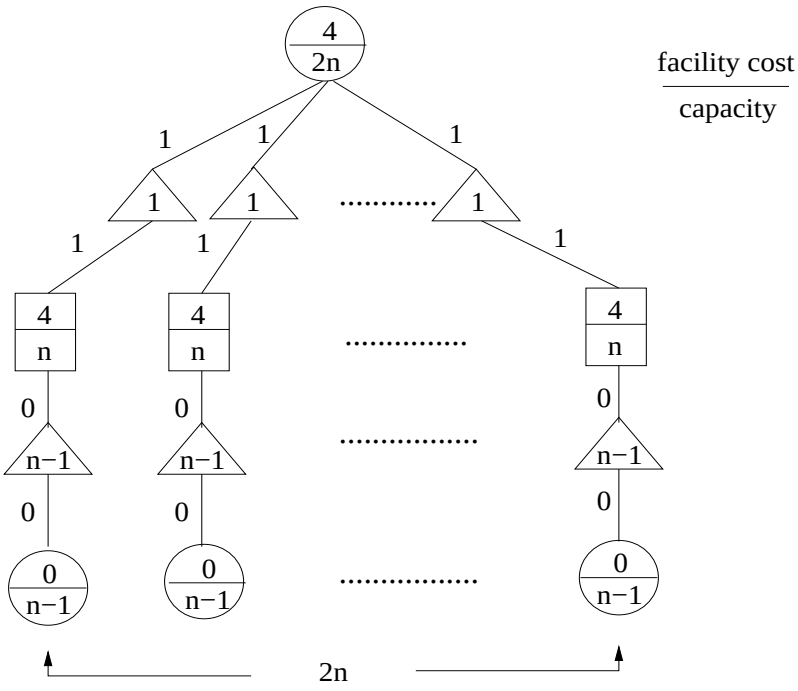

Figure 6: A tight example with performance ratio $3 + 2\sqrt{2} - \epsilon$

symmetry, we can assume r is any of the square-facilities. Since the demands served by R will be routed to t, capacity constraint forces t to be the top circle-facility. By comparing the capacities of the top circle-facility and the square-facilities, we know that $|R| \leq 2$. Now our final claim follows from the fact that routing one unit of demand from the square-facility to the circle-facility will cost 2. Therefore, the current solution is indeed a local optimum.

The current solution has a total facility cost of $8n$ and a total service cost of $2n/\delta = 4n/(1+\sqrt{2})$ in terms of the original (pre-scaling) data. However, if we open all the circle-facilities only, then the total facility cost is 4 and the total service cost is $4n/(1 + \sqrt{2})$. The total costs of these two solutions are $4n(3 + 2\sqrt{2})/(1 + \sqrt{2})$ and $4n/(1 + \sqrt{2}) + 4$, respectively, and their ratio approaches $3 + 2\sqrt{2}$ when n goes to ∞, which provides the desired lower bound of our algorithm.

5 Final remarks

A new local search algorithm for the CFLP has been proposed in this paper. The performance guarantee of the algorithm is shown to be between $3 + 2\sqrt{2} - \epsilon$ and $3 + 2\sqrt{2} + \epsilon$ for any constant $\epsilon > 0$. This is the currently best known approximation ratio for the CFLP. Moreover, it is the first

time that a tight bound of a local search algorithm for the CFLP is developed. Our result is based not only on the introduction of a new type of multi-exchange local improvement, but also on the exploitation of a technique from the area of parallel machine scheduling.

Several significant questions remain open. It is known that the UFLP can not be approximated better than 1.463 in polynomial time and this lower bound naturally also applies to the more general CFLP. But can better (larger) lower bound be established for the CFLP? On the other hand, it would also be challenging to improve the upper bound $3 + 2\sqrt{2} + \epsilon$. In order to obtain a better approximation ratio, it seems that new local improvement procedures are needed. Furthermore, it is known that the natural linear programming relaxation has an unbounded integrality gap. It would be interesting to develop new LP relaxations for the CFLP such that the known techniques for the UFLP such as randomized rounding, primal-dual, and dual-fitting can be applied to the CFLP.

Many other variants of the UFLP have been studied in the literature, e.g., k-median (Arya et al 2001), multi-level facility location (Aardal, Chuak, and Shmoys 1999, Ageev, Ye, and Zhang 2003, Zhang 2004), fault-tolerant facility location (Guha, Meyerson, and Munagala 2001, Swamy and Shmoys 2003), facility location with outlier (Charikar et al 2001). No constant approximation ratio is known for any of these problems with capacity constraints. Since the presence of capacity constraints is natural in practice, it would be important to study all of these problems with capacity constraints.

References

[1] K. Aardal, F. A. Chudak and D. B. Shmoys, "A 3-Approximation Algorithm for the k-Level Uncapacitated Facility Location Problem,". *Information Processing Letters* 72(5-6): 161-167, 1999.

[2] A. Ageev, Y. Ye and J. Zhang, "Improved Combinatorial Approximation Algorithms for the k-Level Facility Location Problem," in *Proceedings of the 30th International Colloquium on Automata, Languages and Programming (30th ICALP)*, LNCS 2719, 145-156, 2003.

[3] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala and V. Pandit, "Local search

heuristic for k-median and facility location problems," in *Proceedings of the ACM Symposium on Theory of Computing*, 21-29, 2001.

[4] M. Charikar and S. Guha, "Improved combinatorial algorithms for facility location and k-median problems," in *Proceedings of the 40th IEEE Foundations of Computer Science*, 378-388, 1999.

[5] M. Charikar, S. Khuller, D. M. Mount and G. Narasimhan, "Algorithms for facility location problems with outliers," in *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, 642-651, 2001.

[6] G. Cornuéjols, G.L. Nemhauser and L.A. Wolsey, "The uncapacitated facility location problem," in: P. Mirchandani, R. Francis (Eds.), *Discrete Location Theory*, Wiley, New York, 119-171, 1990.

[7] F.A. Chudak and D.B Shmoys,"Improved approximation algorithms for the uncapacitated facility location problem," *SIAM J. on Computing*, to appear.

[8] F. A. Chudak and D. Williamson, "Improved approximation algorithms for capacitated facility location problems," in *Proceedings of the 7th Conference on Integer Programming and Combinatorial Optimization (IPCO'99)*, 99-113, 1999.

[9] S. Guha and S. Khuller, "Greedy strikes back: improved facility location algorithms," *Journal of Algorithms,* 228-248, 1999.

[10] S. Guha, A. Meyerson and K. Munagala, "Improved algorithms for fault tolerant facility location," in *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, 636-641, 2001.

[11] M. R. Korupolu, C. G. Plaxton and R. Rajaraman, "Analysis of a Local Search Heuristic for Facility Location Problems," *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms,* 1-10, 1998.

[12] A. A. Kuehn and M. J. Hamburger, "A heuristic program for locating warehouses," *Management Science*, 9:643–666, 1963.

[13] K. Jain, M. Mahdian, and A. Saberi, "A new greedy approach for facility location problems," in *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, 731-740, 2002.

[14] M. Mahdian and M. Pál, "Universal facility location," in *ESA 2003*, to appear.

[15] M. Mahdian, Y. Ye and J. Zhang, "Improved approximation algorithms for metric facility location problems," in *Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, LNCS 2462, 229-242, 2002.

[16] M. Mahdian, Y. Ye and J. Zhang, "A 2-approximation algorithm for the soft-capacitated facility location problem," *Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, LNCS 2764, 129-140, 2003.

[17] M. Pál, É. Tardos and T. Wexler, "Facility location with hard capacities," in *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, 329-338, 2001.

[18] D. B. Shmoys, "Approximation algorithms for facility location problems," *3rd International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, 27-33, 2000.

[19] C. Swamy and D.B. Shmoys, "Fault-tolerant facility location," in *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, 735-736, 2003.

[20] D.B. Shmoys, É. Tardos, and K.I. Aardal, "Approximation algorithms for facility location problems," in *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 265–274, 1997.

[21] J. Zhang, "Approximating the two-level facility location problem via a quasi-greedy approach," to appear in *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2004.