Hierarchical Placement and Network Design Problems

Sudipto Guha* Adam Meyerson† Kamesh Munagala‡

Abstract

In this paper, we give the first constant-approximations for a number of layered network design problems. We begin by modeling hierarchical caching, where caches are placed in layers and each layer satisfies a fixed percentage of the demand (bounded miss rates). We present a constant approximation to the minimum total cost of placing the caches and routing demand through the layers. We extend this model to cover more general layered caching scenarios, giving a constant combinatorial approximation to the well studied multi-level facility location problem. We consider a facility location variant, the Load Balanced Facility Location *problem in which every demand is served by a unique facility and each open facility must serve at least a certain amount of demand. By combining Load Balanced Facility Location with our results on hierarchical caching, we give the first constant approximation for the* Access Network Design *problem.*

1 Introduction

In this paper we consider applications requiring layered placement of resources. Examples of such problems include hierarchical caching on the internet, multi-level facility location problems, and the Access Network Design problem which is a variant of single-sink buy-at-bulk problem. We develop techniques to approximate these problems within constant factors. The access network design problem leads us to formalize Load Balanced Facility Location which we solve as well.

*Department of Computer Science, Stanford University CA 94305. Research Supported by IBM Research Fellowship, NSF Grant IIS-9811904 and NSF Award CCR-9357849, with matching funds from IBM, Mitsubishi, Schlumberger Foundation, Shell Foundation, and Xerox Corporation. Email: sudipto@cs.stanford.edu.

†Department of Computer Science, Stanford University CA 94305. Supported by ARO DAAG-55-97-1-0221. Email: awm@cs.stanford.edu.

‡Department of Computer Science, Stanford University CA 94305. Supported by ONR N00014-98-1-0589. Email: kamesh@cs.stanford.edu.

Hierarchical Caching Considerable research has been devoted to optimizing internet access time through the use of various caching strategies. The central idea is to replicate data to reduce the access costs of the users. Caching schemes model tradeoffs between the storage and management of the copies and the average time to access a certain object in a network. The facility location problem has been used to model such adaptive data replication and placement problems [21]. However in most caching scenarios the caching is performed hierarchically. Small caches are placed close to the users while larger ones reside in the backbone of the network [27, 13].

We consider the problem of hierarchical placement of caches, using the web as our example. We model the caches at each layer of the hierarchy as having a fixed miss rate. Thus we consider a hierarchical caching scheme as follows: demand locations communicate with caches of type 1, which in turn communicate with caches of type 2 and so on, until we reach the servers located at the top-most level. We assume that each cache in level i of the hierarchy has miss rate σ_i. We refer to caches in level i of the hierarchy as type–i caches.

The miss rate of each layer is modeled to be an arbitrary fixed constant. Each demand location in the network could have many users, so it seems natural to model an average user profile instead of considering each user to be distinct. There is also some evidence that the distribution of the fraction of times different pages are accessed is Zipfian [9, 28]. A natural caching scheme is to cache the most popular pages [24, 9, 28]. Under this scheme, the miss rate is independent of the incoming request rate and depends on the Zipf distribution over pages of the incoming request stream and the size of the cache [9]. Capacitated models are frequently introduced to prevent cache overload. Constraints in capacities immediately pose questions of "economies of scale" or "buy at bulk" constraints [22]. We address this issue as well.

The quality of a solution is measured on two axes, the cost of placing the caches and the average time to serve the requests. A simple formulation is to optimize the sum of the placement cost and the service cost, as in facility location problems. For most such problems heuristics that try optimizing both the criterion follow without much difficulty from those which attempt minimizing the sum. The access time will be modeled by a metric distance function on the possible locations in the network. We will define a formal model in Section 2. This problem subsumes facility location problem and is Max SNP hard. We will obtain a constant factor approximation, with running time depending polynomially on the number of demand locations and the number of layers.

This problem arises in a variety of other contexts as well. Consider the problem of caching multimedia clips [14] in a hierarchical fashion. It can be shown analytically that the interval caching scheme illustrated in [14, 3] is optimal for this application. Assuming a statistical distribution for the interval between successive requests we can show that the reduction in rate is a function of the length of the clip, size of the cache and the distribution, but is independent of the actual request rate itself. This implies that if we place the caches in a hierarchy, so that the caches in the same layer have the same size, the reduction in request rate due to a cache at level i is a fraction σ_i that depends just on the layer. In such a scenario, we can define the problem of Hierarchical Placement just as we did for caching web pages above.

Multi-Level Facility Location The Multi-Level facility location problem was previously studied in [25, 17, 2, 26]. In a typical application, we have to route material produced at factories to consumers through a hierarchy of warehouses. We place large warehouses close to the factories, and smaller ones close to the consumers. The cost of placing a warehouse depends on the level the warehouse is to function in, its size, and its location. This problem closely resembles the web caching problems discussed above, except that there is no reduction in the demand from one layer to the next. We provide the first constant factor combinatorial approximation for this problem.

Access Network Design The problem of designing networks using trunks to route demands has received considerable attention. In this problem, commonly known as *Buy-at-Bulk Network Design* [22, 5, 4, 20], we are given demands at nodes in a network which have to be routed to their respective destinations using pipes of certain capacities and costs per unit length. The costs obey economies of scale, in the sense that it is cheaper to buy a pipe of larger capacity than many pipes (which sum to the same capacity) of a smaller capacity. The goal is to optimize the total cost of the pipes we buy to route the demands. Andrews and Zhang [4] define a special case of this problem called the *Access Network Design* problem, where all demands need to be routed to a central core network and the costs and capacities of the pipes obey certain common constraints. They show applications of this problem in designing telephone networks.

Load Balanced Facility Location We provide the first constant factor approximation for the Access Network Design problem by showing its connection to hierarchical placement through the problem of *Load Balanced Facility Location*. This problem differs from standard facility location in that we must route at least L units of demand to each open facility. Load Balanced Facility Location has direct applications; consider a franchise which must open stores to minimize the average distance from customer to store, but which must also guarantee a minimum number of customers to each store so the individual stores remain profitable. We present a constant approximation to this problem, losing a constant factor against the lower bound on demand.

1.1 Previous Results

Classical facility location and its variants have several constant factor approximations [12, 16, 23]. The algorithms in [12, 16] run in $O(n^2 \log n)$ time, while the algorithm in [23] solves an LP and rounds the fractional solution. Jain and Vazirani [16] also give a 4-approximation to the capacitated facility location problem, under the relaxation that more than one facility can be placed at a location. This is a theme running through almost all capacitated facility location problems, since such a relaxation makes it easier to extend a partial assignment of facilities.

A constant factor approximation for the buy at bulk facility location problem follows from [16]. Although not explicitly stated, the only observation one requires is that the algorithm of [16] carries through for capacities which are dependent on location only. Simply making copies of a facility (as many copies as supported by the buy at bulk constraints at this location) solves this problem. Although this is a trivial extension of the result in [16], it will be central in handling capacities and buy at bulk constraints in the context of Hierarchical Placement.

The Multi Level facility location problem has a constant factor approximation [1]. However, this algorithm is not combinatorial.

The problem of *buy-at-bulk* network design was first defined in [22]. Awerbuch and Azar [5] obtain an $O(\log^2 n)$ approximation to this problem even when all demands have different sinks. Their work is based on techniques to approximate any metric by tree metrics [7]. The approximation factor can be improved to $O(\log n \log \log n)$ using Bartal's result in [8] and derandomized using the results in [10, 11]. For *single sink* buy-at-bulk, Salman et al [22] show a constant approximation when there is only a single pipe type. Their method is based on the technique of balancing Steiner and shortest path trees [6, 18]. Andrews and Zhang [4] define the Access Network Design problem, which is a special case of single sink buy-at-bulk where the pipe costs and capacities obey certain common constraints, and provide an $O(K)$ approximation, where K is the number of pipe types. Meyerson et al [20] provide an $O(\log n)$ approximation to single sink network design where the costs on the edges are arbitrary non-decreasing concave functions of demand.

1.2 Organization of the Paper

In the next section, we present a formal model of hierarchical placement problems. In Section 3, we present a constant factor approximation for the simplest problem, where the cache cost depends just on the type (level) of the cache and not on the location. This will illustrate our basic techniques for *layered facility location*. We generalize the technique in Sections 4 and 5.

We show that the Hierarchical Placement Problem is not difficult to approximate if the miss rates of caches are small. This hinges on the idea that since the demands of the requests decrease, the contribution from the higher levels to the service cost decrease. However, as we proceed with a level by level assignment, we may be moving away from the optimal solution in each successive level. We need to modify the problem into suitable facility location problems to reflect the cost of building caches and finding an allocation path at the same time.

We subsequently show that the general problem in which the miss rate of level i is less than or equal to one (σ_i need not be monotonic in i) can be reduced in an approximation preserving fashion to a problem where the miss rates of caches in *every level* are small. We believe this two step approach of solving hierarchical problems is helpful in terms of clarity and simplicity of algorithms.

In Section 6, we define the Access Network Design problem, and exhibit its connection to the small miss rate idea from Section 3. We then reduce this problem to the *Load Balanced Facility Location* problem. In Section 7, we provide constant factor approximation to the load balanced facility location problem. In Section 8, we show how to combine the ideas from Section 3 and Section 7. The main idea is to solve load balanced facility location layer-by-layer, and use the proof technique from Section 3 to bound the cost at each layer.

2 Hierarchical Placement

We are given an undirected network $G(V, E)$ with a distance function on the edges, a set of demand points, and a set of possible locations and caches of types $1, \ldots, k$ with miss rates $\sigma_1, \ldots, \sigma_k$. Each cache is specified by the tuple $\langle u, y, i \rangle$, where u denotes the location of the cache, y its capacity and i its type (level). The cost of such a cache is denoted by $f^y_{u,i}$.

We have to place the caches satisfying the following constraints:

1. Each of the demand points are served by type 1 caches.Each of the type 1 caches placed behave as demand points that need to be served by type 2 caches. We can consider the demand points to be level 0 caches.

2. The demand of a type 1 cache is σ_1 times the amount of the demand served by the type 1 cache. The caches of type 2 onwards behave similarly as the caches of type 1, with miss rates σ_i for level i.

3. The cost of the solution is the cost of placing all the caches and the cost of servicing all the demands. The latter is the sum of the distance times the demand of the demand points, the type 1 caches, and caches of higher types[1].

We present three versions of this problem in increasing order of difficulty[2]: the first shows the general geometric property we would consider to solve this problem, the second problem shows how to deal with costs which are dependent on location. The third combines those techniques in their generality, with capacities and buy at bulk thrown in.

SIMPLE-PLACEMENT**:** In this version, the cost of a cache is independent of the location and just depends on the type of the cache. The caches have no capacity in terms of the amount of incoming demand.

MULTI-LEVEL**:** Here, all the σ_i are 1, but the cost of a facility depends on the location as well as the type of the cache. This is the classic multi-level facility location problem.

GENERAL-PLACEMENT**:** Cache of type i now has have capacity M_i in terms of the amount of incoming demand. We are allowed to place multiple caches at a location. The cost of a cache depends both on the location and the type of the cache.

It is straightforward that these problems extend the facility location problem, and therefore are NP–Hard. In fact these problems cannot be approximated within factor 1.47 unless $P = NP$, [15]. In the next sections, we will present constant factor approximations for these problems.

[1] It is also conceivable that the cost of an access is accounted by the distance to the type (level) 1 cache, and in case of a miss, σ_1 times the distance back to the original demand node and then to the type (level) 2 cache. In this case the demand node is aware of the miss and finding the next level cache itself. This objective function decomposes into several applications of the facility location problem, one for each level. In this model we are assuming that the demand node is not aware of a cache miss and the redirection is performed by the caches, as in most caching scenarios.

[2] There are other intermediate versions, but we omit discussion as the techniques used will be similar.

3 The SIMPLE-PLACEMENT Problem

Recall that in this version of the problem, the cost of a cache of type i is f_i and is independent of the location of the cache. Each cache can support infinite user demand, we will address this restriction in the later sections. We first present a small miss rate theorem, which is crucial to our algorithm.

3.1 Small Miss Rate Theorem

We reduce the problem to a new instance where the miss rates are small. We will used primed quantities to distinguish the variables of new instance from the original one.

Theorem 3.1 *Given an instance of the* SIMPLE-PLACEMENT *problem with parameters $\sigma_1, \ldots, \sigma_k$, the problem can be reduced to a different caching problem with $k' \leq k$ types of caches and a different set of parameters $\sigma'_1, \ldots, \sigma'_{k'}$, such that each σ'_ℓ is a constant less than α where $0 < \alpha < 1$, and at most a $\frac{1}{\alpha}$ blowup in service cost.*

Proof: We show this by providing a new instance in which the σ'_i are at most α, and any solution in the newer instance can be realized in the older instance at the same cost. We will also show that the optimal solution in the older instance can be modified to give a feasible solution in the newer instance of cost at most $\frac{1}{\alpha}$ times.

We first find a j such that $\prod_{i=1}^{j_1} \sigma_j < \alpha$. We will pretend that the caches of types 1 through j form a single cache in the new instance.

We repeat the above for caches of type i for $i > j_1$, that is we find j_2 such that $\prod_{i=j_1+1}^{j_2} \sigma_i < \alpha$. Thus the number k' will be the number of times this step can be performed. The new σ'_1 will be $\prod_{i=1}^{j_1} \sigma_i$ and σ'_2 will be $\prod_{i=j_1+1}^{j_2} \sigma_i$ and so on.

In the new instance the cost of a cache of type 1 will be the sum of the costs of caches of types 1 through j_1 in the original instance. Similarly the cost of a cache of type 2 in the new instance will be the sum of the costs of the caches of types $j_1 + 1$ through j_2.

The demand points remain the same.

It is easy to see that a solution in the new instance gives a solution of the old instance by simply placing caches of types 1 through j_1 in the locations where a type 1 cache is placed in the new instance. And similarly for caches of type 2 and onwards in the new instance are replaced by (possibly) sets of caches of the old instance. It is easy to see that the cost remains unchanged in this process.

Thus we showed that a solution of a certain cost in the new instance can be realized in the older instance at the same cost. We will now show that the optimal solution of the older instance can be modified into a feasible solution of the newer instance by at most a constant factor blowup in the cost.

Consider the optimal solution. Consider the level j_1 at which the product of the parameters $\prod_{i=1}^{j_1} \sigma_i < \alpha$ for the first time. If we were to short-circuit all the paths from the demand points to the type j_1 caches, the cost of service will be at most $\frac{1}{\alpha}$ times original.

This is because the product of the σ_i values of levels 1 through $j_1 - 1$ in the original instance is greater than α. Notice that since every demand point is assigned to an unique type 1 cache and each such to an unique type 2 cache and so on, each demand can be associated with an unique cache of type j.

We can now place caches of type 1 through $j_1 - 1$ at the same location u where a type j_1 cache is placed, and since every cache of type i is assigned to an unique cache of type $i + 1$, the optimal assignment had also built at least these many caches for this cache of type j_1. Thus the cost of caches does not go up in this process. Thus we only increased the service cost by a factor of $\frac{1}{\alpha}$. This completes the reduction. ■

This tells us that any instance of SIMPLE-PLACEMENT can be reduced to the case where the miss rate σ_i is less than α.

3.2 Solving for small miss rate

The algorithm is natural and simple. We repeatedly solve an uncapacitated facility location problem for each of the levels. The reason why such a solution works is that since the miss rate is sufficiently bounded away from one, the secondary demands introduced at the facilities and their contributions decrease fast and geometrically. Recall that each σ_i for each level is at most α.

We solve the uncapacitated facility location for level i by scaling the distances by α^{i-1}, and the demands in their original locations. We continue the process till we have considered all the levels. Let us call the set of facilities opened in level i by P_i. Each demand is assigned to one facility in each of the P_i. Let the routing cost of the level i solution be X_i.

Now, for each demand, we send it to its assigned facility in P_1, and from there to its assigned facility in P_2, and so on till its assigned facility in P_K. This solution is not a tree, but it can be converted to a tree of no greater cost.

We denote the best known approximation ratio for the facility location problem by r. For level i, let the total cost of the caches placed by our solution be F_i and that placed by the optimum solution on the new cache types be F_i^*. Also, let the service cost be S_i and S_i^* respectively at level i.

Let $X = \sum_i X_i$, $S^* = \sum_i S_i^*$, $F^* = \sum_i F_i^*$, $S = \sum_i S_i$ and $F = \sum_i F_i$. Note that the optimum cost is $S^* + F^*$, while our cost is $S + F$.

We first show a bound of our routing cost S_i in terms of X_i (our facility location routing cost of demands at layer 1 to the facilities P_i). We next show how X_i is related to S_i^*.

Lemma 3.1 $S_i \leq X_i + \alpha X_{i-1}$.

Proof: One feasible way of routing the demand at layer $i-1$ to layer i is to send it back to the original demand points along the level $i-1$ facility location solution, at cost αX_{i-1} (note that we scaled the metric by another factor α), and send it back along the level i solution at cost X_i. Since we use the shortest paths to route the demand between the same starting and ending points, our cost can be no more than this. ■

Lemma 3.2 $X_i + F_i \leq r(\sum_{j=0}^{i-1} \alpha^j S_{i-j}^* + F_i^*)$.

Proof: One feasible way of constructing the level i facility location solution (i.e., route the demands to the set P_i) is to route them along the optimum solution to layer i. Since we have scaled the metric by α^{i-1}, the optimum routing cost is at most $\sum_{j=0}^{i-1} \alpha^j S_{i-j}^*$. ■

These lemmas immediately give the following theorem:

Theorem 3.2 $S + F \leq r(\frac{1+\alpha}{1-\alpha} S^* + F^*)$.

Theorem 3.3 *The above algorithm is a constant approximation for the* SIMPLE-PLACEMENT *Problem.*

Proof: First, note that we lost a factor α in the routing cost because of the short-circuiting step in Theorem 3.1. Therefore, we have a $r\frac{1+\alpha}{\alpha(1-\alpha)}$ approximation on the routing cost, and r approximation on the facility cost. Setting $\alpha = \sqrt{2}-1$, we have a $(5.828r, r)$ bicriteria approximation on the routing and facility costs, which gives a 10 approximation. We can use the $(1+\gamma, 1+\ln\frac{2}{\gamma})$ facility location trade-off from [12] to get a 6 approximation to this problem. ■

4 Multi-Level Facility Location

In this version, there is no reduction in demand from one layer to the next, *i.e.*, all the σ are 1. Let f_{ji} be the cost of placing a facility of layer i at location j. This problem has been studied in [25, 17, 2, 26, 23]. We note that this problem has a 3 approximation [1], but this algorithm is not combinatorial. We present a combinatorial $O(1)$ approximation.

We introduce a technique for handling location dependent costs in placement problems. The next section will further generalize our algorithms to handle capacitated facilities as well. We can show the following.

Theorem 4.1 *The Multi-Level facility location problem has a $9.2(1+\epsilon)$-approximation running in time polynomial in the size of the input, k, and $\frac{1}{\epsilon}$.*

Proof: We will show that we can construct a new problem instance which is just capacitated facility location with the freedom to place multiple facilities(caches) at a location, with just a constant factor loss in cost.

In the new instance, let $\langle v, y\rangle$ denote a cache of capacity y at location v. Noticed we are not using the third parameter since all $\sigma_i = 1$ except the level k facilities (caches).

Let us consider the cost of cache $\langle u, y\rangle$, we will compute these for all v, all y in powers of $1+\epsilon$ in the new instance.

1. Construct a labeled directed graph with $K+1$ copies of the original graph. Let the copies be $G(0)$ to $G(K)$. From a node v in $G(i-1)$ (for $1 \leq i \leq j$) draw an arc of cost $y \cdot c_{vw} + f_{wi}$ to w in $G(i)$.
2. For each node v find the shortest path from the copy of v in $G(0)$ to the copy of v in $G(K)$. Denote the cost of this path by R_v^y.
3. The cost of $\langle u, y\rangle$ is set to be R_u^y.

We can convince ourselves that a solution of this new instance can be achieved in the older instance with at most the same cost. Consider a location v with cache in the new instance with capacity x. If in the older instance, we first bring all this demand to location v, and then send it on the round trip denoted by R_v^x. Notice that on this round trip, the demand will be routed through locations and caches of type 1 type 2, and so forth up to type K (corresponding to older instance) before reaching v. Thus we can short circuit using triangle inequality and achieve a routing of cost at most denoted by the cost of the newer instance.

We will now show that the optimal solution of the older instance can be modified into a feasible solution at most a constant factor blowup in cost. Consider all the demand at a cache of type K in the older solution. We can short circuit the demand from points to this cache, say at v, directly. That does not change the service cost. However the cache now costs more, but by not too much. Consider the closest demand point p from this location v, if we send the entire demand at location v to p and back through the path by which demand point p is connected to v, the cost will be at most two times the service cost. Thus if the demand, say y was a power of $1+\epsilon$, then we can choose $x = y$ and R_v^x would have been at most this round trip cost plus the costs of the caches of type 1 through K, (of the original instance) along the path from p to v. However we have to account for the fact that y may not be a power of $1+\epsilon$ and we may end up choosing an x which is at most $(1+\epsilon)y$. Thus the cost of routing the demand may lose another factor of $1+\epsilon$. Thus the cost of the new solution is at most $3(1+\epsilon)$ of the original.

We now use the a constant factor approximation for capacitated facility location in [16] to obtain a $12(1+\epsilon)$ approximation. The ratio can be improved to 9.2. ■

5 The GENERAL-PLACEMENT Problem

We will now combine the ideas in the previous two sections to solve the most general case. Each cache is denoted by $\langle u, y, i\rangle$ where the location is u, capacity restriction y, and type i. Multiple caches can be placed at the same location (of identical or different types) in order to accommodate large demands.

Reduction to Small Miss Rate We show how to reduce the problem with arbitrary σ_i to the case where all σ'_i are below α. The following theorem applies:

Theorem 5.1 *Given an instance of the* GENERAL-PLACEMENT *problem with parameters $\sigma_1, \ldots, \sigma_k$, the problem can be reduced to a different caching problem with $k' \leq k$ types of caches and a different set of parameters $\sigma'_1, \ldots, \sigma'_{k'}$, such that each σ'_ℓ is a constant less than α where $0 < \alpha < 1$, in time polynomial in input size and $\frac{1}{\epsilon}$, and at most a $\frac{3(1+\epsilon)}{\alpha}$ blowup in cost.*

Proof: As before, we will identify again types 1 through k' of the caches in the new instance. That is σ'_1 will be $\prod_{i=1}^{j} \sigma_i$ for the first j when $\prod_{i=1}^{j} \sigma_i \leq \alpha$. The parameter σ'_2 will be $\prod_{i=j+1}^{j'} \sigma_i$ for the first j' for which the quantity $\prod_{i=j+1}^{j'} \sigma_i \leq \alpha$.

The caches in the new instance will be denoted by triples, $\langle v, y, 1\rangle'$ denoting a cache at v of type 1 in the new instance having capacity y (allowing copies to be placed simultaneously). The primed triples will indicate caches in the new instance.

Let us consider how to assign the cost of cache $\langle u, y, 1\rangle'$. We will compute these for all v, all y in powers of 2 and all types 1 through k' in the new instance. (The type $1'$ in the new instance is corresponding to types 1 through j_1 in the old instance). We will use ideas from the Multi–Level Facility Location Problem. The idea is that although the demands decrease, we can approximate the sub-levels in the original problem (1 through j for example) by a Multi–Level facility location problem, appropriately modified to account for capacities.

1. Construct a labeled directed graph with $j+1$ copies of the original graph. Let the copies be $G(0)$ to $G(j)$. From a node v in $G(i-1)$ (for $1 \leq i \leq j$) draw an arc of cost $y \cdot c_{vw} + \min_{\langle u,x,i\rangle} \left\{ \lceil \frac{y}{x} \rceil f^x_{wi} \right\}$ to w in $G(i)$. If there are no $\langle u, x, i\rangle$, we will not introduce the arc or introduce an arc of cost inf (very large positive value). The expression reads complicated, but encodes the best way of routing demand y from v to w under restriction that w has a cache of type i, and $\langle w, x, i\rangle$ are the possible caches that can be used.
2. For each node v find the shortest path from the copy of v in $G(0)$ to the copy of v in $G(j)$. Denote the cost of this path by R^y_{vj}.
3. The cost of $\langle u, y, 1\rangle'$ is set to be R^y_{uj}.
4. Likewise we set the costs $\langle u, y, 2\rangle'$ etc.

The proof that this construction works is similar to the proofs of Theorem 4.1 and Theorem 3.1, and is omitted. ∎

Solving Small Miss rate Case The algorithm is the same as in Section 3. We solve capacitated facility location [16] at each of the new levels in turn, using $\langle u, y, i\rangle'$ as the facilities. The proof of the following theorem proceeds along the same lines as the proof of Theorem 3.3.

Theorem 5.2 *Given an instance of* GENERAL-PLACEMENT *where the σ_i are bounded by α, we can approximate the problem within $O(1)$.*

Combining Theorem 5.1 and Theorem 5.2 gives an $O(1)$ approximation for GENERAL-PLACEMENT.

6 Access Network Design – Preliminaries

We apply the hierarchical placement technique of the previous section to construct a constant factor approximation for *Access Network Design* problem [4].

6.1 The Model

Consider a service provider or utility company trying to design a network to provide service. The agency can use the economies of scale in designing the network, that the cost per unit traffic decreases when routing larger amounts. This is modeled by "buy at bulk" network design problems. The Access Network Design Problem [4] is a special case of single sink buy-at-bulk [22].

In single sink buy-at-bulk, we are given a network $G(V, E)$ with a length function on the edges, and a sink node $s \in V$. We have a set $R \subseteq V$ of demand points which we have to route to the sink by buying pipes and laying them along the edges in the network. We are given K types of pipes. A pipe of type k has capacity u_k and cost per unit length c_k. We can assume that it is always cheaper to buy a single copy of a pipe then to buy other pipes which have equal or greater total capacity.

The goal is to minimize the cost of buying the pipes along the edges to route all demand to the sink. Since the cost per unit demand goes down with the capacity of the pipes, there exists a near-optimum solution (within a factor of two) which always buys edges along a tree [5].

Andrews and Zhang proposed the following alternate formulation, and proved it equivalent to within a factor of two [4]. The pipe of type k has a fixed cost $\phi_k = c_k$, and an incremental cost per unit demand of $\delta_k = \frac{c_k}{u_k}$. The cost per unit length of routing a demand of d using a pipe of type k is therefore $\phi_k + d \cdot \delta_k$.

Numbering the pipes in increasing order of capacity, we immediately have the conditions that $\phi_1 < \phi_2 < \cdots < \phi_K$, and $\delta_1 > \delta_2 > \cdots > \delta_K$.

The Access Network Design problem is a special case of Single Sink buy-at-bulk with additional restrictions on the costs of the pipe types. The main restriction is that a type k pipe is cheaper only when it routes significant demand. Formally, the restrictions can be stated as follows:

1. For $2 \leq k \leq K$, if $d < \frac{\phi_k}{\delta_k}$, then $d\delta_{k-1} + \phi_{k-1} < d\delta_k + \phi_k$. For this to make any physical sense, we would actually require $d < \beta \frac{\phi_k}{\delta_k}$ for some $\beta < 1$. Since it will not effect our proofs, we will simplify our notation by assuming $\beta = 1$.

2. The smallest demand looks like the smallest pipe capacity[3], or more precisely, $d \cdot \delta_1 > \phi_1$.

3. $\sum_{\kappa < k} \phi_\kappa = O(\phi_k)$.

Under these restrictions, Andrews and Zhang [4] showed that there exists a near-optimal (within a constant multiplier on the cost) solution which is a tree satisfying the following properties:

1. Each demand is routed through pipes of consecutive types, i.e. types $1, 2, \ldots, \kappa$. $(\kappa \leq k)$.

2. For all pipe types k, any pipe of that type has at least $u_k = \frac{\phi_k}{\delta_k}$ amount of demand flowing through it.

We can therefore compare ourselves against the optimal solution that satisfies the above mentioned properties. We can assume every pipe type is used for routing all demands by placing self loops of increasing types at the sink, provided the self loops of type k we insert at the sink are not required to have u_k amount of demand in them.

6.2 Relationship with Hierarchical Placement

The Access Network Design problem seems unrelated to the hierarchical placement problems we have defined. However, in each case we have a layered solution with a reduction in cost at each layer. We can prove the following theorem, whose proof is very similar to that of Theorem 3.1.

[3]We can easily remove this restriction with a constant factor loss.

Theorem 6.1 *There exists a solution to the Access Network Design problem in which we only use pipe types satisfying the condition $\sigma_i = \frac{\delta_{i+1}}{\delta_i} \leq \alpha$, and in which any pipe of type i has at least u_i amount of demand flowing through it. The fixed and incremental costs of this solution are each within $\frac{1}{\alpha}$ of the original optimum which used all pipe types and which had at least u_k demand in any pipe of type k.*

The problem reduces to layered facility location, just as in hierarchical placement, except that we need to meet the lower bound of u_k at each layer.

With this in mind, we define the *Load Balanced Facility Location* problem, where we have lower bounds on the demand any open facility must satisfy.

7 Load Balanced Facility Location

This problem is a variant of the classical facility location problem. We are given a network $G(V, E)$ with a distance function $d(\cdot)$ on the edges and a set of demand points. The cost of opening a facility at location i is f_i. In addition, there is a lower bound of L_i on the demand a facility opened at i must satisfy. The goal is to open facilities and allocate the demands to the open facilities so that an open facility at i has at least L_i demand routed to it. The cost of our solution is the sum of the average distance traveled by the demands and the cost of the open facilities. We wish to minimize this cost. We can write an integer program for this problem.

$$\text{Minimize} \sum_i \sum_j d_j c_{ij} x_{ij} + \sum_i f_i y_i$$

$$\begin{array}{rcll} \sum_i x_{ij} & \geq & 1 & \forall j \\ x_{ij} & \leq & y_i & \forall i, j \\ \sum_j d_j x_{ij} & \geq & L_i y_i & \forall i \\ x_{ij}, y_i & \in & \{0,1\} & \forall i, j \end{array}$$

Definition 7.1 *An approximation algorithm for load balanced facility location is a (α, β) approximation for some $\alpha \geq 1$ and $\beta \geq 1$ if the cost of the solution is within α times the optimal cost and facility i, if opened, serves at least $\frac{L_i}{\beta}$ demand.*

Let us denote by r the best known approximation ratio for classical facility location. We present a $(2r, 3)$ approximation to this problem. This generalizes to a $(\frac{1+\alpha}{1-\alpha} r, \frac{1}{\alpha})$ approximation for $\alpha < 1$. Unlike facility location [23, 12, 16], the lower bound makes it hard to round the linear relaxation directly. This arises from the fact that the filtering steps of Lin and Vitter in [19] do not work. Thus fractional solutions cannot be rounded by previous approaches.

Algorithm and Analysis

Firstly, we construct a regular facility location instance from this problem. Each potential facility location i is now assigned a new cost f'_i, which is the sum of f_i and the minimum cost of routing exactly L_i amount of demand to that location. For doing this, we take demand points in increasing order of distance to i till we have collected L_i amount of demand.

Lemma 7.1 *Consider any feasible fractional solution to the load balanced facility location problem of cost C. We can construct a feasible instance of the regular facility location problem of cost at most $2C$.*

Proof: Look at any fractional facility i. Since the feasible solution is routing at least L_i amount of demand to any open facility, the facility cost we assign in the new problem is at most the routing cost of the demand connected to that facility. Thus the total additional facility cost is at most C. ■

We now solve the facility location instance mentioned above. The cost of the solution we obtain is within a factor of $r = 1.728$ to the optimal solution for that instance.

Therefore the total cost in the solution we compute is bounded in terms of the routing cost of the original fractional solution to within a factor of $2r$. Also note that facility location guarantees that each demand point goes to the closest open facility.

We now consider the open facilities in arbitrary order. Suppose facility i serves less than $\frac{L_i}{3}$ amount of demand, we close the facility and route the demands it serves to their closest open facilities. At the end of this process, we are guaranteed that each open facility i serves at least $\frac{L_i}{3}$ amount of demand, and each demand goes to the closest open facility.

We have to show that removing a facility does not increase the total facility plus routing cost of the solution. For this, we show a feasible way to route the demands it serves so that the cost does not increase.

Lemma 7.2 *Removing a facility i serving less than $\frac{L_i}{3}$ amount of demand cannot increase the cost of our solution.*

Proof: Suppose we are closing facility i. Consider the closest demand point j which does not send demand to this facility. Suppose $d(i, j) = D$, where d is the distance metric. If j is being served by i', $d(i', j) < D$, as each demand point goes to the closest open facility.

Also, the facility cost $f'_i \geq \frac{2L_i}{3} D$. This follows because the facility serves only $\frac{L_i}{3}$ amount of demand, while the facility cost f'_i is f_i plus the cost of serving at least L_i units of demand.

When we close the facility, we can afford to use f'_i towards re-routing the demand it serves. We send the demand to i', the facility serving j. The extra cost for doing this is at most the cost of taking the demand from i to j and from there to i'. This distance is at most $2D$, and the demand is at most $\frac{L_i}{3}$, and so the total re-routing cost is at most $\frac{2L_i}{3} D$. ■

The above can be summarized in the following theorem,

Theorem 7.1 *The load balanced facility location problem has a $(2r, 3)$ approximation where each demand is served by its closest open facility.*

We can scale the facility costs to improve the lower bounds. We will state the following tradeoff theorem.

Theorem 7.2 *The load balanced facility location problem has a $(\frac{1+\alpha}{1-\alpha} r, \frac{1}{\alpha})$ approximation for $\alpha < 1$ where each demand is served by its closest open facility.*

Proof: We start off by adding λ times the cheapest way to serve L_i units of demand to facility i to its cost. It is immediate that the approximation is $((\lambda + 1)r, \frac{1}{\alpha})$, where $\lambda = \frac{2\alpha}{1-\alpha}$. ■

8 Access Network Design – The Algorithm

We combine the constant approximation for load balanced facility location with the hierarchical placement arguments from Section 3 to derive a constant factor approximation.

8.1 The Algorithm

Let $\sigma_i = \frac{\delta_i}{\delta_{i-1}}$ for $2 \leq i \leq K$. We can assume with a loss of $\frac{1}{\alpha}$ in the approximation ratio that all $\sigma_i \leq \alpha < 1$. Our algorithm will lay pipes in increasing order of types.

We use the layered facility location technique discussed in Section 3 to decide the routing with layer i pipes. Lemma 3.2 will apply in our case because the incremental cost of routing drops by at least α from one layer to the next.

As before, δ_i is the incremental cost of the ith pipe and ϕ_i is the fixed cost. Note that pipe i becomes profitable only when $u_i = \frac{\phi_i}{\delta_i}$ demand is being routed.

We solve Load Balanced Facility Location from the original demand points, using lower bound zero on the sink and u_i on all other nodes. Set $f_i = 0$ for all nodes. The cost along an edge is δ_{i-1} times its length. Denote the cost of the approximate solution as C_i. We assume the approximation ratio for Load Balanced Facility location is (r, γ). Therefore, we gather at least u_i/γ flow at all the non-sink facility nodes.

We define C_i^* to be the total incremental cost incurred by the optimal solution using pipes of type i. Note that the total cost of the optimal solution is $C^* > \sum_i C_i^*$.

Lemma 8.1 $C_i \leq r(\sum_{j=1}^{j=i-1} \alpha^{i-j-1} C_i^*)$

Proof: Consider the part of the optimal solution which uses pipes of type 1 through $i-1$. At the end of this solution, every node but the sink has either zero or at least u_i flow, because pipe type i must become profitable. By copying this solution, but using incremental cost δ_{i-1} everywhere, we obtain a solution to Load Balanced Facility Location with cost $\sum_{j=1}^{j=i-1} \alpha^{i-j-1} C_i^*$. The inequality claimed follows. ■

It is now easy to see the following.

Lemma 8.2 $\sum_i C_i \leq r \frac{1}{1-\alpha} C^*$

We will now construct our solution. We route from each demand point to the closest facility in the $i = 2$ solution as per our $i = 2$ solution, using pipe type one. We then route from each of these facilities to the closest facility of the $i = 3$ solution using pipe type two. We continue until we finally route from the facility of the $i = K$ solution to the sink using pipe type K. We need to bound the total cost of this solution.

Lemma 8.3 *The total fixed cost for the pipes used in our solution is bounded by* $\gamma(\sum_i C_i)(1+\alpha)$.

Proof: Consider the pipes used to route from the facilities of the $i = j$ solution to the facilities of the $i = j+1$ solution. Suppose the flow at each of the $i = j$ facilities were equal to the flow produced by facility location solution j. This means we have at least u_j/γ at each facility. If we were to route all of this flow backwards to the demand points using pipes of type j, we would pay a total incremental cost of αC_j. If we were then to route back upwards along the paths of solution $j+1$, we would pay incremental cost of C_{j+1}. Routing the demands back and forth fractionally can only be more expensive than routing the demands directly along shortest paths to the nearest facility. It follows that we can pay incremental cost $C_{j+1} + \alpha C_j$. Since we guaranteed at least u_j/γ at each of the facilities to begin with, and kept this flow together, we can conclude that the fixed cost payed for our pipes of type j is at most $\gamma(C_{j+1} + \alpha C_j)$. Of course, in our actual solution we will not have the correct amount of flow at each facility; however, the pipes we will buy will be exactly the same (shortest path pipe from each $i = j$ facility to nearest $i = j+1$). It follows that the total fixed cost is bounded by $\sum_{j=2}^{j=K} \gamma(C_{j+1} + \alpha C_j)$ to which we add the fixed cost for the pipes of type one (which is bounded by C_2 since we start with at least u_1 demand everywhere). The total fixed cost is at most $\gamma(\sum_i C_i)(1+\alpha)$. ■

Lemma 8.4 *The total incremental cost is bounded by* $\frac{1-\alpha}{1-2\alpha} \sum_i C_i$.

Proof: We define X_j to be the total incremental cost incurred in transferring from the facilities found in the $i = j$ solution to the facilities in the $i = j+1$ solution. $X_1 = C_1$ is the incremental cost incurred in transferring from the demand nodes to the $i = 2$ facilities.

Consider X_j. One way to route involves backing up the flow to the $j-1$ facilities at incremental cost αX_{j-1}, then back to the $j-2$ facilities at cost $\alpha^2 X_{j-2}$ and so on, until we reach the original demand points. From here we route along the paths given by the $i = j+1$ solution. Since simply transferring to the closest demand point must be cheaper than this, we can write $X_j \leq C_{j+1} + \sum_{i=1}^{i=j-1} \alpha^{j-i} X_i$. Summing this value over all j, we can bound the total incremental cost by

$$\sum_j X_j \leq \sum_i C_i + \frac{\alpha}{1-\alpha} \sum_j X_j$$

From this it follows that $\sum_j X_j \leq \frac{1-\alpha}{1-2\alpha} \sum_i C_i$. ■

Theorem 8.1 *We have constructed a constant approximation for Access Network Design.*

Proof: The total cost of our solution is bounded by the following.

$$C \leq (\gamma + \gamma\alpha + \frac{1-\alpha}{1-2\alpha}) \sum_i C_i \leq r(\gamma + \gamma\alpha + \frac{1-\alpha}{1-2\alpha})(\frac{1}{1-\alpha}) C^*$$

Inserting the additional α factor necessary to scale the pipe types properly yields an approximation ratio of:

$$C/C^* \leq r(\gamma \frac{1+\alpha}{\alpha(1-\alpha)} + \frac{1}{\alpha(1-2\alpha)})$$

If we set $\alpha = 1/3$ and $\gamma = 3$ and $r = 3.5$, this yields an approximation ratio of 94.5. ■

Open Problems and Acknowledgments

One open problem is to extend the approximation for Access Network Design to single sink buy-at-bulk. We also believe that the load balanced facility location problem has applications in network design with concave cost functions on edges. This connection needs to be explored further.

We would like to thank Matthew Andrews for explaining their work and pointing out difficulties in an earlier draft of this work, and Serge Plotkin for his support.

References

[1] K. Aardal, F. Chudak, and D.B. Shmoys. A 3-approximation algorithm for the k-level uncapacitated facility location problem. *Information Processing Letters*, 72:161–167, 1999.

[2] K. Aardal, M. Labbé, J. Leung, and M. Queyranne. On the two-level uncapacitated facility location problem. *INFORMS J. Comput.*, 8:289–301, 1996.

[3] M. Andrews and K. Munagala. Online algorithms for caching multimedia streams. *8th European Symposium on Algorithms*, 2000.

[4] M. Andrews and L. Zhang. The access network design problem. *39th IEEE Symposium on Foundations of Computer Science*, pages 40–49, 1998.

[5] B. Awerbuch and Y. Azar. Buy-at-bulk network design. *38th IEEE Symposium on Foundations of Computer Science*, pages 542–47, 1997.

[6] B. Awerbuch, A. Baratz, and D. Peleg. Cost-sensitive analysis of communication protocols. *9th ACM Symposium on Principles of Distributed Computing*, pages 177–87, 1990.

[7] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. *37th IEEE symposium on Foundations of Computer Science*, pages 184–193, 1996.

[8] Y. Bartal. On approximating arbitrary metrics by tree metrics. *30th ACM Symposium on Theory of Computing*, 1998.

[9] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: Evidence and implications. *INFOCOM*, 1999.

[10] M. Charikar, C. Chekuri, A. Goel, and S. Guha. Rounding via trees: Deterministic approximation algorithms for group steiner trees and k-median. *30th ACM Symposium on Theory of Computing*, 1998.

[11] M. Charikar, C. Chekuri, A. Goel, S. Guha, and S. Plotkin. Approximating a finite metric by a small number of tree metrics. *39th IEEE Symposium on Foundations of Computer Science*, 1998.

[12] M. Charikar and S. Guha. Improved combinatorial algorithms for facility location and k-median problems. *40th IEEE Symposium on Foundations of Computer Science*, 1999.

[13] C. Chiang, M. Liu, and M. Muller. Caching neighbourhood protocol: A foundation for building dynamic web caching hierarchies with proxy servers. *International Conference on Parallel Processing*, 1999.

[14] A. Dan and D. Sitaram. A generalized interval caching policy for mixed interactive and long video environments. *Multimedia Computing and Networking*, January 1996.

[15] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *9th ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[16] K. Jain and V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. *40th IEEE Symposium on Foundations of Computer Science*, 1999.

[17] L. Kaufman, M. vanden Eede, and P. Hansen. A plant and warehouse location problem. *Operations Research Quarterly*, 28:547–557, 1977.

[18] S. Khuller, B. Raghavachari, and N. Young. Balancing minimum spanning and shortest path trees. *Algorithmica*, 14(4):305–321, 1994.

[19] J.-H. Lin and J. S. Vitter. ϵ-approximations with minimum packing constraint violations. *24th ACM Symposium on Theory of Computing*, 1992.

[20] A. Meyerson, K. Munagala, and S. Plotkin. COST-DISTANCE: Two metric network design. *this proceedings.*

[21] A. Milo and O. Wolfson. Placement of replicated items in distributed databases. *Proceedings of International Conference on Extending Database Technology*, 1988.

[22] F. S. Salman, J. Cheriyan, R. Ravi, and S. Subramanian. Buy-at-bulk network design: Approximating the single-sink edge installation problem. *8th ACM-SIAM Symposium on Discrete Algorithms*, pages 619–628, 1997.

[23] D.B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems. *29th ACM Symposium on Theory of Computing*, 1997.

[24] I. Tatarinov, A. Rousskov, and V. Soloviev. Static caching of web servers. *6th International Conference on Computer Communications and Networks*, 1997.

[25] D. Tcha and B. Lee. A branch-and-bound algorithm for the multi-level uncapacitated location problem. *European J. Oper. Res.*, 18:35–43, 1984.

[26] T. Van Roy and D. Erlenkotter. A dual based procedure for dynamic facility location. *Management Sci.*, 28:1091–1105, 1982.

[27] D. Wessels. Configuring hierarchical squid caches. *National Laboratory for Advanced Network Research*, 1997.

[28] J. Zhang, R. Izmailov, D. Reininger, and M. Ott. Web caching framework: Analytical models and beyond. *IEEE workshop on Internet Applications*, 1999.