The Fixed-Hub Single Allocation Problem: A Geometric Rounding Approach

Dongdong Ge*, Yinyu Ye*, Jiawei Zhang †

October 19, 2007

Abstract

This paper discusses the fixed-hub single allocation problem. In the model hubs are fixed and fully connected; and each terminal node is connected to a single hub which routes all its traffic. The goal is to minimize the cost of routing the traffic in the network. This paper presents linear programming-based algorithms that deliver both high quality solutions and a theoretical worst case bound. Computational results indicate that our algorithms solve large-sized problems efficiently. The algorithms are based on a new randomized dependent rounding method, a geometric rounding, which might be of interest on its own.

Key words: hub location; network design; linear programming; worst case analysis

1. Introduction

Hub-and-spoke networks have been widely used in transportation, logistics, and telecommunication systems. In such networks, traffic is routed from numerous nodes of origin to specific destinations through hub facilities. The use of hub facilities allows for the replacement of direct connections between all nodes with fewer, indirect connections. One main benefit is the economies of scale as a result of the consolidation of flows on relatively few arcs connecting the nodes. In the United States, hub-and-spoke routing is practically universal. Airlines adopted it after the industry was deregulated in 1978. Many logistics service providers such as UPS and Federal Express also have distribution systems using hub-and-spoke structure.

*This research is supported by the Boeing Company. Email: {dongdong, yinyu-ye}@stanford.edu

†Stern School of Business, IOMS-Operations Management, New York University, New York, NY, 10012. Email: jzhang@stern.nyu.edu

Given its widespread use, it is of practical importance to design efficient hub-and-spoke networks. In the literature, such problems are often referred to as hub location problems, in which two major questions need to be addressed: where hubs should be located and how the traffic/flow (be it passengers in transportation, packages in logistics, and communication packets) should be routed.

One important hub location problem is called the p-hub median problem. In this problem, the objective is to locate p hubs in a network and allocate non-hub nodes to hub nodes such that the sum of the costs of transporting flows between all origin-destination pairs in the network is minimized. In 1987, O'Kelly [22] proposed a quadratic integer program for the p-hub median problem. Two primary heuristics, along with the applications to air transportation instances, were also reported by O'Kelly [22] to compute upper bounds of the objective function value. Later on, Klincewicz [18] developed an exchange heuristic and a clustering heuristic using a multi-criteria distance and flow-based allocation procedure. Campbell [7] proposed a greedy exchange heuristic for the p-hub median problem with multiple allocations and two heuristics for the single-allocation problem based on flow information. An efficient tabular search heuristic was suggested by Skorin-Kapov *et al.* [27].

The linearization of the quadratic model was also developed [6, 24, 23, 13, 14], and it can often generate integer solutions without forcing integrality for small-sized problems (up to 25 nodes). A common assumption of these papers is that each non-hub node is required to be assigned to exactly one hub. In this case, the problem is sometimes referred to as the p-hub median problem with single allocation.

Since the work of O'Kelly [22], the p-hub median problem and its variants have received substantial attention; see, for instance, [5, 21, 26, 12, 20]. An overview of research on the p-hub median problem and other hub location problems can be found in [8].

Very recently, Campbell *et al.* [9, 10] proposed the hub arc location problem where the question of interest is where hub arcs, each of which connects two hubs, should be located.

In both hub location problems and hub arc location problems, even when the locations of hubs and/or hub arcs are specified in advance, optimally assigning the non-hub nodes to the hub nodes is still a challenging task. We refer to this problem as the fixed-hub allocation problem. This problem, although only a sub-problem to the hub location or hub arc location problems, is of particular importance. First, in many practical situations, the locations of hubs are pre-determined and remain unchanged in the medium/long term. Second, the number of hubs can be relatively small, which makes it possible to enumerate all possible locations of the hubs. Further, solving the fixed-hub allocation problem efficiently would help us solve the hub location(or hub arc location) problem.

Therefore, we are concerned with the fixed-hub single allocation problem *(FHSAP)*. For convenience, we may also use the notation k-*FHSAP* when the number of hubs is k. The *FHSAP* is known to be NP-hard and is a special case of the quadratic assignment problem. Sohn and Park [28] showed that, although the 2-*FHSAP* is a min-cut problem and thus is polynomial-time solvable, the 3-*FHSAP* is NP-hard already.

In several aforementioned heuristics for the p-hub median problem, instances of the *FHSAP* need to be solved when different subsets of hubs are fixed. And they are solved heuristically, given the complexity of the *FHSAP*. For example, the better one of two heuristics by O'Kelly [22] assigns a city to its nearest hub or the second nearest hub, which enumerates exponentially many allocation combinations. In Campbell's paper [7], given an initial set of hub locations and flow information from the multiple allocations problem, the first heuristic assigns each city to the hub which routes its maximum flow, and the second heuristic assigns each city to a hub such that the total routing cost is minimized. Though the latter gives a tighter bound, it has to consider all possible single allocation combinations.

In this paper, we present a class of linear programming-based algorithms to tackle the *FHSAP*. Computational results show that our algorithms deliver high quality solutions that are very close to optimal. Further, our algorithms are capable of solving very large-scale problems in a reasonable amount of time. Equally important, we establish provable worst case bounds for our algorithms. We discuss our results in more details below.

The first step of our algorithms is to solve a linear programming (LP) relaxation of the *FHSAP*. A natural LP relaxation can be obtained from an LP formulation for the p-hub median problem suggested by Campbell [6]. This LP relaxation is extremely attractive. Skorin-Kapov *et al.* [27] improved this LP relaxation and reported that the modified version was very tight and output integral solutions automatically in 95% of the instances that they tested. However, the size of the LP relaxation is relative large and restricts its applications to large-scale problems. Therefore, in order to solve large-scale problems, we also make use of the LP relaxation presented by Ernst and Krishnamoorthy [13, 14]. The size of this LP is significantly smaller than that in [27, 6]. We further modify this formulation by adding additional flow constraints, which delivers a better lower bound for the $FHSAP$. We consider all three LP relaxations. Although some relaxations are tighter than others, all these LP relaxations often produce undesirable fractional solutions.

Therefore, the second step of our algorithms is to round fractional solutions to integral ones. The novelty of our algorithms is the introduction of a new type of randomized rounding method, which we call *geometric rounding*. Any optimal (fractional) solution of the LP relaxation falls in a simplex. By taking advantage of geometric properties of a simplex, we randomly round a fractional solution, which corresponds to a non-extreme point of the simplex, to an extreme point.

Our geometric rounding technique enables us to establish worst case bounds for our algorithms for certain LP relaxations. To the best our knowledge, no provable bound has been provided to any of the aforementioned heuristics. A polynomial-time *ρ-approximation algorithm* for a minimization problem is defined to be an algorithm that runs in polynomial time and outputs a solution with a cost at most $\rho(\geq 1)$ times the optimal cost. ρ is called *approximation ratio* or *performance guarantee*. We show that our algorithms (based on two of the LP relaxations) have an approximation ratio of 2 for the special case of *k-FHSAP* in which all hubs constitute an equilateral (i.e., distances between hubs are uniform), which leads to a data-dependent performance guarantee for the general *k-FHSAP*.

We consider the geometric rounding technique and its analysis our major contribution of this paper. We expect it will find more applications in designing efficient algorithms for solving other discrete optimization problems. Recently, Ge *et al.* [16] have found that a truthful mechanism in combinatorial auction based on the geometric rounding has the best theoretical performance guarantee for sparse auction games.

The results of the paper are organized as follows. In Section 2, we define the *FHSAP* and present its linear programming relaxations. Section 3 presents our geometric rounding method and its analysis. In Section 4, we prove worst case bounds for our algorithms. Computational results are presented in section 5. Section 6 concludes the paper.

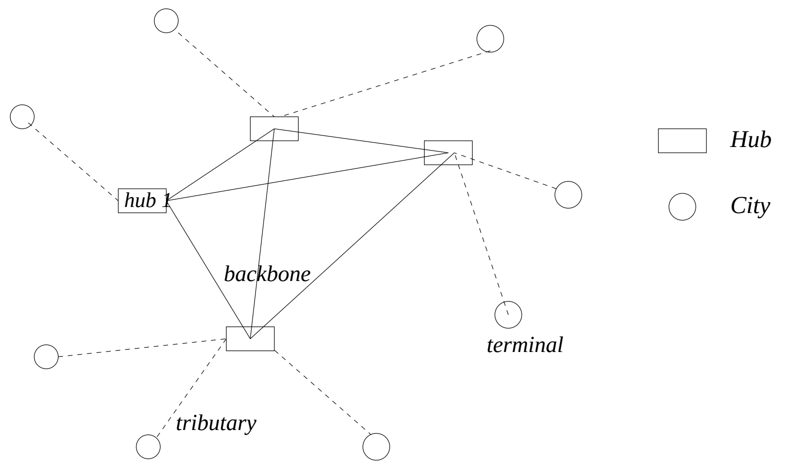

Figure 1: Each city is assigned to one single hub. Routing is done in the two-level network.

2. Problem Description and Formulation

This section defines the fixed-hub single allocation problem, reviews and modifies previously proposed mathematical programs. By the terminology of communication networks, the problem is to build a two-level network [11]; see figure 1. Hubs (airports, routers, concentrators, *etc.*) are transit nodes which route traffic. The network connecting hubs is called the *backbone* network. Terminal nodes (cities, computers, *etc.*) are called access nodes, and they represent the origins and the destinations of the traffic. The model can be described as a backbone/tributary network design problem in which backbone networks are fully connected and tributary networks are star-shape.

In order to route the demands between two terminal nodes, the original node has to deliver all its demands to the hub it is assigned to. Then this hub sends them to the hub the destination node is assigned to (this step is skipped if both nodes are assigned to the same hub). Finally the destination node gets the demands from its hub. No direct routing between two terminal nodes is permitted. Two types of costs are counted: the cost of routing between terminal nodes and transit nodes and the cost of routing between transit nodes. There are often economies of scale for inter-hub traffic.

O'kelly *et al.* [22] first formulated the uncapacitated single allocation p-hub median problem (*USApHMP*) as a quadratic integer program. We consider its adapted form for the *FHSAP*. Assume we are given a set of fixed hubs $\mathcal{H} = \{1, 2, \dots, k\}$ and a set of cities $\mathcal{C} = \{1, 2, \dots, n\}$. Directed demand d_{ij} to be routed from city i to city j is given. The distance from city i to hub s is c_{is}, which is also called the per unit transportation cost. Similarly define c_{st} to be the distance from hub s to hub t. Define $\vec{x} = \{x_{i,s} : i \in \mathcal{C}, s \in \mathcal{H}\}$ to be the assignment variables. The quadratic formulation for the *FHSAP* is then

Problem FHSAP-QP

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i,j \in \mathcal{C}} d_{ij} \left(\sum_{s \in \mathcal{H}} c_{is} x_{i,s} + \sum_{t \in \mathcal{H}} c_{jt} x_{j,t} + \sum_{s,t \in \mathcal{H}} \alpha c_{st} x_{i,s} x_{j,t} \right) & \\
\text{subject to} \quad & \sum_{s \in \mathcal{H}} x_{i,s} = 1, & \forall i \in \mathcal{C}, \\
& x_{i,s} \in \{0,1\}, & \forall i \in \mathcal{C}, s \in \mathcal{H}.
\end{aligned}
$$

All coefficients $d_{ij}, c_{is}, c_{jt}, c_{st} \geq 0$, and $c_{st} = c_{ts}$, $c_{ss} = 0$, $\forall i,j \in \mathcal{C}, \forall s,t \in \mathcal{H}$. α is the discount factor and $0 \leq \alpha \leq 1$. Without loss of generality, α can be assumed to be *one*. Note that the transportation cost from cities to hubs, $\sum_{i,j \in \mathcal{C}} d_{ij} (\sum_{s \in \mathcal{H}} c_{is} x_{i,s} + \sum_{t \in \mathcal{H}} c_{jt} x_{j,t})$, is linear in $\vec{x}$, we call it the *linear cost* of the objective function and denote it by $L(\vec{x})$. Similarly, call the other part of the objective function the *inter-hub cost* or *quadratic cost*, and denote it by $Q(\vec{x})$.

Campbell [6] linearized O'Kelly's model by formulating an alternative MILP for *USApHMP*. Its adapted form for the *FHSAP* can be formulated as follows:

Problem FHSAP-MILP1

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i,j \in \mathcal{C}} \sum_{s,t \in \mathcal{H}} d_{ij} (c_{is} + c_{st} + c_{jt}) X_{ijst} & \\
\text{subject to} \quad & \sum_{s,t \in \mathcal{H}} X_{ijst} = 1, & \forall i,j \in \mathcal{C}, \\
& \sum_{t \in \mathcal{H}} X_{ijst} = x_{i,s}, & \forall i,j \in \mathcal{C}, s \in \mathcal{H}, \\
& \sum_{s \in \mathcal{H}} X_{ijst} = x_{j,t}, & \forall i,j \in \mathcal{C}, t \in \mathcal{H}, \\
& X_{ijst} \geq 0, & \forall i,j \in \mathcal{C}, s,t \in \mathcal{H}, \\
& x_{i,s}, x_{j,t} \in \{0,1\}, & \forall i \in \mathcal{C}, s \in \mathcal{H}.
\end{aligned}
$$

Here X_{ijst} is the portion of the flow from city i to city j via hub s and t sequentially. The formulation involves $O(n^2k^2)$ nonnegative variables and $O(n^2k)$ constraints. This formulation enables us to obtain an LP relaxation for the *FHSAP* by replacing the zero-one constraints with non-negative constraints. We will refer to this LP relaxation as *FHSAP-LP1*. As we have mentioned in the introduction, this LP relaxation is very tight and often produces integer solutions. However, the size of the LP relaxation is relative large, which restricts its applications to large-sized problems.

In order to reduce the time complexity, we consider a flow formulation for the *FHSAP*, which is adapted from a formulation for the *USApHMP* proposed by Ernst and Krishnamoorthy [13, 14]. In this formulation, we do not have to specify the route for a pair of cities i and j, i.e., we do not need decision variable X_{ijst}. Instead, we define $\vec{Y} = \{Y^i_{st} : i \in \mathcal{C}, s, t \in \mathcal{H}, s \neq t\}$ where Y^i_{st} is the total amount of the flow originated from city i and routed from hub s to a different hub t. Define $O_i = \sum_{j\in\mathcal{C}} d_{ij}$; $D_i = \sum_{j\in\mathcal{C}} d_{ji}$. Then the *FHSAP* can be bounded from below by **Problem FHSAP-MILP2**

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{H}} c_{is}(O_i + D_i)x_{i,s} + \sum_{i\in\mathcal{C}}\sum_{s,t\in\mathcal{H}:s\neq t} c_{st}Y^i_{st} && && (1)\\
\text{subject to} \quad & \sum_{s\in\mathcal{H}} x_{i,s} = 1, && \forall i \in \mathcal{C}, && (2)\\
& \sum_{t\in\mathcal{H}:t\neq s} Y^i_{st} - \sum_{t\in\mathcal{H}:t\neq s} Y^i_{ts} = O_i x_{i,s} - \sum_{j\in\mathcal{C}} d_{ij}x_{j,s}, && \forall i \in \mathcal{C}, s \in \mathcal{H}, && (3)\\
& x_{i,s} \in \{0, 1\}, && \forall i \in \mathcal{C}, s \in \mathcal{H}, && (4)\\
& Y^i_{st} \geq 0, && i \in \mathcal{C}, s, t \in \mathcal{H}, s \neq t. && (5)
\end{aligned}
$$

Note that this modified formulation involves only $O(nk^2)$ nonnegative variables and $O(nk)$ linear constraints. In contrast to *FHSAP-MILP1*, the problem size is decreased by a factor n. We can then obtain an LP relaxation *FHSAP-LP2* for the *FHSAP* from the formulation *FHSAP-MILP2*.

A given feasible assignment $\vec{x}$ to the *FHSAP* with the flow vector $\vec{Y}$ is always a feasible solution to *FHSAP-MILP2*. The value of objective function of *FHSAP-MILP2* with this solution is equivalent to the transportation cost. Thus, *FHSAP-MILP2* only provides a lower bound for the *FHSAP*, and there can be a strictly positive gap between the optimal value of *FHSAP-MILP2* and that of the general *FHSAP*, as our simulation results indicate. However, it can be proved that *FHSAP-MILP2* is an exact formulation of *FHSAP* when hubs in the network constitute an equilateral.

It is possible to obtain a stronger LP relaxation than that of *FHSAP-MILP2* by adding a set of valid constraints, which is particularly useful in deriving the worst case bound for our rounding algorithm.

Lemma 1. *Let $\vec{x}$ and $\vec{Y}$ be defined as in Formulation* FHSAP-MILP2. *For any $i \in \mathcal{C}$ and $s \in \mathcal{H}$,*

$$\sum_{t\in\mathcal{H}:t\neq s} Y^i_{st} + \sum_{t\in\mathcal{H}:t\neq s} Y^i_{ts} = \sum_{j\in\mathcal{C}} d_{ij}|x_{i,s} - x_{j,s}|. \tag{6}$$

Proof. We verify equation (6) in two cases.

If $x_{i,s} = 0$, then

$$\sum_{t\in\mathcal{H}:t\neq s} Y^i_{st} + \sum_{t\in\mathcal{H}:t\neq s} Y^i_{ts} = \sum_{t\in\mathcal{H}:t\neq s} Y^i_{ts} = \sum_{j\in\mathcal{C}} d_{ij}x_{js} = \sum_{j\in\mathcal{C}} d_{ij}|x_{i,s} - x_{j,s}|.$$

If $x_{i,s} = 1$, then

$$\sum_{t\in\mathcal{H}:t\neq s} Y^i_{st} + \sum_{t\in\mathcal{H}:t\neq s} Y^i_{ts} = \sum_{t\in\mathcal{H}:t\neq s} Y^i_{st} = \sum_{j\in\mathcal{C}:x_{j,s}=0} d_{ij} = \sum_{j\in\mathcal{C}} d_{ij}(1 - x_{j,s}) = \sum_{j\in\mathcal{C}} d_{ij}|x_{i,s} - x_{j,s}|.$$

Therefore, equation (6) holds in both cases. □

In view of Lemma 1, we obtain a strengthened LP relaxation for the *FHSAP*, which we call *FHSAP-LP2'*, by adding the constraints generated from equation (6) to *FHSAP-LP1*.

Notice that if we sum up all the constraints from equation (6), we get a valid aggregate flow constraint:

$$2\sum_{i\in\mathcal{C}} \sum_{s,t\in\mathcal{H}:s\neq t} Y^i_{st} = \sum_{i,j\in\mathcal{C}} \sum_{s\in\mathcal{H}} d_{ij}|x_{i,s} - x_{j,s}|. \tag{7}$$

Replace all constraints from equation (6) in *FHSAP-LP2'* by the single constraint (7), we get a new LP formulation. We refer to it as *FHSAP-LP3*.

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{H}} c_{is}(O_i+D_i)x_{i,s} + \sum_{i\in\mathcal{C}}\sum_{s,t\in\mathcal{H}:s\neq t} c_{st}Y^i_{st} && && (8)\\
\text{subject to} \quad & \sum_{s\in\mathcal{H}} x_{i,s} = 1, && \forall i\in\mathcal{C}, && (9)\\
& \sum_{t\in\mathcal{H}:t\neq s} Y^i_{st} - \sum_{t\in\mathcal{H}:t\neq s} Y^i_{ts} = O_i x_{i,s} - \sum_{j\in\mathcal{C}} d_{ij}x_{j,s}, && \forall i\in\mathcal{C}, s\in\mathcal{H}, && (10)\\
& 2\sum_{i\in\mathcal{C}}\sum_{s,t\in\mathcal{H}:s\neq t} Y^i_{st} = \sum_{i,j\in\mathcal{C}}\sum_{s\in\mathcal{H}} d_{ij}y_{i,j,s}, && && (11)\\
& x_{i,s} - x_{j,s} \le y_{i,j,s}, && \forall i,j\in\mathcal{C}, s\in\mathcal{H}, && (12)\\
& x_{j,s} - x_{i,s} \le y_{i,j,s}, && \forall i,j\in\mathcal{C}, s\in\mathcal{H}, && (13)\\
& Y^i_{st}, x_{i,s}, y_{i,j,s} \ge 0, && i\in\mathcal{C}, s,t\in\mathcal{H}, s\neq t. && (14)
\end{aligned}
$$

The numbers of variables and constraints in *FHSAP-LP3* are both $O(n^2k+nk^2)$. Although it doesn't reduce the size of the formulation of *FHSAP-LP2'* significantly, computational results indicate that *FHSAP-LP3* reduces the running time remarkably at the minor expense of the effectiveness of the algorithm. More importantly, *FHSAP-LP3* is still sufficient for us to derive the worst case bound of our rounding algorithm.

In the next section, we discuss our rounding algorithm, that is, how to round fractional solutions of LP relaxations to integral ones.

3. Geometric Rounding

3.1 Rounding Procedure

Notice that a solution to the *FHSAP* can be completely defined by the assignment variable $\vec{x}$. After solving the LP relaxation *FHSAP-LP1*, *FHSAP-LP2* or *FHSAP-LP3*, we only need to focus on rounding the fractional assignment variables to binary integers. Notice that, in all three relaxations presented above, for a terminal node i, any optimal solution $x_i = (x_{i,1}, \ldots, x_{i,k})$ on node i must fall into a standard $k-1$ dimensional simplex:

$$\{w \in R^k | w \ge 0, \sum_{i=1}^{k} w_i = 1\}.$$

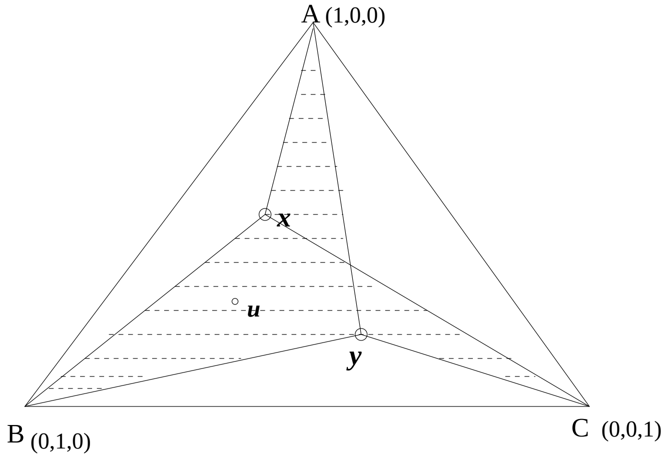

Figure 2: By the geometric rounding method, $\hat{x} = (1, 0, 0)$, $\hat{y} = (0, 0, 1)$ as the graph indicates.

We denote this simplex by Δ_k.

Therefore, a fractional assignment vector on node i corresponds to a non-vertex point in the simplex Δ_k. Our goal is to round any fractional solution to a vertex point of Δ_k, which is of the form:

$$(w \in R^k | w_i \in \{0, 1\}, \sum_{i=1}^{k} w_i = 1).$$

It is clear that Δ_k has exactly k vertices. We denote the vertices of Δ_k by $v_1, v_2, \cdots, v_k$, where the ith coordinate of v_i is 1.

Before presenting the rounding procedure, we will review some simple geometric concepts first. For a point $x \in \Delta_k$, connect x with all vertices $v_1, \ldots, v_k$ of Δ_k. Denote the polyhedron which exactly has vertices $\{x, v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_k\}$ by $A_{x,i}$. Thus simplex Δ_k can be partitioned into k polyhedrons $A_{x,1}, \ldots, A_{x,k}$, and the interiors of any distinct pair of these k polyhedrons do not intersect. Denote the volume of $A_{x,i}$ by $V_{x,i}$, and the volume of Δ_k by V_k.

We are now ready to present our randomized rounding algorithm. Notice that this rounding procedure is applicable to problems besides the *FHSAP*, as long as the feasible set of the problems is the set of vertices of a simplex.

Geometric Rounding Algorithm(FHSAP-GRA):

1. Solve an LP relaxation of the *FHSAP*: *LP1*, *LP2* or *LP3*, and get an optimal solution $\vec{x}^*$.

2. Generate a random vector u, which follows a uniform distribution in Δ_k.

3. For each $x_i^* = (x_{i,1}^*, \ldots, x_{i,k}^*)$, if u falls into $A_{x_i^*,s}$, let $\hat{x}_{i,s} = 1$; other components $\hat{x}_{i,t} = 0$.

Remark. There are several direct methods to generate a uniform random vector u from the standard simplex Δ_k. We choose the following method: generate k independent unit-exponential random numbers $a_1, ..., a_k$, i.e., $a_i \sim exp(1)$. Then vector u, whose ith coordinate is defined as

$$u_i = \frac{a_i}{\sum_{i=1}^k a_i},$$

is uniformly distributed in Δ_k.

Notice that in the geometric rounding algorithm, the random assignment choices the algorithm makes are highly dependent. Several (randomized) dependent rounding schemes have been proposed in the literature by Ageev and Sviridenko [1], Bertsimas *et al.* [2], Calinescu *et al.* [3], Gandhi *et al.* [15] and Srinivasan [29]. These dependent rounding methods have been successfully applied to tackled various combinatorial optimization problems. We also note that when $k = 2$, our geometric rounding method is exactly the same as the dependent rounding method proposed by Bertsimas *et al.* [2].

In the last step of our algorithm, we need to decide which polyhedron the generated point falls into. This can be easily done by using the following fact.

Lemma 2. *Given $w = (w_1, w_2, \ldots, w_k) \in \Delta_k$, vector u in Δ_k is in the interior of polyhedron $A_{w,s}$ only if s minimizes $\frac{u_l}{w_l}, 1 \leq l \leq k$.*

Proof. By symmetry we only need to discuss the case $s = 1$. If vector u falls into polyhedron $A_{w,1}$, vector u can be written as a convex combination of vertices of $A_{w,1}$. i.e., there exist nonnegative α_i's, such that $\sum_{i=1}^k \alpha_i = 1$ and $u = \alpha_1 w + \sum_{i=2}^k \alpha_i v_i$. It follows that

$$u_1 = \alpha_1 w_1, \quad u_i = \alpha_1 w_i + \alpha_i, \forall i \geq 2.$$

Then, for each $i \geq 2$,

$$\frac{u_i}{w_i} = \frac{\alpha_1 w_i + \alpha_i}{w_i} \geq \alpha_1 = \frac{u_1}{w_1}.$$

This completes the proof. $\square$

Thus, deciding which polyhedron the generated point falls into is an easy task if the index set $\arg\min_{1\le l\le k}\{\frac{u_l}{x^*_{i,l}}\}$ is a singleton. In case it is not, this can be done randomly, as it happens with probability zero if u is generated uniformly at random.

3.2 Analysis of Geometric Rounding

In this subsection, we prove several properties of the geometric rounding procedure, which are useful in establishing the performance guarantee of our algorithm. The proofs of these properties are established with the help of a few properties of exponential distribution, which we summarize below.

Lemma 3. *The following statements hold.*

- *Assume that $a_1, a_2, \cdots, a_k$ are k independent random variables with $a_i \sim exp(\lambda_i)$. Then for any $1 \le j \le k$,*
$$Prob(a_j = \min_{1\le i\le k} a_i) = \frac{\lambda_j}{\sum_{i=1}^k \lambda_i}.$$
- *If two random variables $Z \sim exp(\mu)$ and $W \sim exp(\lambda)$ are independent, then for any α and β with $0 \le \alpha \le \beta$,*
$$Prob(\alpha Z < W < \beta Z) = \mu(\frac{1}{\mu+\lambda\alpha} - \frac{1}{\mu+\lambda\beta}).$$

We are ready to prove our first result, which provides an exact estimation of the expected linear cost.

Theorem 4. *For any given $i \in \mathcal{C}, l \in \mathcal{H}$, $E[\hat{x}_{i,l}] = x^*_{i,l}$.*

Proof. For any $\hat{x}_{i,l}$, according to lemma 2,

$$E[\hat{x}_{i,l}] = Prob(\hat{x}_{i,l} = 1) = Prob(u \text{ falls into } A_{x^*_i,l}) = Prob(\frac{u_l}{x^*_{i,l}} = \min_{1\le t\le k}\{\frac{u_t}{x^*_{i,t}}\})$$

Recall that $u_i = \frac{a_i}{\sum_{j=1}^k a_j}$ and $a_i \sim exp(1)$ for any $1 \le i \le k$. This fact implies that $\frac{a_l}{x^*_{i,l}} \sim exp(x^*_{i,l})$. By Lemma 3, we have

$$Prob(\frac{u_l}{x^*_{i,l}} = \min_{1\le t\le k}\{\frac{u_t}{x^*_{i,t}}\}) = Prob(\frac{a_l}{x^*_{i,l}} = \min_{1\le t\le k}\{\frac{a_t}{x^*_{i,t}}\}) = \frac{x^*_{i,l}}{\sum_{t=1}^k x^*_{i,t}} = x^*_{i,l}.$$

This completes the proof. □

The following theorem states that if two non-vertex points x and y are close in distance, then the rounded points $\hat{x}$ and $\hat{y}$ should not be too far from each other in expectation. One way to measure the distance of two points x and y is by the l_1 norm of $x - y$. For any x and y, define $d(x, y) := \sum_s |x_s - y_s|$. Then we have

Theorem 5. *For any $x, y \in \Delta_k$, if we randomly round x and y to vertices $\hat{x}$ and $\hat{y}$ in Δ_k by the procedure in* FHSAP-GRA*, then*

$$E[d(\hat{x}, \hat{y})] \leq 2d(x, y).$$

Proof. Rather than proving the theorem directly, we prove an equivalent claim: *For any $0 \leq m \leq k$, if x and y have the same values on m corresponding coordinates, then $E[d(\hat{x}, \hat{y})] \leq 2d(x, y)$.* This claim can be proved by induction on m.

If $m = k$, then $x = y$. With probability 1, x and y will be rounded to the same vertex. Therefore, $E[d(\hat{x}, \hat{y})] = 0$ as well. Thus, the desired claim holds in this case.

Assume the claim holds for $m = k, k-1, \cdots, m'+1, m'$, where $m' \geq 1$. Now we consider the case where x and y have the same values on $m = m' - 1$ corresponding coordinates. Without loss of generality, assume $\frac{x_1}{y_1} \geq \frac{x_2}{y_2} \geq \cdots \geq \frac{x_k}{y_k}$ (define $\frac{x_i}{y_i} = +\infty$ if $y_i = 0$). Because $\sum_i x_i = \sum_i y_i = 1$, $x_i, y_i \geq 0$, we must have $\frac{x_1}{y_1} > 1 > \frac{x_k}{y_k}$ assuming $x \neq y$.

We first consider the case in which both x_k and y_1 are nonzero. For any $s : 0 < s < 1$ let

$$t = s + \frac{1-s}{y_1}, \quad r = s + \frac{1-s}{x_k}.$$

Further, we define two new points

$$x(s) = (x_1 s, x_2 s, \cdots, x_{k-1} s, x_k r),$$

$$y(s) = (y_1 t, y_2 s, \cdots, y_{k-1} s, y_k s).$$

Notice that $s = 0$ implies $x_1 s < y_1 t$, $s = 1$ implies $x_1 s > y_1 t$, and r, t increase as s decreases, so there exists $0 < s < 1$, such that $x_1 s = y_1 t$. Similarly, we can find $0 < s' < 1$, such that $x_k r' = y_k s'$. In the following proof, assume $s \geq s'$ (the case $s \leq s'$ can be handled similarly). Then we know $x_1 s = y_1 t$; $x_k r \leq x_k r' = y_k s' \leq y_k s$. This implies that $x(s)$ and $y(s)$ have the same values on m' corresponding coordinates.

Now we are ready to bound $E[d(\hat{x},\hat{y})]$. First, by the triangle inequality in the l_1 metric,

$$E[d(\hat{x},\hat{y})] \le E[d(\hat{x},\hat{x}(s)) + d(\hat{x}(s),\hat{y}(s)) + d(\hat{y}(s),\hat{y})].$$

From Lemma 6 below, we can show that $E[d(\hat{x},\hat{x}(s))] = d(x,x(s))$, and $E[d(\hat{y},\hat{y}(s))] = d(y,y(s))$. Further, by the assumption of the induction, and by the fact that $x(s)$ and $y(s)$ have the same values on m' corresponding coordinates, we know that

$$E[d(\hat{x}(s),\hat{y}(s))] \le 2d(x(s),y(s)).$$

Therefore, in order to show $E[d(\hat{x},\hat{y})] \le 2d(x,y)$, it is sufficient to prove the inequality:

$$d(x,x(s)) + 2d(x(s),y(s)) + d(y,y(s)) \le 2d(x,y),$$

or

$$d(x,x(s)) + d(y,y(s)) \le 2d(x,y) - 2d(x(s),y(s)).$$

By the definition of $d(x,y)$, the above inequality is equivalent to

$$2(r-1)x_k + 2(t-1)y_1 \le 2((1-s)\sum_{i=2}^{k-1}|x_i - y_i| + (x_1 - y_1) + (y_k - x_k) - (y_k s - x_k r)),$$

which can be further reduced to the following trivial inequality

$$0 \le (1-s)(x_1 + y_k + \sum_{i=2}^{k-1}|x_i - y_i|).$$

This completes the proof for the case both x_k and y_1 are non-zero. If x_k(or y_1 or both) is 0, replace the $x_k r$ (or $y_1 t$ or both) in the proof above with $1-s$. The proof is similar. □

Our proof of Theorem 5 has used a fact that is formalized in the following Lemma.

Lemma 6. *Assume $x, x(s) \in \Delta_k$, and $x(s) = (sx_1, sx_2, \ldots, sx_{k-1}, sx_k + (1-s))$, $0 < s < 1$. Then*

$$E[d(\hat{x},\hat{x}(s))] = d(x,x(s)).$$

Proof. By definition, $d(x,x(s)) = (1-s)\sum_{i=1}^{k-1} x_i + (r-1)x_k = 2(1-s)(1-x_k)$. If $d(\hat{x},\hat{x}(s)) \neq 0$, then $d(\hat{x},\hat{x}(s)) = 2$. Thus,

$$E[d(\hat{x},\hat{x}(s))] = 2 * Prob(d(\hat{x},\hat{x}(s)) \neq 0).$$

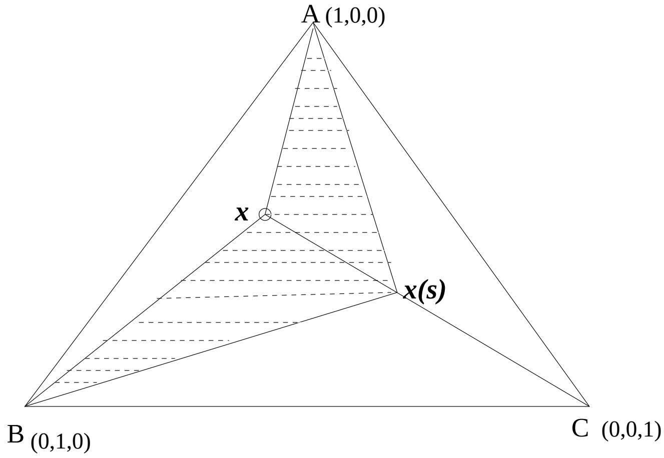

Figure 3: $v_k, x(s), x$ are collinear.

Notice that for any i, $1 \leq i \leq k-1$, $\frac{u_i}{x_i} \leq \frac{u_i}{x(s)_i}$. Then, Lemma 2 implies that, given vector u, if $x(s)$ is rounded to vertex v_i, $1 \leq i \leq k-1$, x must be rounded to the same vertex. It follows that the case $x(s)$ and x are rounded to two different vertices happens only when $x(s)$ is rounded to v_k and x is rounded to a different vertex. In view of Lemma 2, we have

$$\begin{aligned}
& Prob(d(\hat{x}, \hat{x}(s)) \neq 0) \\
= \; & Prob(\frac{u_k}{x(s)_k} < \min_{1 \leq i \leq k-1}\{\frac{u_i}{x(s)_i}\} \quad \text{and} \quad \frac{u_k}{x_k} > \min_{1 \leq i \leq k-1}\{\frac{u_i}{x_i}\}) \\
= \; & Prob(\frac{a_k}{x(s)_k} < \min_{1 \leq i \leq k-1}\{\frac{a_i}{x(s)_i}\} \quad \text{and} \quad \frac{a_k}{x_k} > \min_{1 \leq i \leq k-1}\{\frac{a_i}{x_i}\}),
\end{aligned}$$

where the last equality holds because $u_i = \frac{a_i}{\sum_{j=1}^{k} a_j}$ for any $1 \leq i \leq k$. If we define $Z = \min_{1 \leq i \leq k-1}\{\frac{a_i}{x_i}\}$ and $W = \frac{a_k}{x(s)_k}$, then it follows that

$$Prob(d(\hat{x}, \hat{x}(s)) \neq 0) = Prob(\alpha Z < W < \beta Z)$$

with $\alpha = \frac{x_k}{sx_k + (1-s)}$ and $\beta = \frac{1}{s}$. Recall that $a_i \sim exp(1)$ for any $1 \leq i \leq k$. Therefore,

$$Z \sim exp(x_1 + x_2 + \ldots + x_{k-1}) = exp(1 - x_k) \quad \text{and} \quad W \sim exp(sx_k + (1-s)).$$

By Lemma 3, we obtain that $Prob(\alpha Z < W < \beta Z) = (1-s)(1-x_k)$. Thus, $E[d(\hat{x}, \hat{x}(s))] = 2(1-s)(1-x_k)$. This completes the proof. $\square$

Remark. The bound proved in Theorem 5 is essentially tight. It can be verified by a simple example in Δ_3. Let $x = (1-s, s, 0)$, $y = (1-s, 0, s)$. Then $d(x, y) = 2s$. But $E[d(\hat{x}, \hat{y})] = \frac{4s}{1+s} = \frac{2}{1+s} d(x, y)$. Therefore, the ratio $2/(1+s)$ approaches 2 as s goes to 0.

To end this subsection, we emphasize that the main results, i.e., Theorem 4 and 5 hold, regardless of which LP relaxation we solve.

4. Worst-Case Analysis

We estimate the performance guarantee of *FHSAP-GRA* in this section. We will always assume the discount factor $\alpha = 1$ for the convenience of the discussion because all theoretical analysis in this paper will hold for different α's.

Our goal is to bound the expected value of

$$\sum_{i,j\in\mathcal{C}} d_{ij}\left(\sum_{s\in\mathcal{H}} c_{is}\hat{x}_{i,s} + \sum_{t\in\mathcal{H}} c_{jt}\hat{x}_{j,t} + \sum_{s,t\in\mathcal{H}} \alpha c_{st}\hat{x}_{i,s}\hat{x}_{j,t}\right).$$

Recall that

$$L(\hat{x}) = \sum_{i,j\in\mathcal{C}} d_{ij}\left(\sum_{s\in\mathcal{H}} c_{is}\hat{x}_{i,s} + \sum_{t\in\mathcal{H}} c_{jt}\hat{x}_{j,t}\right)$$

and

$$Q(\hat{x}) = \sum_{i,j\in\mathcal{C}} d_{ij}\sum_{s,t\in\mathcal{H}} \alpha c_{st}\hat{x}_{i,s}\hat{x}_{j,t}.$$

We first focus on a special case in which the subgraph of hubs is an equilateral, i.e., distances between hubs are uniform. Without loss of generality, assume $c_{st} = 2$ for any two different hubs s and t. In this case, since $\hat{x}$ is a feasible solution to the *FHSAP*,

$$\begin{aligned}
\sum_{s,t\in\mathcal{H}} c_{st}\hat{x}_{i,s}\hat{x}_{j,t} &= 2\sum_{s,t\in\mathcal{H}:s\neq t} \hat{x}_{i,s}\hat{x}_{j,t} \\
&= 2(1 - \sum_{s\in\mathcal{H}} \hat{x}_{i,s}\hat{x}_{j,s}) \\
&= \sum_{s\in\mathcal{H}} |\hat{x}_{i,s} - \hat{x}_{j,s}| \\
&= d(\hat{x}_i, \hat{x}_j).
\end{aligned}$$

Thus, in this special case,

$$Q(\hat{x}) = \sum_{i,j\in\mathcal{C}} d_{ij} d(\hat{x}_i, \hat{x}_j).$$

4.1 Bounds With Respect to *FHSAP-LP1*

In this subsection, we assume that the LP relaxation *FHSAP-LP1* is used in algorithm *FHSAP-GRA*. We further assume that $(\vec{x}^*, \vec{X}^*)$ is an optimal solution to *FHSAP-LP1*. Our main result of this subsection is summarized in the following Theorem.

Theorem 7. *Assume that $c_{st} = 2$ for all $s \neq t$, then*

$$E[L(\hat{x})] + E[Q(\hat{x})] \leq \sum_{i\,j \in \mathcal{C}} \sum_{s,t \in \mathcal{H}} d_{ij}(c_{is} + c_{jt}) X^*_{ijst} + 2 \sum_{i\,j \in \mathcal{C}} \sum_{s,t \in \mathcal{H}} d_{ij} c_{st} X^*_{ijst}.$$

Proof. From Theorem 4, we know that

$$E[L(\hat{x})] = \sum_{i,j \in \mathcal{C}} d_{ij} (\sum_{s \in \mathcal{H}} c_{is} E[\hat{x}_{i,s}] + \sum_{t \in \mathcal{H}} c_{jt} E[\hat{x}_{j,t}]) = \sum_{i,j \in \mathcal{C}} d_{ij} (\sum_{s \in \mathcal{H}} c_{is} x^*_{i,s} + \sum_{t \in \mathcal{H}} c_{jt} x^*_{j,t}).$$

From the constraints of *FHSAP-LP1*, $x^*_{i,s} = \sum_{t \in \mathcal{H}} X^*_{ijst}$ and $x^*_{j,t} = \sum_{s \in \mathcal{H}} X^*_{ijst}$.

Thus,

$$\begin{aligned} E[L(\hat{x})] &= \sum_{i,j \in \mathcal{C}} d_{ij} (\sum_{s \in \mathcal{H}} c_{is} x^*_{i,s} + \sum_{t \in \mathcal{H}} c_{jt} x^*_{j,t}) \\ &= \sum_{i,j \in \mathcal{C}} d_{ij} (\sum_{s \in \mathcal{H}} c_{is} \sum_{t \in \mathcal{H}} X^*_{ijst} + \sum_{t \in \mathcal{H}} c_{jt} \sum_{s \in \mathcal{H}} X^*_{ijst}) \\ &= \sum_{i\,j \in \mathcal{C}} \sum_{s,t \in \mathcal{H}} d_{ij} (c_{is} + c_{jt}) X^*_{ijst}. \end{aligned}$$

Further,

$$\begin{aligned} E[Q(\hat{x})] &= \sum_{i,j \in \mathcal{C}} d_{ij} E[d(\hat{x}_i, \hat{x}_j)] \\ &\leq 2 \sum_{i,j \in \mathcal{C}} d_{ij} d(x^*_i, x^*_j) \\ &= 2 \sum_{i,j \in \mathcal{C}} d_{ij} \sum_{s \in \mathcal{H}} |x^*_{i,s} - x^*_{j,s}|, \end{aligned}$$

where the inequality holds because of Theorem 5. Further,

$$x^*_{i,s} - x^*_{j,s} = \sum_{t \in \mathcal{H}} X^*_{ijst} - \sum_{t \in \mathcal{H}} X^*_{ijts} = \sum_{t \in \mathcal{H} : t \neq s} (X^*_{ijst} - X^*_{ijts}).$$

Thus,

$$\sum_{s \in \mathcal{H}} |x^*_{i,s} - x^*_{j,s}| \leq \sum_{s \in \mathcal{H}} \sum_{t \in \mathcal{H} : t \neq s} (X^*_{ijst} + X^*_{ijts}),$$

which implies that

$$E[Q(\hat{x})] \leq 2\sum_{i,j\in\mathcal{C}} d_{ij}\sum_{s\in\mathcal{H}}\sum_{t\in\mathcal{H}:t\neq s}(X^*_{ijst}+X^*_{ijts}) = 4\sum_{i\,j\in\mathcal{C}}\sum_{s,t\in\mathcal{H}:s\neq t} d_{ij}X^*_{ijst} = 2\sum_{i,j\in\mathcal{C}}\sum_{s,t\in\mathcal{H}} d_{ij}c_{st}X^*_{ijst}.$$

This completes the proof. □

4.2 Bounds with Respect to ***FHSAP-LP3***

In this subsection, we assume that the LP relaxation *FHSAP-LP3* is used in algorithm *FHSAP-GRA*. We further assume that $(\vec{x}^*, \vec{Y}^*)$ is an optimal solution. Although *FHSAP-LP3* has less variables and less constraints than *FHSAP-LP1*, we can still prove a bound that is similar to Theorem 7.

Theorem 8. *Assume that $c_{st} = 2$ for all $s \neq t$, then*

$$E[L(\hat{x})] + E[Q(\hat{x})] \leq \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{H}} c_{is}(O_i + D_i)x^*_{i,s} + 2\sum_{i\in\mathcal{C}}\sum_{s,t\in\mathcal{H}:s\neq t} c_{st}Y^{i*}_{st}.$$

Proof. The proof is similar to that of Theorem 7. First,

$$E[L(\hat{x})] = \sum_{i,j\in\mathcal{C}} d_{ij}(\sum_{s\in\mathcal{H}} c_{is}x^*_{i,s} + \sum_{t\in\mathcal{H}} c_{jt}x^*_{j,t}) = \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{H}} c_{is}(O_i + D_i)x^*_{i,s}.$$

Second,

$$\begin{aligned} E[Q(\hat{x}) &= E[\sum_{i,j\in\mathcal{C}} d_{ij}\sum_{s\in\mathcal{H}} |\hat{x}_{i,s} - \hat{x}_{j,s}|] \\ &\leq \sum_{i,j\in\mathcal{C}}\sum_{s\in\mathcal{H}} d_{ij}(2|x^*_{i,s} - x^*_{j,s}|) \\ &= 2\sum_{i\in\mathcal{C}}\sum_{s,t\in\mathcal{H}:s\neq t} c_{st}Y^{i*}_{st}, \end{aligned}$$

where the last equality follows from the aggregate flow constraint of *FHSAP-LP3*. This completes the proof. □

Remark. Notice that the LP relaxation *FHSAP-LP2'* has individual flow constraints from lemma 1, it is a stronger LP formulation. So Theorem 8 holds for *GRA-LP2'* as well.

4.3 Performance Guarantee

Theorem 7 and 8 immediately imply that the algorithm *FHSAP-GRA* has a performance guarantee of 2 when the subgraph of hubs is an equilateral. In fact, the approximation ratio on the inter-hub cost is at most 2, and the expected value of the city-to-hub cost is the same as that in the LP relaxation. Therefore, the performance guarantee of our algorithm can be improved depending on the ratio of the city-to-hub cost relative to the inter-hub cost.

Now we state our main theorem regarding the performance guarantee of algorithm *FHSAP-GRA* for the general *FHSAP*.

Define $L = max\{c_{st} \colon s, t \in \mathcal{H}, s \neq t\}$, and $l = min\{c_{st} \colon s, t \in \mathcal{H}, s \neq t\}$. Further, let $r = \frac{L}{l}$, i.e., r is the ratio of the longest edge to the shortest edge among all inter-hub edges.

Theorem 9. *The algorithm* FHSAP-GRA *using the LP relaxation* FHSAP-LP1 *or* FHSAP-LP3 *has a performance guarantee of* $2r$*.*

Proof. Given an instance I of the *FHSAP*, we build another instance denoted by I_L, in which all of the inter-hub edges have the uniform length L. Thus, the subgraph of hubs is an equilateral for instance I_L. Let LP_I and LP_{I_L} denote the optimal objective value of the LP relaxations of instance I and I_L, respectively.

It is clear that

$$LP_{I_L} \geq LP_I \geq \frac{1}{r} LP_{I_L}.$$

Further, the expected cost of the solution generated by *FHSAP-GRA* for instance I should be no more than the expected cost of the solution generated by *FHSAP-GRA* for instance I_L, which is at most $2LP_{I_L} \leq 2rLP_I$, where the factor of 2 comes from Theorem 7 and 8. $\square$

We would like to point out that the ratio $2r$ is a worst case bound. The ratio is relative small when r is small. If a network is constructed in a way so that r is small, then even the worst case performance of our algorithm will not be too bad. In next section, we implemented our algorithm. The computational results suggest that it delivers solutions that are very close to the optimal ones.

Table 1: n=50, k=5.

Discount	Distribution	GRA-LP1		GRA-LP2			GRA-LP3		
		CPU	Gap1	CPU	Gap1	Gap2	CPU	Gap1	Gap2
$\alpha = 0.05$	U[0,20]	2.16	0.00%	0.03	0.00%	0.18%	2.94	0.00%	0.18%
	U[4,20]	2.4	0.00%	0.02	0.00%	0.56%	2.18	0.00%	0.56%
	U[14,20]	2.28	0.00%	0.02	0.00%	0.04%	1.26	0.00%	0.04%
	U[20,20]	2.14	0.00%	0.02	0.00%	0.00%	1.07	0.00%	0.00%
$\alpha = 0.25$	U[0,20]	2.59	0.00%	0.03	0.48%	7.28%	1	0.48%	7.28%
	U[4,20]	2.79	0.00%	0.03	0.19%	4.18%	5.12	0.19%	4.18%
	U[14,20]	3.22	0.00%	0.04	0.31%	1.55%	1.76	0.00%	1.23%
	U[20,20]	3.36	0.00%	0.04	0.27%	0.27%	1.27	0.00%	0.00%
$\alpha = 0.5$	U[0,20]	3.03	0.00%	0.03	0.77%	11.34%	2.12	0.77%	11.34%
	U[4,20]	3.1	0.07%	0.03	0.09%	5.81%	2.6	0.38%	6.12%
	U[14,20]	3.71	0.00%	0.05	1.61%	1.61%	2.26	0.00%	0.00%
	U[20,20]	3.1	0.00%	0.04	9.25%	9.25%	1.36	0.00%	0.00%
$\alpha = 1$	U[0,20]	3.3	0.00%	0.04	4.24%	12.19%	3.5	3.55%	11.45%
	U[4,20]	3.08	0.00%	0.04	1.83%	1.83%	1.58	0.00%	0.00%
	U[14,20]	2.55	0.00%	0.04	4.47%	4.47%	2.2	0.00%	0.00%
	U[20,20]	2.04	0.00%	0.04	0.00%	0.00%	2.14	0.00%	0.00%

5. Computational Results

Computational results for the implementation of *FHSAP-GRA* are reported in this section. We applied *FHSAP-GRA* to both randomly generated instances of three different sizes (Table 1, 2, 3, 4) and a benchmark problem data set (Table 5). All linear programs in the experiments were solved by CPLEX version 9.0 at a Stanford workstation (CPU: dual 3GHZ/ memory: 8GB), and the rounding procedures were conducted on a notebook (CPU: Pentium 1.5GHZ/memory: 1.0GB).

In all randomly generated examples, demands between cities are uniformly distributed on the interval $[0, 100]$ and all hub-to-city distances are uniformly distributed on the interval $[1, 11]$. We kept altering the distribution interval of inter-hub distances and the discount factor α.

The benchmark problem set we used is called *AP* (*Australia Post*) data set (Table 5) [13], which was collected from a real postal delivery network in Australia. It stores the coordinates and demands of 200 nodes (cities). Ernst and Krishnamoorthy solved p-hub location problems for *AP* data set, and we tested our algorithms on hubs their solutions specified. Some of the hub-to-city cost coefficients are non-symmetric in the *AP* data set, so we made adjustment to it accordingly by specifying in-flow and out-flow coefficients separately for each $x_{i,s}$.

In all the experiments, we run the rounding procedure 5000 times for those instances whose LP relaxations only have fractional optimal solutions. Considering that the running time of the

Table 2: n=100, k=10.

Discount	Distribution	GRA-LP1		GRA-LP2			GRA-LP3		
		CPU	Gap1	CPU	Gap1	Gap2	CPU	Gap1	Gap2
$\alpha = 0.05$	U[0,20]	768	0.00%	0.16	0.07%	6.42%	108	0.07%	6.42%
	U[4,20]	583	0.00%	0.31	0.22%	9.10%	89	0.22%	9.10%
	U[14,20]	567	0.00%	0.27	0.00%	0.16%	127	0.00%	0.16%
	U[20,20]	712	0.00%	1.65	0.33%	0.33%	123	0.00%	0.00%
$\alpha = 0.25$	U[0,20]	1115	0.00%	1.03	0.48%	7.28%	250	0.48%	7.28%
	U[4,20]	1329	0.00%	1.87	0.19%	4.18%	166	0.19%	4.18%
	U[14,20]	2276	0.00%	0.88	0.23%	1.66%	134	0.00%	1.43%
	U[20,20]	1318	0.00%	1.05	0.27%	0.27%	213	0.00%	0.00%
$\alpha = 0.5$	U[0,20]	1567	0.70%	0.87	0.77%	11.34%	178	0.77%	11.34%
	U[4,20]	9802	0.00%	1.12	10.95%	11.62%	159	10.89%	11.56%
	U[14,20]	9972	0.00%	2.2	0.51%	3.51%	113	0.32%	3.31%
	U[20,20]	10103	0.00%	1.08	9.25%	9.25%	230	0.00%	0.00%
$\alpha = 1$	U[0,20]	15249	0.00%	0.85	10.95%	51.17%	148	10.95%	51.17%
	U[4,20]	16851	0.00%	3.12	2.76%	15.07%	329	2.30%	14.55%
	U[14,20]	15439	0.00%	3.22	5.86%	7.47%	322	0.92%	2.45%
	U[20,20]	13780	0.00%	4.07	0.00%	0.00%	310	0.00%	0.00%

Table 3: n=200, k=10.

Discount	Distribution	GRA-LP2		GRA-LP3	
		CPU	Gap2	CPU	Gap2
$\alpha = 0.05$	U[0,20]	1.14	4.58%	620	4.58%
	U[4,20]	11.7	1.71%	1814	1.71%
	U[14,20]	1.8	0.11%	1816	0.11%
	U[20,20]	9.8	0.00%	765	0.00%
$\alpha = 0.25$	U[0,20]	2.9	21.19%	798	21.19%
	U[4,20]	11.9	10.18%	2518	10.18%
	U[14,20]	11.7	1.67%	1168	1.60%
	U[20,20]	8.2	0.22%	1408	0.00%
$\alpha = 0.5$	U[0,20]	6.2	33.20%	1957	33.20%
	U[4,20]	12.6	15.09%	3392	15.09%
	U[14,20]	14.7	6.68%	1333	5.39%
	U[20,20]	20.2	5.04%	1981	0.00%
$\alpha = 1$	U[0,20]	22.5	33.11%	2549	33.11%
	U[4,20]	23.1	11.88%	1750	12.80%
	U[14,20]	27.3	0.72%	3311	0.00%
	U[20,20]	32.7	0.00%	3278	0.00%

Table 4: n=1000, k=10.

Discount	Distribution	GRA-LP2			Heuristic
		LP	GRA	CPU	
$\alpha = 1$	U[0,20]	339820	479006	1957.1	587066
	U[4,20]	538346	611967	9887.2	594342
$\alpha = 0.5$	U[0,20]	276165	381070	508.4	418775
	U[4,20]	371262	426419	1741.5	461169
$\alpha = 0.25$	U[0,20]	269322	302734	1509.2	310079
	U[2,20]	270349	312992	513.9	316065
$\alpha = 0.05$	U[2,20]	210725	215030	1140.1	215274
	U[4,20]	213948	217007	119.2	217138

Table 5: AP benchmark problems.

n	k	Optimal	GRA-LP3				GRA-LP2			
			LP3	GRA3	CPU	Gap1	LP2	GRA2	CPU	Gap1
50	5	132367	132122	132372	6.94	0.004%	132120	132372	0.02	0.004%
50	4	143378	143200	143378	4.04	0.000%	143139	143378	0.01	0.000%
50	3	158570	158473	158570	1.92	0.000%	158139	158570	0.01	0.000%
40	5	134265	133938	134265	2.17	0.000%	133908	134265	0.02	0.000%
40	4	143969	143924	143969	1.16	0.000%	143707	143969	0.01	0.000%
40	3	158831	158831	158831	0.60	0.000%	158642	158831	0.01	0.000%
25	5	123574	123574	123574	0.23	0.000%	123574	123574	0.01	0.000%
25	4	139197	138727	139197	0.17	0.000%	138727	139316	0.01	0.085%
25	3	155256	155139	155256	0.09	0.000%	154786	155256	0.01	0.000%
20	5	123130	122333	123130	0.11	0.000%	122329	123130	0.01	0.000%
20	4	135625	134833	135625	0.08	0.000%	134827	135625	0.01	0.000%
20	3	151533	151515	151533	0.05	0.000%	150724	151533	0.01	0.000%
10	5	91105	89962	91105	0.02	0.000%	89961	91105	0.01	0.000%
10	4	112396	111605	112396	0.01	0.000%	111321	112396	0.01	0.000%
10	3	136008	135938	136008	0.01	0.000%	135223	136008	0.01	0.000%

algorithm is mainly spent in solving linear programs, CPU times reported in all tables are the running times for solving the LP relaxation of each instance.

Table 1 describes medium-sized examples, each of which has 50 cities and 5 hubs. Table 2 describes large-sized examples, each of which has 100 cities and 10 hubs. Table 1 and 2 present computational results for 32 instances by *FHSAP-GRA* with three different LP relaxations: *LP1*, *LP2*, and *LP3*. The running times and percentage gaps are given for each instance.

Denote algorithm *FHSAP-GRA* with the LP relaxation *LPi* by *GRA-LPi* ($i = 1, 2, 3$). For each algorithm *GRA-LPi*, denote the optimal objective value of the LP relaxation *LPi* by LP_i, and denote the value of an integral solution by algorithm *GRA-LPi* by GRA_i. Recall that LP_1 is known to be a very tight lower bound, we define $Gap1 = (\frac{GRA_i}{LP_1} - 1) * 100\%$ to measure the solution quality of each GRA_i. Similarly we define $Gap2 = (\frac{GRA_i}{LP_3} - 1) * 100\%$ considering that it is difficult to calculate LP_1 when the problem size becomes large.

The computational results in table 1 and 2 show that *FHSAP-GRA* with different LP relaxations delivers solutions with variable qualities and time complexity. We have the following observations. They are compatible with the literature and analysis developed in this article.

The first, LP_1 is a very tight lower bound. It automatically generates optimal integral assignments for 30 out of 32 instances. Furthermore, *GRA-LP1* gives near optimal solutions for the remaining 2 instances. However, the running time increases rapidly to an intractable level when the problem size is increased to $(100, 10)$ and the discount factor approaches 1. Therefore, *GRA-LP1* is especially efficient for medium-sized problems.

The second, GRA_2 is of particular value in real applications for large-sized problems. We can observe that the much smaller running time comes at the expense of marginally larger gaps. It performs at most 1% worse than optimal assignments on 22 out of 32 instances comparing to *GRA-LP1*. Results also reveal that the effectiveness of the solutions by *GRA-LP2* decreases as the discount factor gets larger.

The third, *GRA-LP3* delivers high-quality solutions for large-sized problems in a reasonable amount of time. It generates solutions at most 1% worse than optimal ones on 28 out of 32 instances. And it outperforms *GRA-LP2* on 15 out of 32 instances and is only inferior to *GRA-LP2* on 1 instance. Moreover, *GRA-LP3* always performs extremely well on instances where the graph of hubs has a (near) equilateral structure. We also observed that LP_3 is a tighter lower bound than LP_2. It improves LP_2 on 19 out of 32 instances.

For larger problems in table 3 we didn't attempt to compute their tight lower bounds LP_1 due to the excessive running time. However, *GRA-LP3* is still manageable at this size, which provides us a good lower bound LP_3 in most instances. There is one example in which both *GRA-LP2* and *GRA-LP3* have large values of $GAP2$. It is caused more possibly by the looseness of the lower bound LP_2 rather than by the algorithm itself.

For very large-sized problems in table 4, we only implemented *GRA-LP2* because of its infeasible demands on memory. We reported the running times of these problems, the lower bounds from *FHSAP-LP2* and the costs of rounded solutions. We also presented upper bounds derived from choosing the better one of two commonly used quick heuristics: the nearest neighborhood allocation heuristic and one-hub allocation heuristic. The former assigns every city to its nearest hub and the later assigns all cities to one single hub. We can observe that *GRA-LP2* outperforms them on 7 out of 8 instances.

In table 5, we tested 15 *AP* benchmark problems by *GRA-LP1*, *GRA-LP2* and *GRA-LP3* on fixed-hubs specified in their paper. Since solving *FHSAP-LP1* already produced optimal integral assignments for all 15 problems in less than 120 seconds, we omitted it in the table. *GRA-LP3* obtained optimal assignments on 14 out of 15 problems, and only 0.004% higher than the optimal cost on the remaining one, with much less time than *GRA-LP1*. *GRA-LP2* is the fastest algorithm, and generated optimal assignments on 13 of 15 problems. It performed 0.004% or 0.09% worse than the optimal cost on the remaining two problems.

6. Conclusion and the Future Work

In this paper we study the fixed-hub single allocation problem. We propose an LP-based algorithm for the general problem, which exhibits excellent performance in our computational study. Further, our algorithm enjoys a worst-case performance guarantee. To the best our knowledge, this is the first worst-case analysis for a heuristic proposed for the fixed-hub single allocation problem. Our results rely on a new randomized rounding technique, which is of interest on its own.

There are many interesting problems worth exploring in the future. There may exist other topologies of hubs which can be approached by constant approximation algorithms and to which a good embedding ratio from a metric graph can be found. The hub location problem with setup cost and capacity constraints is also useful in practice. Moreover, the geometric rounding may have applications in linear programming and other quadratic semi-assignment problems.

Acknowledgments

The authors would like to thank two referees for their insightful comments. We acknowledge that the (simplified) proofs of Theorem 4 and Lemma 6 are suggested by one of the referees. We also thank Huan Yang for his help in implementation and data collection.

References

[1] A. Ageev, M. Sviridenko. Pipage rounding: a new method of constructing algorithms with proven performance guarantee, *Journal of Combinatorial Optimization*, 8:307-328, 2004.

[2] D. Bertsimas, C. Teo, R. Vohra. On dependent randomized rounding algorithms, *Operations Research Letters*, 24(3): 105-114, 1999.

[3] G. Calinescu, C. Chekuri, M. Pal, J. Vondrak. Maximizing a submodular function subject to a matroid constraint. *In the Proceedings of the Conference on Integer Programming and Combinatorial Optimization (IPCO)*, 2007.

[4] R. Bollapragada, J. Camm, U. Rao, J. Wu. A Two-phase Greedy Algorithm to Locate and Allocate Hubs for Fixed-wireless Broadband Access, *Operations Research Letters*, Vol. 33,134-142, 2005.

[5] D. Bryan, M. E. O'Kelly. Hub and Spoke Networks in Air Transportation: an Analytical Review, *Journal of Regional Science*, Vol. 39(2), 275-295, 1999.

[6] J. F. Campbell. Integer Programming Formulation of Discrete Hub Location Problems, *European Journal of Operational Research*, Vol. 72, 387-405, 1994.

[7] J. F. Campbell. Hub Location and the p-Hub Median Problem, *Operations Research*, Vol. 44, 923-935, 1996.

[8] J. F. Campbell, A. T. Ernst, M. Krishnamoorthy. Hub Location Problems, *Facility Location: Application and Theory*, Z. Drezner and H.W. Hamcher(EDs.), Springer, 2002.

[9] J. F. Campbell, A. T. Ernst, and M. Krishnamoorthy. Hub Arc Location Problems: Part I-Introduction and Results, *Management Science*, Vol. 51, No. 10, 1540-1555, 2005.

[10] J. F. Campbell, A. T. Ernst, and M. Krishnamoorthy, Hub Arc Location Problems: Part II-Formulations and Optimal Algorithms, *Management Science*, Vol. 51, No. 10, 1556-1571, 2005.

[11] G. Carello, F. Della Croce, M. Ghirardi, R. Tadei. Hub Location Problem in Telecommunication Network Design: A Local Search Approach, *Networks*, Vol. 44, 94-105, 2004.

[12] S. Chamberland, B. Sanso, O. Marcotte. Topological Design of Two-level Telecommunication Netwroks with Modular Switches. *Operations Research*, Vol. 48, 745-760, 2000.

[13] A.T. Ernst, M. Krishnamoorthy. Efficient Algorithms for the Uncapacitated Single Allocation p-hub Median Problem. *Location Science*, Vol. 4, No.3, 139–154, 1996.

[14] A.T. Ernst, M. Krishnamoorthy. Solution Algorithms for the Capacitated Single Allocation Hub Location Problem. *Annals of Operations Research*, Vol. 86, 141–159, 1999.

[15] R. C. Gandhi, S. Khuller, S. Parthasarathy, A. Srinivasan. Dependent Rounding and its Applications to Approximation Algorithms. *Journal of the ACM*, Vol. 53, 324–360, 2006.

[16] D. Ge, Q. Qi, Y. Ye. Geometric Rounding and its application in combinatorial Auction. *In preparation.*

[17] H. Jeffreys, B.S. Jeffreys. Change of Variable in an Integral. *Methods of Mathematical Physics*, 3rd ed., Cambridge University Press, 32-33, 1988.

[18] J.G. Klincewicz. Heuristics for the p-hub location problem. *European Journal of Operational Research*, Vol. 53, 25-37, 1991.

[19] J.G. Klincewicz. Avoiding local optima in the p-hub location problem using Tabu Search and grasp. *Annals of Operations Research*, Vol. 40, 283-302, 1992.

[20] M. Labbé, H. Yaman, E. Gourdin. A Branch and Cut Algorithm for Hub Location Problems with Single Assignment, *Mathematical Programming*, Vol. 102, 371-405, 2005.

[21] R.E. Marsten, M.R. Muller. A Mixed-Integer programming Approach to Air Cargo Fleet planning, *Management Science*, Vol. 26, 1096-1107, 1980.

[22] M. E. O'Kelly. A Quadratic Integer Program for the Location of Interacting Hub Facilities, *European Journal of Operational Research*, Vol. 33, 393-402, 1987.

[23] M. E. O'Kelly, D. Bryan, D. Skorin-Kapov, J. Skorin-Kapov. Hub Network Design with Single and Multiple Allocation: a Computational Study, *Location Science*,Vol. 4, 125-38, 1996.

[24] M. E. O'Kelly, D. Skorin-Kapov, J. Skorin-Kapov. Lower bounds for the hub location problem, *Management Science*, Vol. 41, 713-721, 1995.

[25] H. Pirkul; D. A. Schilling. An Efficient Procedure for Designing Single Allocation Hub and Spoke Systems, *Management Science*, Vol. 44, No. 12-2, 235-242, 1998.

[26] W. B. Powell, Y. Sheffi. Design and Implementation of an Interactive Optimization System for Network Design in the Motor Carrier Industry, *Operations Research*, Vol. 37, 12-29, 1989.

[27] D. Skorin-Kapov, J. Skorin-Kapov, M. E. O'kelly. Tight Linear Programming Relaxations of Uncapacitated p-hub Median Problems, *European Journal of Operational Research*, Vol. 94, 582-593, 1996.

[28] J. Sohn and S. Park. The Single-Allocation Problem in the Interacting Three-Hub Network, *Networks*, Vol. 35(1), 17-25, 2000.

[29] A. Srinivasan. Distributions on Level-Sets with Applications to Approximation Algorithms. *In the proceedings of IEEE Symposium on Foundations of Computer Science(FOCS)*, 2001.

[30] V. Vazirani. Approximation Algorithms, Springer, 2004.