An Oblivious $O(1)$-Approximation for Single Source Buy-at-Bulk

Ashish Goel [*]
Stanford University

Ian Post [†]
Stanford University

August 26, 2009

Abstract

We consider the single-source (or single-sink) buy-at-bulk problem with an unknown concave cost function. We want to route a set of demands along a graph to or from a designated root node, and the cost of routing x units of flow along an edge is proportional to some concave, non-decreasing function f such that $f(0) = 0$. We present a polynomial time algorithm that finds a distribution over trees such that the expected cost of a tree for any f is within an $O(1)$-factor of the optimum cost for that f. The previous best simultaneous approximation for this problem, even ignoring computation time, was $O(\log |\mathcal{D}|)$, where $\mathcal{D}$ is the multi-set of demand nodes.

We design a simple algorithmic framework using the ellipsoid method that finds an $O(1)$-approximation if one exists, and then construct a separation oracle using a novel adaptation of the Guha, Meyerson, and Munagala [GMM01] algorithm for the single-sink buy-at-bulk problem that proves an $O(1)$ approximation is possible for all f. The number of trees in the support of the distribution constructed by our algorithm is at most $1 + \log |\mathcal{D}|$.

[*]Departments of Management Science and Engineering, and by courtesy, Computer Science, Stanford University. Email: ashishg@stanford.edu. Research supported by an NSF ITR grant and the Stanford-KAUST alliance for academic excellence.

[†]Department of Computer Science, Stanford University. Email: itp@stanford.edu. Research supported by an NSF ITR grant and the Stanford-KAUST alliance for academic excellence.

1 Introduction

We study the single-source (or single-sink) buy-at-bulk network design problem with an unknown concave cost function. We are given an undirected graph $G = (V, E)$ with edge lengths l_e and a set of demand nodes $\mathcal{D} \subseteq V$ with integer demands d_v and want to route these demands to a designated root node r as cheaply as possible, where the cost of routing along a particular edge is proportional to some function f of the amount of flow sent along the edge. In many applications it is natural to assume that f is a concave, non-decreasing function such that $f(0) = 0$, capturing the case where we benefit from some kind of economy of scale when aggregating flows together. We call such functions *aggregation functions* and define $\mathcal{F}$ as the set of all aggregation functions.

When the function f is given, the problem becomes the well-studied single-sink buy-at-bulk (SSBaB) problem. SSBaB is NP-hard, since it contains the Steiner tree problem as a special case. The problem was introduced by Salman et al. [SCRS97] who gave algorithms for special cases. Awerbuch and Azar [AA97] gave an $O(\log^2 n)$-approximation using metric tree embedding, which subsequently improved to $O(\log n)$ using better metric embeddings [Bar98, FRT03]. Building on their own work on hierarchical facility location [GMM00], Guha, Meyerson, and Munagala (GMM) gave the first constant-factor approximation [GMM01], an algorithm that features prominently in our results. Recent work [Tal02, GKR03, JR04, GI06] has reduced the approximation ratio to 24.92 and also provided an elegant cost-sharing framework for thinking about this problem.

However, for some applications we may want to assume that f is unknown or is known to vary over time. For instance, we may be aggregating observations in a sensor network where we do not know the amount of redundancy among different observations or where the redundancy is known to change. In this setting, it is desirable to find a solution that is robust to changes in f and provides a constant-factor approximation *simultaneously* for all $f \in \mathcal{F}$. Moreover, from a purely theoretical perspective, the existence of a good algorithm that is independent of f reveals non-trivial structure in the problem.

We will focus on randomized algorithms. Given the concavity of f, we may assume without loss of generality that the optimal routing graph is a tree. Let $\mathcal{T}$ be the set of all trees in G spanning $\mathcal{D}$ and r, and let T_f^* be the optimal tree for some fixed f. We use the shorthand $f(T)$ to denote the cost of T under f, i.e. $\sum_e l_e f(x_{T,e})$ where $x_{T,e}$ is the amount of flow tree T routes on edge e. There are two natural objectives which capture simultaneous approximation for multiple cost functions. First, we can try to minimize

$$R_1 = \max_{f \in \mathcal{F}} \frac{\mathbf{E}[f(T)]}{f(T_f^*)} \tag{1}$$

which essentially gives a distribution over trees such that in expectation, each function f is well-approximated. Second, and much more difficult, we can look for an algorithm that uses the objective

$$R_2 = \mathbf{E}\left[\max_{f \in \mathcal{F}} \frac{f(T)}{f(T_f^*)}\right] \tag{2}$$

A bound on (2) subsumes (1) and proves there exists a single tree that is simultaneously good for all f. We call R_1 the *oblivious* approximation ratio and R_2 the *simultaneous* approximation ratio. In this paper, we will work with the weaker, oblivious objective (1).

Both objectives have been studied in the literature. The tree embeddings used by Awerbuch and Azar [AA97] give an $O(\log^2 n)$ oblivious approximation, which was later reduced to $O(\log n)$ [Bar98, FRT03]. Goel and Estrin [GE03] improved this to $O(\log |\mathcal{D}|)$ and also prove the same bound on the stronger simultaneous objective. Gupta et al. [GHR06] achieve a $O(\log^2 n)$ oblivious approximation for a generalization where both the function and the demands are unknown. Khuller et al. [KRY95] studied special case of simultaneously approximating $f(x) = x$ and $f(x) = 1$ for $x \geq 1$, i.e. the shortest-path and Steiner trees, and

prove an $O(1)$ simultaneous approximation. These 2 functions constitute opposite extremes of functions in $\mathcal{F}$, and one may wonder if an $O(1)$ approximation for these 2 functions also works for all $f \in \mathcal{F}$ lying "in-between". However, it is not difficult to construct a graph and a set of demands such that the shortest-path and Steiner trees are identical, but this tree is an $\omega(1)$-approximation for other $f \in \mathcal{F}$. Enachescu et al. [EGGM05] achieve an $O(1)$ simultaneous value but only for grid graphs, assuming spatial correlation among nearby nodes. This naturally leads to the following questions:

Is $R_1 = O(1)$ achievable? If yes, is there a polynomial algorithm that guarantees $R_1 = O(1)$?

We answer both questions in the affirmative. We first write a simple LP formulation of the problem and show that using the ellipsoid method on the dual we can find an $O(1)$ approximation to the optimal ratio, whatever it happens to be for a given problem instance. We also show that given an appropriate separation oracle the optimum is constant and compute an explicit distribution over $1 + \lceil \log (\sum_v d_v) \rceil$ trees in polynomial time. This general approach is along the lines of small metric tree embeddings [CCG+98] and oblivious congestion minimization [Räc08].

Our key result is the construction of the necessary separation oracle subroutine, running in polynomial time, that proves a constant is achievable. We build our oracle around the GMM algorithm for SSBaB, using a modified analysis to solve a different problem in which we bound the cost of the GMM tree by a combination of different trees under different cost functions.

1.1 Organization of the Paper

In Section 2 we present an LP formulation and a framework using an approximate separation oracle that finds a constant-factor approximation to the optimal oblivious approximation ratio. In Section 3 we present our primary result, which proves the oblivious approximation ratio is constant and constructs the separation oracle required by Section 2 assuming some extra conditions on the input, and in Section 4 we complete the proof by showing those extra assumptions can be removed. We conclude with some open problems (including whether $R_2 = O(1)$ can be achieved).

2 LP Formulation and Algorithm Framework

Let R_1 be the worst-case optimal oblivious ratio, i.e.

$$R_1 = \max_{G,l,\mathcal{D},r} \min_{\mathcal{M}} \max_{f} \frac{\mathbf{E}_{T\sim\mathcal{M}}[f(T)]}{f(T_f^*)}$$

where $\mathcal{M}$ is a distribution over $\mathcal{T}$. In this section we discuss the problem of finding an $O(1)$-oblivious approximation if one exists.

By losing a factor of 2 in the approximation ratio we can restrict our analysis to a smaller class of aggregation functions. Let $D = 2^{\lceil \log(\sum_v d_v) \rceil}$, the total amount of demand rounded up to the nearest power of 2. We never route more than D flow on any edge, and d_v is integral, so we only care about $f(x)$ for integers $0 \le x \le D$. Suppose $f \in \mathcal{F}$, and $2^i < x < 2^{i+1}$. By the monotonicity of f, $f(2^i) \le f(x) \le f(2^{i+1})$, and by the concavity of f, $f(2^{i+1}) \le 2f(2^i)$, so with a loss of a factor of 2 we can interpolate between $f(2^i)$ and $f(2^{i+1})$ and assume f is piecewise linear with breakpoints only at powers of 2. Let $A_i(x) = \min\{x, 2^i\}$ and T_i^* the optimal aggregation tree for A_i. We call $A_i(x)$ the i-th atomic function following the terminology of Goel and Estrin [GE03], and it is easy to see that any $f \in \mathcal{F}$ that is linear between successive powers of 2 can be written as a linear combination of $\{A_i\}_{0 \le i \le \log D}$. Therefore, it suffices to design an algorithm $\mathcal{A}$ minimizing $\max_i \mathbf{E}_{\mathcal{A}}[A_i(T_{\mathcal{A}})]/A_i(T_i^*)$.

Our algorithm makes use of the standard SSBaB problem where f is known. We assume that f is given in the form of a set of K pipes $\{(\sigma_k, \delta_k)\}_{0 \le k \le K-1}$, where the cost of routing x flow on pipe k is equal to $\sigma_k + x\delta_k$. Then $f(x)$ is defined as the cost of using the cheapest pipe for x flow: $\min_k \sigma_k + x\delta_k$. We assume that $\sigma_0 \le \sigma_1 \le \cdots \le \sigma_{K-1}$, and by concavity we can assume $\delta_0 \ge \delta_1 \ge \cdots \ge \delta_{K-1}$. Define $u_k = \frac{\sigma_k}{\delta_k}$, the point at which the cost due to $\delta_k x$ begins to outweigh the cost due to σ_k. We call u_k the *capacity* of pipe k; the name arises from an alternate formulation (equivalent up to a factor of 2) of SSBaB where pipes have a fixed cost σ_k for a fixed capacity u_k. Let π_{BaB} be the best-known approximation ratio for SSBaB. Currently $\pi_{BaB} = 24.92$ using an algorithm by Grandoni and Italiano [GI06].

We also employ an approximation algorithm for a special case of SSBaB, the single-sink rent-or-buy (SSRoB) problem. Here $f(x)$ is characterized by 2 pipes: $(0, 1)$ and $(M, 0)$, i.e. we can pay x to route x flow or pay M to route any amount of flow. Let π_{RoB} be the best-known SSRoB approximation ratio. Eisenbrand et al. [EGRS08] give a 2.92-approximation.

If we can calculate $A_i(T)$ and $A_i(T_i^*)$ for every i and $T \in \mathcal{T}$ then the following linear program finds the optimal distribution of trees.

$$\begin{array}{lll} \min & \theta & \\ \text{s.t.} & & \sum_{T \in \mathcal{T}} x_T \quad \ge 1 \\ & \forall 0 \le i \le \log D, & \theta A_i(T_i^*) - \sum_{T \in \mathcal{T}} x_T A_i(T) \quad \ge 0 \\ & & x, \theta \quad \ge 0 \end{array} \tag{3}$$

In other words, we want a distribution $\{x_T\}_{T \in \mathcal{T}}$ of trees minimizing $\max_i \frac{\sum_T x_t A_i(T)}{A_i(T_i^*)}$. However, this approach is not directly tractable, as T_i^* is NP-hard to find, and $|\mathcal{T}|$ is exponentially large.

We solve an SSRoB approximation for each A_i to get $A_i(\tilde{T}_i)$—a π_{RoB}-approximation—and replace $A_i(T_i^*)$ with $A_i(\tilde{T}_i)$ in the constraints, so that all quantities in the LP are polynomial-time computable. Now consider the dual of (3), which is given by

$$\begin{array}{lll} \max & \beta & \\ \text{s.t.} & & \sum_{i=0}^{\log D} \alpha_i A_i(\tilde{T}_i) \quad \le 1 \\ & \forall T \in \mathcal{T} & \beta - \sum_{i=0}^{\log D} \alpha_i A_i(T) \quad \le 0 \\ & & \alpha, \beta \quad \ge 0 \end{array} \tag{4}$$

With an approximate separation oracle for the dual (4), we can approximate the solution in polynomial time using the ellipsoid method, and then transform it into an approximate solution to the primal (3). More formally:

Theorem 2.1. *With a randomized π_{BaB}-approximation to SSBaB, we can find a $2\pi_{RoB}\pi_{BaB}R_1$-approximation in expectation to the primal LP* (3) *that runs in polynomial time with high-probability.*

The proof uses a SSBaB approximation algorithm to construct an approximate separation oracle for (4). However, we will not prove this theorem because it is a special case of the following more general result, assuming that R_1 is a constant which will follow from Theorem 3.8.

Theorem 2.2. *If there exists a polynomial-time algorithm $\mathcal{A}$ and a given constant c such that $\forall\, \alpha_0, \ldots, \alpha_{K-1} \ge 0$, $\mathcal{A}$ finds $T_{\mathcal{A}}$ such that $\boldsymbol{E}_{\mathcal{A}}\left[\sum_i \alpha_i A_i(T_{\mathcal{A}})\right] \le c \sum_i \alpha_i A_i(T_i^*)$ then we can construct an algorithm that runs in polynomial-time with high probability, makes $O(\text{poly}(\log D))$ calls to $\mathcal{A}$ with high probability, and achieves an expected oblivious approximation ratio of $2c\pi_{RoB}$ using a distribution over $1 + \log D$ trees.*

Proving that such an algorithm $\mathcal{A}$ exists for a constant c is the primary result of this paper and is discussed in sections 3 and 4.

Remark 2.3. *If $\mathcal{A}$ is deterministic then the algorithm always runs in polynomial time and the expected ratio is $c\pi_{RoB}$, and if it is randomized then the algorithm runs in polynomial time with high probability and the expected ratio is $2c\pi_{RoB}$. For randomized $\mathcal{A}$ the ratio can also be reduced to $(1+\epsilon)c\pi_{RoB}$ with a $\frac{1}{\epsilon}$-factor increase in the runtime.*

Proof of Theorem 2.2. Let $A_i(\tilde{T}_i)$ be a π_{RoB}-approximation to $A_i(T_i^*)$ as above. We construct an approximate separation oracle $\mathcal{S}(\vec{\alpha}, \beta)$ for the dual (4) as follows:

1. Check if $\sum_i \alpha_i A_i(\tilde{T}_i) > 1$. If so, we have a violated constraint and are done.
2. Run $\mathcal{A}(\vec{\alpha})$ until it returns a tree T such that $\sum_i \alpha_i A_i(T) < 2c \sum_i \alpha_i A_i(\tilde{T}_i)$.
3. If $\sum_i \alpha_i A_i(T) < \beta$, return T. Otherwise, return feasible.

For a fixed β, let $\mathcal{P}_\beta$ be the polytope defined by $\sum_i \alpha_i A_i(\tilde{T}_i) \leq 1$, and $\beta - \sum_i \alpha_i A_i(T) \leq 0$ for all $T \in \mathcal{T}$. We run the following procedure to find the desired distribution of trees:

1. Run the ellipsoid method to check the feasibility of $\mathcal{P}_{2c}$, starting with the initial bounding box $0 \leq \alpha_i \leq 1 \; \forall i$ and using $\mathcal{S}$ as the separation oracle. It will terminate as infeasible.
2. Let $\mathcal{C}$ be the set of constraints returned by $\mathcal{S}$ proving $\mathcal{P}_{2c}$ is infeasible. It consists of $\sum_{i=0}^{\log D} \alpha_i A_i(\tilde{T}_i) \leq 1$, and $2c - \sum_{i=0}^{\log D} \alpha_i A_i(T) \leq 0$ for T in some subset of trees $\mathcal{T}'$.
3. In the dual LP (2), restrict the constraints to $\mathcal{C}$, and take the dual to get

$$\begin{array}{lll} \min & \theta & \\ \text{s.t.} & & \sum_{T \in \mathcal{T}'} x_T \geq 1 \\ & \forall 0 \leq i \leq \log D, & \theta A_i(\tilde{T}_i) - \sum_{T \in \mathcal{T}'} x_T A_i(T) \geq 0 \\ & & x, \theta \geq 0 \end{array} \tag{5}$$

4. Find a vertex optimal solution to (5), and return the distribution $\{x_T^*\}$.

First, we claim that $\mathcal{S}(\vec{\alpha}, \beta)$ will find a violated constraint whenever $\beta \geq 2c$ and will do so in polynomial time with high probability. If $\sum_i \alpha_i A_i(\tilde{T}_i) \leq 1$ is violated, then we are done. If not, we know $\mathcal{A}(\vec{\alpha})$ finds $T_{\mathcal{A}}$ such that

$$\mathbf{E}_{\mathcal{A}} \left[\sum_i \alpha_i A_i(T_{\mathcal{A}}) \right] \leq c \sum_i \alpha_i A_i(T_i^*) \leq c \sum_i \alpha_i A_i(\tilde{T}_i) \leq c$$

By Markov's inequality $\Pr_{\mathcal{A}} \left[\sum_i \alpha_i A_i(T_{\mathcal{A}}) \geq 2c \sum_i \alpha_i A_i(\tilde{T}_i) \right] \leq \frac{1}{2}$, so with high probability $O(\log n)$ invocations of $\mathcal{A}$—each running in polynomial time—suffice in step 2 of $\mathcal{S}$ to find a T satisfying $\sum_i \alpha_i A_i(T) < 2c \sum_i \alpha_i A_i(\tilde{T}_i)$. Now if $\beta \geq 2c$, the constraint $\beta - \sum_i \alpha_i A_i(T) \leq 0$ is violated.

With the necessary separation oracle, the ellipsoid algorithm can solve feasibility of $\mathcal{P}_\beta$ in $O(\text{poly}(\log D))$ iterations, so using $\mathcal{S}$ it will conclude $\mathcal{P}_{2c}$ is infeasible[1]. The set of constraints $\mathcal{C}$ returned by $\mathcal{S}$ during the execution constitutes a proof of infeasibility, and $\mathcal{C}$ consists of $\sum_{i=0}^{\log D} \alpha_i A_i(\tilde{T}_i) \leq 1$, and $\beta - \sum_{i=0}^{\log D} \alpha_i A_i(T) \leq 0$ for each T in some set of trees $\mathcal{T}'$.

Consider writing (4) with only the constraints in $\mathcal{C}$. Taking the dual yields (5), which only has variables x_T for $T \in \mathcal{T}'$. The ellipsoid algorithm concluded $\mathcal{P}_{2c}$ is infeasible after $O(\text{poly}(\log D))$ iterations, so $|\mathcal{T}'|$ is only polynomially-large in the input size, implying we can solve (5) exactly in polynomial time.

[1] In practice $\mathcal{A}$ may find violated constraints for $\beta < 2c$, and we can do binary search to find the smallest infeasible β. However, we cannot improve the provable guarantee beyond $\beta = c$, and this comes at a cost to the runtime.

Find a vertex-optimal solution θ^*, x_T^* to (5). The constraints in $\mathcal{C}$ are enough to restrict the optimal dual objective to be at most $2c$, so by duality $\theta^* \leq 2c$. Therefore, for all i

$$\sum_{T \in \mathcal{T}'} x_T^* A_i(T) \leq \theta^* A_i(\tilde{T}_i) \leq 2cA_i(\tilde{T}_i) \leq 2c\pi_{RoB} A_i(T_i^*)$$

Divide by $A_i(T_i^*)$ to get the oblivious ratio:

$$\max_i \frac{\sum_T x_T^* A_i(T)}{A_i(T_i^*)} \leq 2c\pi_{RoB}$$

Moreover, we claim $\{x_T^*\}$ is a distribution over only $1 + \log D$ trees. The LP (5) has $|\mathcal{T}'| + 1$ variables and $2 + \log D$ constraints, and the vertex-optimal solution θ^*, x_T^* must have $|\mathcal{T}'| + 1$ tight constraints, implying at least $|\mathcal{T}'| - \log D - 1$ non-negativity constraints must be tight. We know θ^* is positive, so only at most $1 + \log D$ of the variables x_T can be non-zero. □

3 The Separation Oracle Subroutine $\mathcal{A}$

By Theorem 2.1 we can find an $O(1)$-approximation to R_1, whatever it may be, but it remains to prove that this optimal ratio is a constant. In this section we construct the procedure $\mathcal{A}$ required by Theorem 2.2 using the GMM algorithm for SSBaB.

Our contribution is adapting a special case of the analysis of the GMM algorithm, namely those cases that arise when $f(x) = \sum_i \alpha_i A_i(x)$, to solve a different problem–that of bounding the cost of the output by $\sum_i \alpha_i A_i(T_i^*)$ rather than $f(T_f^*)$. The GMM algorithm and proof works in stages and bounds the cost of the pipes laid in each stage by a different chunk of the optimal tree T_f^*. On the other hand, in our proof we bound the cost of each stage by the cost of a *different* tree evaluated under a *different* cost function.

3.1 Background: The GMM Algorithm

For completeness, we summarize the GMM algorithm and the key lemmas and definitions. See the original paper [GMM01] for a thorough treatment. We are given a graph, demands $\mathcal{D}$, and pipes $\{(\sigma_k, \delta_k)\}_{k \in [K]}$ as described in Section 2. We assume the costs of successive pipes differ "significantly": for some constant γ such that $0 < \gamma < \frac{1}{2}$, we have that $\delta_{k+1} < \gamma\delta_k$ and $\sigma_k < \gamma\sigma_{k+1}$. For the SSBaB problem, it is easy to satisfy these constraints for arbitrary pipes with only an $O(1)$-factor loss. For our problem, it is harder but still possible, and this is discussed in Section 4.

We define g_k as the indifference point between pipe k and $k+1$, which is the solution to the equation $\sigma_k + \delta_k g_k = \sigma_{k+1} + \delta_{k+1} g_k$, and we define b_k as the solution to $\sigma_{k+1} + \delta_{k+1} b_k = 2\gamma(\sigma_k + \delta_k b_k)$, which we interpret as the point at which pipe $k+1$ becomes "significantly" cheaper than pipe k. It is easy to see that $u_k \leq b_k \leq u_{k+1}$ for all k.

The algorithm uses $O(1)$-approximations for Steiner tree and load-balanced facility location (LBFL), a generalization of the standard facility location problem. In the LBFL problem we have a graph and demands as in SSBaB, a facility cost F_v for each node v, and a lower bound L_v on the demand that a facility at v must service. The objective is to choose facilities and routing paths so as to minimize the sum of the cost of the open facilities and the distances traveled by the demands to a servicing facility. To approximate the LBFL we must relax the lower bound. Using [GMM00] we can approximate the optimal LBFL cost to within $2\pi_F$ while reducing the lower bound by a factor of at most 3. Here π_F denotes the best approximation to the normal facility location problem, currently $\pi_F = 1.52$ by Mahdian et al. [MYZ02]. We use π_S to denote the best approximation ratio for Steiner tree, currently 1.55 due to Robins and Zelikovsky [RZ00].

Now we can describe the GMM algorithm itself. At stage k, we lay pipe type k, and we break each stage into a Steiner tree step and a "shortest-path" tree step based on whether the cost of pipe k is dominated by the term σ_k or the term $\delta_k x$. The effective demands will also change each stage. Let $\mathcal{D}^{(k)}$ be the demand nodes at the start of stage k, and $d_v^{(k)}$ the stage k demand at $v \in \mathcal{D}^{(k)}$. Initially $\mathcal{D}^{(0)} = \mathcal{D}$.

1. *Steiner Tree:* Find a π_S-approximate Steiner tree on $\mathcal{D}^{(k)} \cup \{r\}$ with edge cost per unit length σ_k. Route all demands toward r. Cut the farthest-upstream edge with more than u_k flow, recalculate the flow, and repeat to get a forest with at least u_k flow at each root other than r and at most u_k flow on each edge.

2. *Consolidation:* Let t be a subtree not containing r and S_t the demand nodes in $\mathcal{D}^{(k)}$ it contains. Choose $v \in S_t$ with probability $\frac{d_v^{(k)}}{\sum_{u \in S_t} d_u^{(k)}}$ and route all demand in t back to v using pipe k.

3. *Shortest Path Tree:* Approximately solve a LBFL problem with facility lower bound b_k and edge cost per unit length δ_k on the *original* demands $\mathcal{D}$ (not $\mathcal{D}^{(k)}$ and $d_v^{(k)}$). This creates a forest of shortest-path trees with at least b_k flow at each root. If b_k demand does not exist, route everything to r.

4. *Consolidation:* Let t be subtree in the above forest servicing the demands S_t in $\mathcal{D}$. Choose $v \in S_t$ with probability $\frac{d_v}{\sum_{u \in S_t} d_u}$, and route the true, current demand $d_v^{(k)}$ in S_t back to v. Let $\mathcal{D}^{(k+1)}$ be the set of nodes chosen for consolidation and $d_v^{(k+1)}$ the demand at these nodes after consolidation.

Next, we mention the crucial lemmas in the GMM analysis used in our proof. See [GMM01] for the proofs.

Lemma 3.1 (GMM Lemma 4.1)**.** *Let $\hat{d}_v$ be the current demand at some $v \in \mathcal{D}$ immediately after any consolidation step. Then $\boldsymbol{E}[\hat{d}_v] = d_v$, i.e. the original demand.*

Using an algorithm that is a 3-approximation to the LBFL facility lower bounds, we have the following:

Lemma 3.2 (GMM Lemma 4.5)**.** *For every $v \in \mathcal{D}^{(k)}$, we have $\boldsymbol{E}[d_v^{(k)}] \geq \frac{b_{k-1}}{3}$.*

Define P_k^δ to be the *incremental* cost (due to δ) of the pipes laid in the *facility location* step in stage k and P_k^σ to be the *fixed* cost (due to σ) of the pipes laid in the *Steiner tree* step in stage k. All of the other costs incurred by the GMM algorithm can be bounded by P_k^δ and P_k^σ, so our analysis need only consider these quantities:

Lemma 3.3 (GMM Lemmas 4.2, 4.4, and 4.8)**.** *Let P_k^δ and P_k^σ as defined above. Then $\boldsymbol{E}[f(T_{GMM})] \leq 4 \sum_k \boldsymbol{E}[P_k^\delta + P_k^\sigma]$, where T_{GMM} is the final tree.*

3.2 Adapting the GMM Algorithm

From Theorem 2.2 we are given $\vec{\alpha}$ such that $\alpha_i \geq 0$, and $\sum_i \alpha_i A_i(\tilde{T}_i) \leq 1$. We want to find a tree T using the GMM algorithm such that $\sum_i \alpha_i A_i(T) \leq c \sum_i \alpha_i A_i(T_i^*)$. Define $L = \sum_i \alpha_i A_i(T_i^*)$, the *multi-level cost*, and $f(x) = \sum_i \alpha_i A_i(x)$, the concave cost function. Using this notation our objective becomes to find T such that $f(T) \leq cL$. Define K as the number of non-zero α_i, and for $0 \leq k \leq K-1$ define $p(k) = j$ where j is the index of the k-th non-zero α_i.

First, we claim that given $\vec{\alpha}$ we can define the pipes $\{(\sigma_k, \delta_k)\}$ used by the GMM algorithm, and given SSBaB pipes satisfying some minor conditions we can recover $\vec{\alpha}$. The following lemmas characterize the equivalence between the 2 types of parameters:

Lemma 3.4. *Given $\vec{\alpha}$ satisfying $\alpha_i \geq 0$ with K non-zero α_i, the SSBaB pipes $\{(\sigma_k, \delta_k)\}_{0 \leq k \leq K}$ defined by $\delta_k = \sum_{j \geq k} \alpha_{p(j)}$ and $\sigma_k = \sum_{j<k} \alpha_{p(j)} 2^{p(j)}$ define the function $f(x)$. That is, $f(x) = \sum_i \alpha_i A_i(x) = \min_k\{\sigma_k + \delta_k x\}$.*

Lemma 3.5. *Suppose we are given $K+1$ SSBaB pipes $\{(\sigma_k, \delta_k)\}_{0 \leq k \leq K}$ such that $\sigma_0 = 0$ and g_k is a power of 2 for all k. For $0 \leq k \leq K-1$, let $p(k) = \log g_k$, $\alpha_{p(k)} = \delta_k - \delta_{k+1}$, and $\alpha_j = 0$ whenever $j \neq p(k)$ for all k. Then $\sum_i \alpha_i A_i(x) = \min_k\{\sigma_k + \delta_k x\}$.*

Proof of Lemma 3.4. By definition $f(x) = \sum_k \alpha_{p(k)} A_{p(k)}(x)$. For any k, $f(x)$ is linear from $2^{p(k-1)}$ to $2^{p(k)}$ (we will assume $2^{p(-1)} = 0$ for consistency of notation), which will correspond to pipe k. For $x \in [2^{p(k-1)}, 2^{p(k)}]$, the functions $A_{p(0)}(x), \ldots, A_{p(k-1)}(x)$ have leveled off, and $A_{p(k)}(x), \ldots, A_{p(K-1)}(x)$ are growing at rate 1. Define δ_k as the slope of $f(x)$ in this interval: $\delta_k = \sum_{j \geq k} \alpha_{p(j)}$.

Now we can define σ_k to match $f(x)$ in the interval $[2^{p(k-1)}, 2^{p(k)}]$:

$$\begin{aligned}
\sigma_k + \delta_k 2^{p(k-1)} = \sum_i \alpha_i A_i(2^{p(k-1)}) &= \sum_{j<k} \alpha_{p(j)} 2^{p(j)} + \sum_{j \geq k} \alpha_{p(j)} 2^{p(k-1)} \\
&= \sum_{j<k} \alpha_{p(j)} 2^{p(j)} + \delta_k 2^{p(k-1)} \\
\Rightarrow \sigma_k &= \sum_{j<k} \alpha_{p(j)} 2^{p(j)}
\end{aligned}$$

We also add a $K+1$st pipe such that $\delta_K = 0$ and $\sigma_K = \sum_k \alpha_{p(k)} 2^{p(k)}$ to cover the interval after every $A_{p(k)}$ has leveled off. Now, we claim $f(x) = \min_j\{\sigma_j + \delta_j x\}$: for each k we know $f(x) = \sigma_k + \delta_k x$ whenever $x \in [2^{p(k-1)}, 2^{p(k)}]$ by our choice of δ_k and σ_k, and by the concavity of $f(x)$ for each j we have $\sigma_j + \delta_j x > f(x)$ when $x < 2^{p(j-1)}$ or $x > 2^{p(j)}$. Therefore no other pipe can be cheaper in this interval. Concavity also ensures that $\sigma_k < \sigma_{k+1}$ and $\delta_k > \delta_{k+1}$ for all k, yielding valid SSBaB pipes. □

Proof of Lemma 3.5. Let $K+1$ be the number of pipes, and $\delta_0 > \cdots > \delta_K$, $0 = \sigma_0 < \cdots < \sigma_K$. Since we never route more than D flow we may assume the cost function levels off at some $x \leq D$, so that $\delta_K = 0$. Define $p(k) = \log g_k$ for $0 \leq k \leq K-1$: when we change pipes at g_k the slope of $f(x)$ drops, which can occur only because the term $\alpha_{p(k)} A_{p(k)}(x)$ levels off. Recover $\alpha_{p(k)}$ by reversing the definitions in the proof of Lemma 3.4: we have $\delta_k = \sum_{j \geq k} \alpha_{p(j)}$, so for $k \leq K-1$ let $\alpha_{p(k)} = \delta_k - \delta_{k+1}$.

We now show by induction that $\sum_k \alpha_{p(k)} A_{p(k)}(x) = \min_j\{\sigma_j + \delta_j x\}$. For the base case $x \in [0, g_0]$, we have

$$\min_j\{\sigma_j + \delta_j x\} = \delta_0 x = (\delta_0 - \delta_K)x = \sum_{k=0}^{K-1} (\delta_k - \delta_{k+1})x = \sum_k \alpha_{p(k)} x = \sum_k \alpha_{p(k)} A_{p(k)}(x)$$

Now assume that for $x \in [0, g_{i-1}]$ that $\sum_k \alpha_{p(k)} A_{p(k)}(x) = \min_j\{\sigma_j + \delta_j x\}$. For $x \in (g_{i-1}, g_i]$, we know that $f(x) = \sigma_i + \delta_i x$. Therefore,

$$\begin{aligned}
\sigma_i + \delta_i x &= \left(\sigma_{i-1} + \delta_{i-1} 2^{p(i-1)}\right) + \delta_i (x - 2^{p(i-1)}) \\
&= \sum_k \alpha_{p(k)} A_{p(k)}(2^{p(i-1)}) + \sum_{k=i}^{K-1} (\delta_k - \delta_{k+1})(x - 2^{p(i-1)}) \\
&= \sum_{k<i} \alpha_{p(k)} A_{p(k)}(2^{p(i-1)}) + \sum_{k \geq i} \alpha_{p(k)} x \\
&= \sum_k \alpha_{p(k)} A_{p(k)}(x)
\end{aligned}$$

We use that pipes $i-1$ and i have equal cost at g_{i-1} in the first line and the induction hypothesis in the second line. □

We note that $\alpha_{p(k)}$ corresponds not to a particular SSBaB pipe, but to a breakpoint between pipes: when we switch from pipe k to $k+1$ at $2^{p(k)}$ flow, the slope of f drops from δ_k to δ_{k+1}, which is caused by the term $\alpha_{p(k)}A_{p(k)}(x)$ leveling off.

Given the above equivalence, we will use $\vec{\alpha}$ and $\{(\sigma_k, \delta_k)\}_k$ interchangeably for the remainder of the paper, using whichever representation is more convenient and converting from one form to another using Lemmas 3.4 and 3.5. However, the additional constraints that for some parameter $0 < \gamma < \frac{1}{2}$ we have $\delta_{k+1} < \gamma\delta_k$ and $\sigma_k < \gamma\sigma_{k+1}$ for all pipes k, will restrict the possible vectors $\vec{\alpha}$ that can be run through the algorithm:

Definition 3.6. *Call $\vec{\alpha}$ γ-*regular *if the pipes found using Lemma 3.4 satisfy $\delta_{k+1} < \gamma\delta_k$ and $\sigma_k < \gamma\delta_{k+1}$.*

We note the following constraints that γ-regularity imposes on $\vec{\alpha}$:

Lemma 3.7. *If $\delta_{k+1} < \gamma\delta_k$, then $\alpha_{p(k)} > (1-\gamma)\delta_k$ and $\alpha_{p(k)} > \frac{1-\gamma}{\gamma}\alpha_{p(k+1)}$.*

Proof. These follow immediately from $\alpha_{p(k)} = \delta_k - \delta_{k+1}$ and $\delta_{k+1} < \gamma\delta_k$. □

3.3 Approximation guarantee assuming regular $\vec{\alpha}$

We will first prove the existence of the separation oracle procedure $\mathcal{A}$ in Theorem 2.2 for γ-regular $\vec{\alpha}$ and later prove in Section 4 that arbitrary $\vec{\alpha}$ can be regularized with only an $O(1)$ change in $f(x)$ and L:

Theorem 3.8. *Let $\vec{\alpha}$ be γ-regular, and let $f(x) = \sum_i \alpha_i A_i(x)$, and $L = \sum_i \alpha_i A_i(T_i^*)$. Then the GMM algorithm finds a tree T_{GMM} such that $\boldsymbol{E}[f(T_{GMM})] = O(L)$.*

Roughly, our proof bounds the cost of the pipes laid in phase k of the algorithm by $\alpha_{p(k)}A_{p(k)}(T^*_{p(k)})$. Using Lemma 3.3 we concentrate on P_k^δ and P_k^σ and ignore the other costs. First, we bound the cost of the Steiner tree steps:

Lemma 3.9. *Let π_S be the approximation ratio for Steiner tree. Then we have $\sum_k \boldsymbol{E}[P_k^\sigma] \le \frac{3\pi_S}{1-\gamma}L$.*

Proof. We need to bound the cost of a Steiner tree spanning the current demands $\mathcal{D}^{(k)}$ with cost per unit length σ_k. If $k = 0$, then $\sigma_k = 0$ and we have nothing to bound, so assume $k > 0$.

We use the edges in $T^*_{p(k-1)}$. Note that it spans $\mathcal{D} \cup \{r\}$ and hence $\mathcal{D}^{(k)} \cup \{r\}$, and let $W_k \subseteq T^*_{p(k-1)}$ be the subset of edges spanning these nodes. By Lemma 3.2 each $v \in \mathcal{D}^{(k)}$ has aggregated at least $\mathbf{E}[d_v^{(k)}] \ge \frac{b_{k-1}}{3}$ demand. At the end of the previous LBFL phase, we chose a node v for consolidation from the set of all u routing to facility f with probability $\frac{d_v}{\sum_{u\to f} d_u} \le \frac{3d_v}{b_{k-1}}$. An edge is in W_k only if some $v \in \mathcal{D}^{(k)}$ routes through it, so by the union bound an edge carrying x_e^* demand in $T^*_{p(k-1)}$ is in W_k with probability at most $\frac{3x_e^*}{b_{k-1}}$.

The tree W_k pays σ_k for any amount of flow, whereas $T^*_{p(k-1)}$ pays $A_{p(k-1)}(x_e^*) = \min\{2^{p(k-1)}, x_e^*\}$ to send x_e^* flow on e. Then the cost of W_k is

$$\begin{aligned}
\mathbf{E}[W_k] &= \sigma_k \sum_e \Pr[e \in W_k] l_e = \sigma_k \sum_e \Pr[e \in W_k] l_e \frac{A_{p(k-1)}(x_e^*)}{\min\{x_e^*, 2^{p(k-1)}\}} \\
&\le \sigma_k \sum_{e: x_e^* \le 2^{p(k-1)}} \frac{3x_e^*}{b_{k-1}} \frac{A_{p(k-1)}(x_e^*)}{x_e^*} l_e + \sigma_k \sum_{e: x_e^* > 2^{p(k-1)}} 1 \cdot \frac{A_{p(k-1)}(x_e^*)}{2^{p(k-1)}} l_e \qquad (6) \\
&= 3\frac{\sigma_k}{b_{k-1}} \sum_{e: x_e^* \le 2^{p(k-1)}} A_{p(k-1)}(x_e^*) l_e + \frac{\sigma_k}{2^{p(k-1)}} \sum_{e: x_e^* > 2^{p(k-1)}} A_{p(k-1)}(x_e^*) l_e
\end{aligned}$$

We need to bound $\frac{\sigma_k}{b_{k-1}}$ and $\frac{\sigma_k}{2^{p(k-1)}}$. For the former term,

$$\frac{\sigma_k}{b_{k-1}} = \frac{\sigma_k(2\gamma\delta_{k-1} - \delta_k)}{\sigma_k - 2\gamma\sigma_{k-1}} \leq \frac{\sigma_k(2\gamma\delta_{k-1} - \delta_k)}{2\gamma\sigma_k(1-\gamma)} \leq \frac{2\gamma(\delta_{k-1} - \delta_k)}{2\gamma(1-\gamma)} = \frac{\alpha_{p(k-1)}}{1-\gamma}$$

using that $b_{k-1} = \frac{\sigma_k - 2\gamma\sigma_{k-1}}{2\gamma\delta_{k-1} - \delta_k}$ by definition, the γ-regularity constraints on σ_{k-1}, and the fact that $2\gamma < 1$. For the latter term,

$$\frac{\sigma_k}{2^{p(k-1)}} = \frac{\sigma_{k-1} + \alpha_{p(k-1)}2^{p(k-1)}}{2^{p(k-1)}} \leq \frac{\gamma\sigma_k + \alpha_{p(k-1)}2^{p(k-1)}}{2^{p(k-1)}} = \gamma\frac{\sigma_k}{2^{p(k-1)}} + \alpha_{p(k-1)}$$
$$\Rightarrow (1-\gamma)\frac{\sigma_k}{2^{p(k-1)}} \leq \alpha_{p(k-1)} \Rightarrow \frac{\sigma_k}{2^{p(k-1)}} \leq \frac{\alpha_{p(k-1)}}{1-\gamma}$$

using the formula for σ_k in Lemma 3.4 and γ-regularity.

Plug these into the final line in equation (6) above:

$$\begin{aligned}\mathbf{E}[W_k] \leq & \frac{\alpha_{p(k-1)}}{1-\gamma}\left(3\sum_{e:x_e^* \leq 2^{p(k-1)}} A_{p(k-1)}(x_e^*)l_e + \sum_{e:x_e^* > 2^{p(k-1)}} A_{p(k-1)}(x_e^*)l_e\right) \\ = & \left(\frac{3}{1-\gamma}\right)\alpha_{p(k-1)}A_{p(k-1)}(T^*_{p(k-1)})\end{aligned}$$

We lose another factor of π_S in approximating the Steiner tree. Sum over all k to bound $\sum_k \mathbf{E}[P_k^\sigma]$ by $\frac{3\pi_S}{1-\gamma}L$. □

Analyzing the LBFL step requires an additional lemma bounding the difference between g_k and b_k:

Lemma 3.10. *For every k, $g_k \leq b_k \leq \frac{1-2\gamma^2}{\gamma}g_k$.*

Proof. The bound $g_k \leq b_k$ follows from Lemma 3.5 in GMM [GMM01]. For the other inequality, from the definition of b_k and g_k we have

$$g_k = \frac{\sigma_{k+1} - \sigma_k}{\delta_k - \delta_{k+1}} \qquad b_k = \frac{\sigma_{k+1} - 2\gamma\sigma_k}{2\gamma\delta_k - \delta_{k+1}} \qquad \Rightarrow \frac{b_k}{g_k} = \frac{\sigma_{k+1} - 2\gamma\sigma_k}{\sigma_{k+1} - \sigma_k} \cdot \frac{\delta_k - \delta_{k+1}}{2\gamma\delta_k - \delta_{k+1}}$$

For the ratio of σ terms,

$$\begin{aligned}\frac{\sigma_{k+1} - 2\gamma\sigma_k}{\sigma_{k+1} - \sigma_k} &= \frac{\sigma_{k+1} - \sigma_k}{\sigma_{k+1} - \sigma_k} + (1-2\gamma)\frac{\sigma_k}{\sigma_{k+1} - \sigma_k} \\ &< 1 + (1-2\gamma)\frac{\sigma_k}{\left(\frac{1}{\gamma} - 1\right)\sigma_k} = 1 + \frac{\gamma - 2\gamma^2}{1-\gamma} = \frac{1-2\gamma^2}{1-\gamma}\end{aligned}$$

Similarly, for the δs,

$$\begin{aligned}\frac{\delta_k - \delta_{k+1}}{2\gamma\delta_k - \delta_{k+1}} &= \frac{2\gamma\delta_k - \delta_{k+1}}{2\gamma\delta_k - \delta_{k+1}} + (1-2\gamma)\frac{\delta_k}{2\gamma\delta_k - \delta_{k+1}} \\ &< 1 + (1-2\gamma)\frac{\delta_k}{(2\gamma - \gamma)\delta_k} = \frac{1-\gamma}{\gamma}\end{aligned}$$

Combining the 2 bounds,

$$\frac{b_k}{g_k} \leq \frac{1-2\gamma^2}{1-\gamma}\frac{1-\gamma}{\gamma} = \frac{1-2\gamma^2}{\gamma}$$

□

Now we can bound the LBFL cost $\mathbf{E}[P_k^\delta]$:

Lemma 3.11. *We have that $\sum_k \boldsymbol{E}[P_k^\delta] \leq 2\pi_F \frac{1-2\gamma^2}{\gamma-\gamma^2} L$ where π_F is the approximation ratio for the standard (non-load-balanced) facility location problem.*

Proof. In the shortest path tree step, the GMM algorithm solves an LBFL problem on the original demands $\mathcal{D}$ with facility lower bound b_k and edge cost per unit length δ_k. We will construct a feasible solution using the edges of $T^*_{p(k)}$. Orient the edges towards r, and find the farthest upstream (i.e. away from r) edge routing at least b_k flow. Cut the edge, and place a facility at the upstream node. Subtract this flow from downstream edges, and repeat the procedure. If we finish with less than b_k flow at the root node, we route each demand still reaching the root from its source vertex along the tree to the nearest existing facility (according to distances in $T^*_{p(k)}$). Let F_k be the resulting forest, and note that it has at least b_k flow at each facility.

For an edge e let x_e be the amount F_k routes on e when the demands $\mathcal{D}$ are routed, and x^*_e the amount that $T^*_{p(k)}$ routes on e. We now show that $x_e \leq x^*_e$. If we finish cutting $T^*_{p(k)}$ with at least b_k at the root then all flows are a subset of the flows in $T^*_{p(k)}$ so $x_e \leq x^*_e$. If we end up with too little demand for a facility in the final step then some of those demands will not be flowing downstream towards r in F_k. For each edge they take towards r, they are following the routing in $T^*_{p(k)}$, so $x_e \leq x^*_e$. For each e edge taken away from r, we are no longer following $T^*_{p(k)}$, but we must be moving upstream towards the nearest facility. This implies that in the tree $T^*_{p(k)}$ edge e carried more than b_k flow because all demand at the upstream facility flowed through e towards r. Since we are sending strictly less than b_k demand upstream we still have $x_e \leq x^*_e$.

The forest F_k never routes more than b_k flow, so $x_e \leq b_k$. When $x^*_e \leq g_k$, $x^*_e = A_{p(k)}(x^*_e)$, so $x_e \leq A_{p(k)}(x^*_e)$. Since $A_{p(k)}$ levels off at g_k, this may not hold for $x^*_e > g_k$, but by Lemma 3.10 $b_k \leq \frac{1-2\gamma^2}{\gamma} g_k$. Therefore $x_e \leq b_k \leq \frac{1-2\gamma^2}{\gamma} A_{p(k)}(x^*_e)$ when $x^*_e \geq g_k$.

Now let y_e be the flow F_k routes on edge e when the current, stage k demands $\mathcal{D}^{(k)}$ are used. By Lemma 3.1, $\mathbf{E}[\hat{d}_v] = d_v$ for each $v \in \mathcal{D}$. Summing over all the demands that contribute to an edge's flow, we have $\mathbf{E}[y_e] = x_e$.

The cost of F_k with δ_j cost per unit edge length is

$$\mathbf{E}\left[\delta_k \sum_e l_e y_e\right] = \delta_k \sum_e l_e x_e \leq \delta_k \sum_e l_e \left(\frac{1-2\gamma^2}{\gamma} A_{p(k)}(x^*_e)\right) \leq \left(\frac{\alpha_{p(k)}}{1-\gamma}\right)\left(\frac{1-2\gamma^2}{\gamma}\right) A_{p(k)}(T^*_{p(k)})$$

using $\frac{1-2\gamma^2}{\gamma} > 1$ and $\alpha_{p(k)} \geq (1-\gamma)\delta_k$ from Lemma 3.7.

We can find an approximate LBFL solution that is a $2\pi_F$-approximation to the optimal cost and reduces the facility lower bound by a factor of at most 3. Therefore

$$\mathbf{E}[P_k^\delta] \leq 2\pi_F \mathbf{E}[F_k] \leq \left(2\pi_F \frac{1-2\gamma^2}{\gamma-\gamma^2}\right) \alpha_{p(k)} A_{p(k)}(T^*_{p(k)})$$

Sum over all values of k to bound the expected cost by $2\pi_F \frac{1-2\gamma^2}{\gamma-\gamma^2} L$. □

Proof of Theorem 3.8. Combining the bounds in Lemmas 3.3, 3.11, and 3.9:

$$\mathbf{E}[f(T_{GMM})] \leq 4\left(2\pi_F \frac{1-2\gamma^2}{\gamma-\gamma^2} + \frac{3\pi_S}{1-\gamma}\right) L$$

□

This completes the analysis of $\mathcal{A}$ for γ-regular $\vec{\alpha}$. If arbitrary $\vec{\alpha}$ can be γ-regularized for some $0 < \gamma < \frac{1}{2}$ it follows that $R = O(1)$.

Recent algorithms for SSBaB are based on the Gupta, Kumar, and Roughgarden (GKR) algorithm [GKR03, GKPR07], which achieves a better approximation ratio than GMM with a simpler analysis, and one may wonder whether we could reap the same benefits by basing our proof around this algorithm instead. One round of GKR is roughly equivalent to one round of GMM—starting with about g_{k-1} demand at a subset of nodes and ending with about g_k demand at a smaller subset—but the GKR analysis bounds the entire cost of a round using only one tree, whereas GMM requires two. However, each tree required by GMM can be easily constructed from some T_i^* in $O(\alpha_i A_i(T_i^*))$, but building the tree needed by GKR and within the right bounds seems trickier. Note that Lemmas 3.9 and 3.11 use two different trees, $T^*_{p(k-1)}$ and $T^*_{p(k)}$, analyzed in two different ways, either fixed or linear cost per edge. Although this conveniently matches the GMM algorithm, it also required for the proof to work. Using only a single Steiner tree on a subset of the nodes as in GKR allows less flexibility, so a proof may require a different approach or more substantial changes to the original GKR analysis.

4 Handling Arbitrary $\vec{\alpha}$

Given any $\vec{\alpha}$, where $\alpha_i \geq 0$, defining $f(x)$, a concave cost function, and L, the multi-level cost, we need to find regular $\vec{\alpha}'$ defining $f'(x)$ and L' such that $f(x) = O(f'(x))\ \forall x$, and $L' = O(L)$. Then applying Theorem 3.8 to $\vec{\alpha}'$ gives $f'(T_{GMM}) = O(L')$, and

$$f(T_{GMM}) = O(f'(T_{GMM})) = O(L') = O(L)$$

satisfying the precondition of Theorem 3.8. Note that we can allow f to grow and L to shrink arbitrarily in the transformation to f' and L', but we need to bound increases in L and decreases in f. By scaling by $\sum_i \alpha_i$ we may assume without loss of generality that $\sum_i \alpha_i = 1$.

First, we prove a simple bound on the change between each term $A_i(T_i^*)$ in L.

Lemma 4.1. *For any i and any $k > 0$, $A_i(T_i^*) \leq A_{i+k}(T_{i+k}^*) \leq 2^k A_i(T_i^*)$.*

Proof. Note $A_i(x) \leq A_{i+k}(x) \leq 2^k A_i(x)$ for $k > 0$. Therefore

$$A_i(T_i^*) \leq A_i(T_{i+k}^*) \leq A_{i+k}(T_{i+k}^*) \leq A_{i+k}(T_i^*) \leq 2^k A_i(T_i^*)$$

□

To regularize the values we run $\vec{\alpha}$ through a series of three procedures, one for each of the following lemmas, each of which changes $\vec{\alpha}$ to satisfy an additional set of constraints. None of the procedures are conceptually difficult, but the details are quite intricate. We will state the lemmas, give a brief sketch of the ideas, and present the complete proofs in the appendix.

The first lemma is only a helper used in satisfying the σ constraints. The proof serves as a warmup for the later lemmas, which use similar ideas but are more involved.

Lemma 4.2. *Given arbitrary $\vec{\alpha}$, we can find $\vec{\alpha}'$ such that the corresponding f',L', δ', σ' satisfy $f(x) \leq f'(x)$, $L' \leq 2L$, and $\frac{\sigma'_{K-1}}{\delta'_{K-1}} \leq D$, where K is the number of pipes, and D is the total demand rounded up to a power of 2.*

The following 2 lemmas perform the actual regularization.

Lemma 4.3. *Given $\vec{\alpha}$ satisfying $\frac{\sigma_{K-1}}{\delta_{K-1}} \leq D$, we can find $\vec{\alpha}'$ such that the corresponding f',L', δ', σ' satisfy $f(x) \leq 3f'(x)$, $L' = O(L)$, $\frac{\sigma'_{K-1}}{\delta'_{K-1}} \leq D$, and $\delta'_{k+1} < \gamma\delta'_k$ for all k.*

Lemma 4.4. *Given $\vec{\alpha}$ satisfying $\frac{\sigma_{K-1}}{\delta_{K-1}} \leq D$ and $\delta_{k+1} < \gamma\delta_k$, we can find $\vec{\alpha}'$ such that such the corresponding f',L', δ', σ' satisfy $f(x) \leq \frac{5}{2}f'(x)$, $L' = O(L)$, $\delta'_{k+1} < \gamma\delta'_k$, and $\sigma'_k < \gamma\sigma'_{k+1}$ for all k.*

The proofs are based around the following idea: check if $\delta_{k+1} \geq \gamma\delta_k$ or $\sigma_k \geq \gamma\sigma_{k+1}$, and discard pipes that violate the constraints. The additional difficulty, relative to the analysis of GMM, arises from the special form that f must satisfy and the need to bound the increase in L. When we remove pipes in general the indifference points between subsequent pipes will no longer be powers of 2, so f can no longer be defined in terms of $\vec{\alpha}$. We fix this by modifying the parameters of an offending pipe until the new breakpoint is a power of 2. To avoid drastic changes in L or f, we achieve this by holding the cost of the given pipe k fixed at its indifference point with either $k-1$ of $k+1$ and "rotating" the line $\sigma_k + \delta_k x$ around this fixed point until the other indifference point is fixed.

Analyzing the increase in L caused by these procedures is the technical crux in the regularization analysis, as removing pipes can shift "α-mass" in the multi-level cost onto much more expensive trees. We consider each pipe removal and the terms in L it affects. If α-mass is shifted from $A_i(T_i^*)$ to $A_{i+l}(T_{i+l}^*)$, where $l = O(1)$, then the current chunk of L has increased by $O(1)$. If not, we show that the conditions requiring $l = \omega(1)$ imply there exist large terms in L above $i + l$ that can absorb the increase with only an $O(1)$-factor loss. We only charge against each L-term $O(1)$ times during the entire regularization, so the total increase is bounded by $O(1)$.

We summarize the consequences of the regularization procedure below:

Theorem 4.5. *The algorithm $\mathcal{A}$ required by Theorem 2.2 exists for a constant c, and the oblivious approximation ratio R_1 is constant.*

5 Open Problems

A number of interesting open problems remain to be solved. First, we have only achieved an $O(1)$-ratio for the objective $R_1 = \max_f \mathbf{E}[f(T)]/f(T_f^*)$, but Goel and Estrin [GE03] have shown an $O(\log|\mathcal{D}|)$-approximation for the much harder objective $R_2 = \mathbf{E}\left[\max_f f(T)/f(T_f^*)\right]$, proving there exists a single tree that is *simultaneously* an $O(\log|\mathcal{D}|)$-approximation for all $f \in \mathcal{F}$. Achieving a constant for this stronger objective or showing a lower bound remains an important open question.

Second, although our algorithm proves that an $O(1)$-approximate distribution exists, the ellipsoid algorithm tells us little about what these trees actually look like. A combinatorial algorithm that yields insight as to the actual structure of these trees would also be of interest. Third, we have made little attempt to optimize the constant c in the approximation ratio, and the resulting value is huge due to the regularization procedure. Shaving large factors off our bound on R_1 may be a simple question, and it would be particularly interesting to find an oblivious approximation algorithm that is competitive with standard SSBaB for known f.

References

[AA97] B. Awerbuch and Y. Azar. Buy-at-bulk network design. In *Proceedings of the 38th annual IEEE Symposium on Foundations of Computer Science*, 1997.

[Bar98] Y. Bartal. On approximating arbitrary metrices by tree metrics. In *Proceedings of the 30th annual ACM Symposium on Theory of Computing*, pages 161–168, 1998.

[CCG+98] M. Charikar, C. Chekuri, A. Goel, S. Guha, and S. Plotkin. Approximating a finite metric by a small number of tree metrics. *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 379–388, Nov 1998.

[EGGM05] M. Enachescu, A. Goel, R. Govindan, and R. Motwani. Scale-free aggregation in sensor networks. *Theoretical Computer Science*, 344(1):15–29, 2005.

[EGRS08] F. Eisenbrand, F. Grandoni, T. Rothvoß, and G. Schäfer. Approximating connected facility location problems via random facility sampling and core detouring. In *Proceedings of the 19th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1174–1183, 2008.

[FRT03] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the 35th annual ACM Symposium on Theory of Computing*, pages 448–455, 2003.

[GE03] A. Goel and D. Estrin. Simultaneous optimization for concave costs: single sink aggregation or single source buy-at-bulk. In *Proceedings of the 14th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 499–505, 2003.

[GHR06] A. Gupta, M.T. Hajiaghayi, and H. Räcke. Oblivious network design. In *Proceedings of the 17th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 970–979, 2006.

[GI06] F. Grandoni and G.F. Italiano. Improved approximation for single-sink buy-at-bulk. *Lecture Notes in Computer Science*, 4288:111, 2006.

[GKPR07] A. Gupta, A. Kumar, M. Pal, and T. Roughgarden. Approximation via cost sharing: Simpler and better approximation algorithms for network design. *J. ACM*, 2007.

[GKR03] A. Gupta, A. Kumar, and T. Roughgarden. Simpler and better approximation algorithms for network design. In *Proceedings of the 35th annual ACM Symposium on Theory of Computing*, pages 365–372, 2003.

[GMM00] S. Guha, A. Meyerson, and K. Munagala. Hierarchical placement and network design problems. In *Proceedings of the 41st annual IEEE Symposium on Foundations of Computer Science*, pages 603–612, 2000.

[GMM01] S. Guha, A. Meyerson, and K. Munagala. A constant factor approximation for the single sink edge installation problems. In *Proceedings of the 33rd annual ACM Symposium on Theory of Computing*, pages 383–388, 2001.

[JR04] R. Jothi and B. Raghavachari. Improved approximation algorithms for the single-sink buy-at-bulk network design problems. In *Proceedings of the 9th Scandinavian Workshop on Algorithm Theory*, pages 336–348, 2004.

[KRY95] S. Khuller, B. Raghavachari, and N. Young. Balancing minimum spanning trees and shortest-path trees. *Algorithmica*, 14(4):305–321, 1995.

[MYZ02] M. Mahdian, Y. Ye, and J. Zhang. Improved Approximation Algorithms for Metric Facility Location Problems. In *Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 229–242, 2002.

[Räc08] H. Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings of the 40th annual ACM Symposium on Theory of Computing*, pages 255–264, 2008.

[RZ00] G. Robins and A. Zelikovsky. Improved Steiner tree approximation in graphs. In *Proceedings of the 11th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 770–779, 2000.

[SCRS97] FS Salman, J. Cheriyan, R. Ravi, and S. Subramanian. Buy-at-bulk network design: Approximating the single-sink edge installation problem. In *Proceedings of the 8th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 619–628, 1997.

[Tal02] K. Talwar. The Single-Sink Buy-at-Bulk LP Has Constant Integrality Gap. In *Proceedings of the 9th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 475–486, 2002.

A Proofs of regularization lemmas

Lemma 4.2. *Given arbitrary $\vec{\alpha}$, we can find $\vec{\alpha}'$ such that the corresponding f',L', δ', σ' satisfy $f(x) \leq f'(x)$, $L' \leq 2L$, and $\frac{\sigma'_{K-1}}{\delta'_{K-1}} \leq D$, where K is the number of pipes, and D is the total demand rounded up to a power of 2.*

Proof. Let k be the first pipe such that $\frac{\sigma_k}{\delta_k} \geq D$. Note $k > 0$ since $\frac{\sigma_0}{\delta_0} = 0$. Remove all pipes above k. Now we modify the parameters of pipe k to satisfy the desired constraint. Increase δ_k, while decreasing σ_k so as to hold $\sigma_k + \delta_k 2^{p(k-1)}$ fixed, until $\frac{\sigma_k}{\delta_k} = D$. Geometrically, we are rotating the line $y = \sigma_k + \delta_k x$ counter-clockwise around the point $(2^{p(k-1)}, \sigma_k + \delta_k 2^{p(k-1)})$. Let δ'_k, σ'_k be the new parameters for pipe k. Let f' be the new cost function formed by modifying pipe k and removing pipes $k+1, \ldots, K-1$ and L' the associated multi-level cost.

Claim: The function $f'(x)$ is concave, and $f(x) \leq f'(x)$ for all x.

Initially $\delta_k < \delta_{k-1}$ and $\sigma_k > \sigma_{k-1}$, and we continuously decrease σ_k while increasing δ_k. We know $\sigma_{k-1} + \delta_{k-1} 2^{p(k-1)} = \sigma'_k + \delta'_k 2^{p(k-1)}$, so if we decrease σ'_k to σ_{k-1} the modified pipe k will match pipe $k-1$. However, we have that $\frac{\sigma_{k-1}}{\delta_{k-1}} < D = \frac{\sigma'_k}{\delta'_k}$, so we stop before reaching that point. Therefore $\sigma'_k > \sigma_{k-1}$ and $\delta'_k < \delta_{k-1}$, which implies $f'(x)$ is concave since the switchover between pipes $k-1$ and k is unchanged. We only increased the rate of growth for $x \geq 2^{p(k-1)}$, so $f'(x) \geq f(x)$ for all x.

Claim: The new multi-level cost L' is at most $2L$.

There is a term $\alpha_{p(j)}$ for each changeover between pipes as well as the implicit breakpoint at D when f levels off. Increasing δ_k and removing pipes $k+1, \ldots, K-1$ so that pipe k is used all the way to D corresponds in L to pushing α-mass from the terms $\alpha_{p(k-1)} A_{p(k-1)}(T^*_{p(k-1)}) + \cdots + \alpha_{p(K-1)} A_{p(K-1)}(T^*_{p(K-1)})$ onto the term $\delta'_k A_{\log D}(T^*_{\log D})$ because $p'(k) = \log D$.

By the definition of σ'_k and δ'_k and Lemma 3.4 we have

$$\delta'_k = \frac{\sigma'_k}{D} = \sum_{j<k} \alpha'_{p'(j)} \frac{2^{p'(j)}}{D}$$

The terms $\alpha_{p(0)}, \ldots \alpha_{p(k-2)}$ are unchanged, and $\alpha_{p(k-1)}$ drops due the decreased difference between δ_{k-1} and δ_k. There are no non-zero α'_i between $p(k-1)$ and $\log D$. This gives us

$$\delta'_k = \sum_{j<k} \alpha'_{p'(j)} \frac{2^{p'(j)}}{D} \leq \sum_{j<k} \alpha_{p(j)} \frac{2^{p(j)}}{D}$$

Next we use Lemma 4.1 to relate $\frac{2^{p(j)}}{D} A_{p(j)}(T^*_{p(j)})$ and $A_{\log D}(T^*_{\log D})$:

$$\delta'_k A_{\log D}(T^*_{\log D}) \leq \sum_{j<k} \alpha_{p(j)} \frac{2^{p(j)}}{D} A_{\log D}(T^*_{\log D}) \leq \sum_{j<k} \alpha_{p(j)} A_{p(j)}(T^*_{p(j)}) \leq L$$

Finally, $L' = \sum_{j<k} \alpha'_{p(j)} A_{p(j)}(T^*_{p(j)}) + \delta'_k A_{\log D}(T^*_{\log D}) \leq 2L$.

□

Lemma 4.3. *Given $\vec{\alpha}$ satisfying $\frac{\sigma_{K-1}}{\delta_{K-1}} \leq D$, we can find $\vec{\alpha}'$ such that the corresponding f',L', δ', σ' satisfy $f(x) \leq 3f'(x)$, $L' = O(L)$, $\frac{\sigma'_{K-1}}{\delta'_{K-1}} \leq D$, and $\delta'_{k+1} < \gamma\delta'_k$ for all k.*

Proof. We repeat the following two steps until $\delta_{k+1} < \gamma\delta_k$ for all k.

1. *Deletion Step:* The basic idea here is the same as that used by GMM Lemma 3.2 [GMM01] to satisfy the constraints on the δ's: whenever a pipe violates the constraint $\delta_{k+1} \geq \gamma\delta_k$, we remove the pipe.

   Let k be the smallest index such that $\delta_{k+1} \geq \gamma\delta_k$, and let l be the smallest integer such that $\delta_{k+l} < \frac{\gamma}{3}\delta_k$. If such an l exists, then remove pipes $k+1, \ldots, k+l-1$, and change $f(x)$ in the interval $[2^{p(k)}, 2^{p(k+l-1)}]$ by using the cheaper of pipe k and $k+l$. If no such l exists then remove all pipes above k, and replace them with pipe k. Note that this does not break the condition set in Lemma 4.2.

2. *Rotation Step:* Pipes k and $k+l$ now have equal cost at some point g, but g may not be a power of 2, in which case $f(x)$ is no longer in the form $\sum_i \alpha_i A_i(x)$, and $\vec{\alpha}'$ is no longer defined.

   We want to modify the pipes to change g while not affecting L or f too much. As in Lemma 4.2, we hold the cost of pipe k fixed when routing $2^{p(k-1)}$ flow (where we switch from $k-1$ to k), and reduce δ_k until pipes k and $k+l$ meet at the next power of 2, increasing σ_k to maintain k's cost at $2^{p(k-1)}$. This corresponds to rotating the line $y = \sigma_k + \delta_k x$ clockwise around the point $(2^{p(k-1)}, \sigma_k + \delta_k 2^{p(k-1)})$. Let δ'_k and σ'_k be the new parameters for pipe k. Note that $f'(x)$ now has the proper structure again, and $\vec{\alpha}'$ and L' are well-defined. We never increase σ_0 above 0 since we hold this point fixed when adjusting pipe 0.

First, we bound the change to δ_k in the rotation step. This allows us to prove that the constraints on the δ's are satisfied, and $f(x)$ decreases by at most an $O(1)$-factor.

Claim: After rotation $\delta'_k \geq \frac{\delta_k}{3}$.

Before adjustment, we are indifferent between k and $k+l$ at (g, y_k) where $y_k = \sigma_k + \delta_k g = \sigma_{k+l} + \delta_{k+l} g$. The difference in costs between k and $k+l$ at $2^{p(k-1)}$ flow remains unchanged because we hold the cost of pipe k fixed at $2^{p(k-1)}$. Let $x_k = g - 2^{p(k-1)}$, the distance after $2^{p(k-1)}$ at which their costs are equal. Before rotation, the pipes' costs approach each other at a rate of $\delta_k - \delta_{k+l}$. If we reduce δ_k by a factor of 3, then $\frac{\delta_k}{3} - \delta_{k+l} \leq \frac{1}{3}(\delta_k - \delta_{k+l})$, so it takes at least $3x_k$ for pipe k to grow

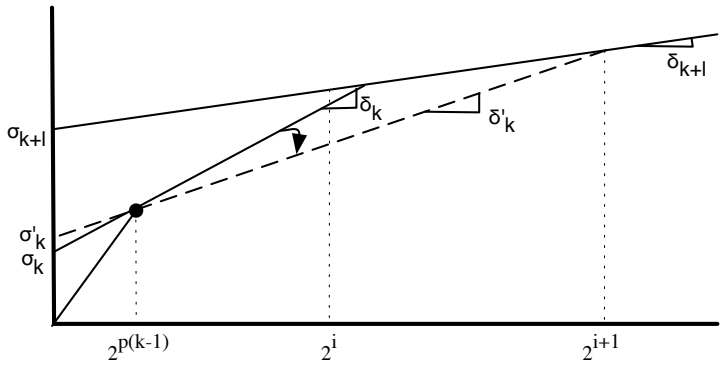

Figure 1: To ensure the indifference point between pipes k and $k+l$ is a power of 2 we "rotate" pipe k around it's starting point until it meets $k+l$ at a power of 2.

from $\sigma_k + \delta_k 2^{p(k-1)}$ to y_k, during which pipe $k+l$'s cost only increases, so pipe k does not surpass $k+l$ until after $2^{p(k-1)} + 3x_k$.

The original pipe k met pipe $k+1$ (now removed) at some point $2^{p(k)} \geq 2^{p(k-1)+1}$ before meeting $k+l$ at g. Therefore $g \geq 2^{p(k-1)+1}$, which implies $x_k = g - 2^{p(k-1)} \geq \frac{g}{2}$. After reducing δ_k to $\frac{\delta_k}{3}$, pipes k and $k+l$ now meet after $2^{p(k-1)} + 3x_k = g + 2x_k \geq 2g$. There must be a power of 2 between g and $2g$, and we reduce δ_k only until we hit the next power of 2, so $\delta'_k \geq \frac{\delta_k}{3}$.

Claim: When the procedure is finished $\delta'_{k+1} < \gamma\delta'_k$ for all k.

By the choice of l, $\delta_{k+l} < \frac{\gamma}{3}\delta_k \leq \gamma\delta'_k$, using the previous claim. Further $\delta'_k < \delta_k < \gamma\delta_{k-1}$, so no previously-satisfied constraints are broken. We renumber the pipes, and repeat the process for the next constraint violation. When we are done, all the remaining pipes will satisfy $\delta'_{k+1} < \gamma\delta'_k$.

Claim: For all x, $f(x) \leq 3f'(x)$.

Note that removing pipes $k+1, \ldots, k+l-1$ only changes f in the interval $(2^{p(k-1)}, 2^{p(k+l-1)})$, and we only remove or adjust pipes in this interval once. Initially, removing pipes can only increase $f(x)$, but then we reduce δ_k by a factor of at most 3, which may decrease $f(x)$ by a factor of at most 3.

Now, we must bound the potential increase in L. To avoid confusion due to relabeling indexes after removing pipes, we change notation slightly. Suppose the procedure completes after K' iterations. Let $\alpha'_{p'(0)}, \ldots, \alpha'_{p'(K'-1)}$ be the final non-zero α's, and $\alpha_{p(0)}, \ldots, \alpha_{p(K-1)}$ the original α's. For $0 \leq k \leq K'-1$ let $\alpha_{p(s_k)}, \ldots, \alpha_{p(s_{k+1}-1)}$ be the L-terms affected by the kth iteration of the procedure: either they are removed and merged into $\alpha'_{p'(k)}$ or $\alpha'_{p'(k)} = \alpha_{p(s_k)}$ if the constraint is already satisfied. We need to analyze how mass is shifted between terms in L. Define $L_k = \sum_{i=s_k}^{s_{k+1}-1} \alpha_{p(i)} A_{p(i)}(T^*_{p(i)})$, the portion of L that round k affects.

Consider round k in which we remove old pipes $s_k + 1, \ldots, s_{k+1} - 1$ and adjust δ'_k. The old $\delta_{s_{k+1}}$ becomes δ'_{k+1}. Rotating δ'_k increases $\alpha'_{p'(k-1)}$ because $\alpha'_{p'(k-1)} = \delta'_{k-1} - \delta'_k$ but reduces the total α-mass above $p'(k-1)$ because $\delta'_k = \sum_{j \geq k} \alpha'_{p(j)}$, decreasing L. The remaining α-mass on $\alpha_{p(s_k)} A_{p(s_k)}(T^*_{p(s_k)}), \ldots,$ $\alpha_{p(s_{k+1}-1)} A_{p(s_{k+1}-1)}(T^*_{p(s_{k+1}-1)})$ merges into $\alpha'_{p'(k)} A_{p'(k)}(T^*_{p'(k)})$ where $p'(k)$ is somewhere between $p(s_k)$ and $p(s_{k+1})$. If mass from some $\alpha_{p(i)}$ moves down to $\alpha'_{p'(k)}$ where $p'(k) < p(i)$, then we can ignore it, as it will only reduce L. If it moves up, then we will charge the increase to some higher term in L.

Let $c_\delta < \frac{\gamma}{3}$ be some small constant. There are 2 cases to consider: either $\delta_{s_{k+1}} \geq c_\delta \delta'_k$ or $\delta_{s_{k+1}} < c_\delta \delta'_k$.

Case 1: $c_\delta \delta'_k > \delta_{s_{k+1}} = \delta'_{k+1}$.

Intuitively, this means there is a big drop between $\delta_{s_{k+1}-1} \geq \frac{\gamma}{3}\delta'_k$ and $\delta_{s_{k+1}} < c_\delta \delta'_k$, so $\alpha_{p(s_{k+1}-1)}$ must be fairly large: $\alpha_{p(s_{k+1}-1)} = \delta_{s_{k+1}-1} - \delta_{s_{k+1}} \geq (\frac{\gamma}{3} - c_\delta)\delta'_k$. We will charge any increase in L this iteration to the term $\alpha_{p(s_{k+1}-1)} A_{p(s_{k+1}-1)}(T^*_{p(s_{k+1}-1)})$. Note that we are always in this case when we remove the last pipe because we can view the last pipe as intersecting a dummy pipe with $\delta = 0$ at D.

In order to bound $\alpha'_{p'(k)} A_{p'(k)}(T^*_{p'(k)})$ by $\alpha_{p(s_{k+1}-1)} A_{p(s_{k+1}-1)}(T^*_{p(s_{k+1}-1)})$ we must show that $p(s_{k+1}-1) \geq p'(k)$. Note $2^{p'(k)}$ is the cost at which the new, rotated pipe k surpasses the old pipe s_{k+1}. New pipe k intersects pipe $s_{k+1}-1$ before s_{k+1}, and $\delta'_k > \delta_{s_{k+1}-1}$, so pipes k and s_{k+1} meet before $s_{k+1}-1$ and s_{k+1} do. Therefore $g \leq 2^{p(s_{k+1}-1)}$, and when we reduce δ'_k to fix the breakpoint we never need to raise g beyond $2^{p(s_{k+1}-1)}$ before hitting a power of 2. Therefore

$$\begin{aligned}
\alpha_{p(s_{k+1}-1)} A_{p(s_{k+1}-1)}(T^*_{p(s_{k+1}-1)}) &\geq \left(\frac{\gamma}{3} - c_\delta\right) \delta'_k A_{p(s_{k+1}-1)}(T^*_{p(s_{k+1}-1)}) && \text{(by assumption)} \\
&\geq \left(\frac{\gamma}{3} - c_\delta\right) \alpha'_{p'(k)} A_{p(s_{k+1}-1)}(T^*_{p(s_{k+1}-1)}) && \text{(using } \delta'_k = \sum_{j \geq k} \alpha'_{p(j)}) \\
&\geq \left(\frac{\gamma}{3} - c_\delta\right) \alpha'_{p'(k)} A_{p'(k)}(T^*_{p'(k)})
\end{aligned}$$

We can charge the increase in $\alpha'_{p'(k)}$ to $\alpha_{p(s_{k+1}-1)}$ in the current chunk L_k, with a loss of $\left(\frac{\gamma}{3} - c_\delta\right)^{-1} = \frac{3}{\gamma - 3c_\delta}$, and this charge can only occur once for each L_k.

Case 2: $c_\delta \delta'_k \leq \delta_{s_{k+1}}$.

In this case there is no large collection of mass that we can easily guarantee is above $p'(k)$ in the current interval, but we do know there must be a lot of mass somewhere above $p(s_{k+1}-1)$ because $\delta_{s_{k+1}}$ is large. The α-mass $\alpha_{p(s_{k+1})} + \ldots + \alpha_{p(s_{k+2}-1)} = \delta_{s_{k+1}} - \delta_{s_{k+2}}$ is "used" in the next iteration and contributes L_{k+1} to L. We know $\gamma \delta_{s_{k+1}} = \gamma \delta'_{k+1} > \delta_{s_{k+2}}$, which implies $\sum_{i=s_{k+1}}^{s_{k+2}-1} \alpha_{p(i)} = \delta_{s_{k+1}} - \delta_{s_{k+2}} > (1-\gamma)\delta_{s_{k+1}}$. Now we can bound the increase

$$\begin{aligned}
\alpha'_{p'(k)} A_{p'(k)}(T^*_{p'(k)}) &\leq \left(\delta'_k - \delta_{s_{k+1}}\right) A_{p'(k)}(T^*_{p'(k)}) && (\alpha'_{p'(k)} = \delta'_k - \delta_{s_{k+1}}) \\
&\leq \left(\frac{1}{c_\delta} - 1\right) \delta_{s_{k+1}} A_{p'(k)}(T^*_{p'(k)}) && \text{(by assumption)} \\
&\leq \left(\frac{1}{c_\delta} - 1\right) \left(\frac{1}{1-\gamma} \sum_{i=s_{k+1}}^{s_{k+2}-1} \alpha_{p(i)}\right) A_{p'(k)}(T^*_{p'(k)}) && \text{(shown above)} \\
&\leq \frac{1 - c_\delta}{c_\delta(1-\gamma)} \sum_{i=s_{k+1}}^{s_{k+2}-1} \alpha_{p(i)} A_{p(i)}(T^*_{p(i)}) && (p'(k) < p(s_{k+1}) \leq p(i)) \\
&= \frac{1 - c_\delta}{c_\delta(1-\gamma)} L_{k+1}
\end{aligned}$$

Therefore we can charge the increase in L due to iteration k to the portion L_{k+1} used in the next iteration.

For a particular segment L_k of L, the $k-1$th iteration may been bounded by $\frac{1-c_\delta}{c_\delta(1-\gamma)}$ increase in L_k, and the kth iteration may charge against a $\frac{3}{\gamma - 3c_\delta}$ increase. Each type of charge can occur at most once per

chunk. Therefore the total increase in each piece, and hence the total increase in $L = \sum_k L_k$ is

$$\frac{1 - c_\delta}{c_\delta(1-\gamma)} + \frac{3}{\gamma - 3c_\delta}$$

This completes the proof. □

Lemma 4.4. *Given $\vec{\alpha}$ satisfying $\frac{\sigma_{K-1}}{\delta_{K-1}} \le D$ and $\delta_{k+1} < \gamma\delta_k$, we can find $\vec{\alpha}'$ such that such the corresponding f',L', δ', σ' satisfy $f(x) \le \frac{5}{2}f'(x)$, $L' = O(L)$, $\delta'_{k+1} < \gamma\delta'_k$, and $\sigma'_k < \gamma\sigma'_{k+1}$ for all k.*

Proof. The proof follows Lemma 4.3 but moves backwards through the pipes rather than forwards.

1. *Deletion Step:* Let k be the highest index such that $\sigma_{k-1} \ge \gamma\sigma_k$, and $l > 1$ the smallest integer such that $\sigma_{k-l} < \frac{2\gamma}{5}\sigma_k$. Such an l must exist because $\sigma_0 = 0$. Remove pipes $k-l+1, \ldots, k-1$, and replace them with the cheaper of pipes $k-l$ and k.

2. *Rotation Step:* As in Lemma 4.3, $f(x)$ may no longer be a linear combination of terms $A_i(x)$ because the new indifference point may not be a power of 2. We use a similar procedure as before to remedy this. Hold pipe k's cost for $2^{p(k)}$ flow fixed, and reduce σ_k while increasing δ_k to maintain the invariant until k and $k-l$ meet at a power of 2. Geometrically we are rotating $y = \sigma_k + \delta_k x$ counter-clockwise around $(2^{p(k)}, \sigma_k + \delta_k 2^{p(k)})$. Let σ'_k, δ'_k be the new parameters. Note that $\vec{\alpha}'$ and L' are now well-defined.

First, we analyze the change to σ_k and δ_k required by the rotation step and use this result to prove the constraints on both the σ's and δ's are satisfied at the end without changing $f(x)$ too much.

Claim: After rotation $\sigma'_k \ge \frac{2}{5}\sigma_k$, and $\delta'_k \le \frac{8}{5}\delta_k$.

Suppose the unmodified pipe k and $k-l$ meet at $g = \frac{\sigma_k - \sigma_{k-l}}{\delta_{k-l} - \delta_k}$. We will bound the adjustment required to guarantee they meet before $\frac{g}{2}$. Reduce σ_k to $\frac{2}{5}\sigma_k = \sigma'_k$. The modified pipe k has the same cost as the old at $2^{p(k)}$. If k is the final pipe then from Lemma 4.2 we know D = $2^{p(k)} \ge \frac{\sigma_k}{\delta_k}$. Otherwise, pipe k costs the same as $k+1$ at $2^{p(k)}$, so we have that $2^{p(k)} = \frac{\sigma_{k+1} - \sigma_k}{\delta_k - \delta_{k+1}} \ge \frac{\sigma_k}{\delta_k}$, using $\gamma\sigma_{k+1} > \sigma_k$ (the constraint fixed in the previous iteration). In either case $\delta_k 2^{p(k)} \ge \sigma_k$. Now,

$$\sigma_k + \delta_k 2^{p(k)} = \frac{2}{5}\sigma_k + \delta'_k 2^{p(k)}$$
$$\Rightarrow \delta'_k 2^{p(k)} = \frac{3}{5}\sigma_k + \delta_k 2^{p(k)} \le \left(1 + \frac{3}{5}\right)\delta_k 2^{p(k)} \Rightarrow \delta'_k \le \frac{8}{5}\delta_k$$

The constraints on the δs were satisfied before removing pipe $k-1$, so $\delta_{k-l} > \frac{1}{\gamma^2}\delta_k$. This implies

$$\frac{\delta_k}{\delta_{k-l} - \delta_k} \le \frac{\delta_k}{\frac{1}{\gamma^2}\delta_k - \delta_k} \le \frac{1}{4-1} = \frac{1}{3}$$

using $\gamma < \frac{1}{2}$. We combine this with the bound on δ'_k to bound the change in $\delta_{k-l} - \delta'_k$:

$$\begin{aligned}\delta_{k-l} - \delta'_k \ge (\delta_{k-l} - \delta_k) - \frac{3}{5}\delta_k &= (\delta_{k-l} - \delta_k)\left(1 - \frac{3}{5}\frac{\delta_k}{\delta_{k-l} - \delta_k}\right) \\ &\ge (\delta_{k-l} - \delta_k)\left(1 - \frac{3}{5}\cdot\frac{1}{3}\right) = \frac{4}{5}(\delta_{k-l} - \delta_k)\end{aligned}$$

Now we have enough information to bound the new switchover point.

$$\frac{\sigma'_k - \sigma_{k-l}}{\delta_{k-l} - \delta'_k} = \frac{\frac{2}{5}\sigma_k - \sigma_{k-l}}{\delta_{k-l} - \delta'_k} \leq \frac{\frac{2}{5}(\sigma_k - \sigma_{k-l})}{\delta_{k-l} - \delta'_k} \leq \frac{\frac{2}{5}(\sigma_k - \sigma_{k-l})}{\frac{4}{5}(\delta_{k-l} - \delta_k)} = \frac{1}{2}\frac{\sigma_k - \sigma_{k-l}}{\delta_{k-l} - \delta_k} = \frac{1}{2}g$$

There must be a power of 2 between $\frac{g}{2}$ and g, so we need to reduce σ_k by at most a factor of $\frac{2}{5}$. Finally, note that pipes 0 and 1 meet no sooner than 1, and $k > 1$ since it is always true that $\gamma\sigma_1 > \sigma_0 = 0$. Therefore $g > 1$, and hence the new changeover point is at least 1, so we do not need to worry about a term A_{-1}.

Claim: When the procedure finishes $\delta'_{k+1} < \gamma\delta'_k$ and $\sigma'_k < \gamma\sigma'_{k+1}$ for all k.

We chose l such that $\sigma_{k-l} < \frac{2\gamma}{5}\sigma_k$, so $\sigma_{k-l} < \gamma\sigma'_k$. Before starting, we had $\gamma^2\delta_{k-l} > \gamma\delta_{k-1} > \delta_k$, and $\gamma < \frac{1}{2}$, which implies $\delta'_k \leq \frac{8}{5}\delta_k < \frac{8}{5}\gamma^2\delta_{k-l} < \gamma\delta_{k-l}$. Note that the rotation step does not break any previously-satisfied constraints on larger k's.

Claim: For all x, $f(x) \leq \frac{5}{2}f'(x)$.

Only 1 round affects the interval $(2^{p(k-l-1)}, 2^{p(k)})$. Removing pipes only increases $f(x)$, and if we adjust σ_k, then it decreases by a factor of at most $\frac{2}{5}$, while δ_k increases, so $f'(x) \geq \frac{2}{5}f(x)$.

Now we analyze the increase in L. First, unlike in Lemma 4.3, the rotation step works against us, and we need to bound the increase.

Claim: Rotation only increases L by an $O(1)$-factor.

When adjusting pipe k, we increase δ_k without changing δ_{k+1}, which increases $\alpha_{p(k)}$. We have that $\alpha_{p(k)} \geq (1-\gamma)\delta_k$, and $\delta'_k \leq \frac{8}{5}\delta_k$, so

$$\alpha'_{p(k)} = \delta'_k - \delta_{k+1} \leq (\delta_k - \delta_{k+1})\left(1 + \frac{3}{5}\frac{\delta_k}{\delta_k - \delta_{k+1}}\right) \leq \alpha_{p(k)}\left(1 + \frac{3}{5}\frac{\delta_k}{\delta_k(1-\gamma)}\right) = \frac{8-5\gamma}{5(1-\gamma)}\alpha_{p(k)}$$

causing L to increase by at most $\frac{8-5\gamma}{5(1-\gamma)}$.

Second, we need to bound the increase in L caused by removing pipes. Let K' be the number of iterations and final pipes and $\alpha'_{p'(0)}, \ldots, \alpha'_{p'(K'-1)}$ the resulting non-zero α's. Iteration k, for $1 \leq k \leq K'$, deletes pipes $s_{k+1}+1, \ldots, s_k-1$ which removes $\alpha_{p(s_{k+1})}, \ldots, \alpha_{p(s_k-1)}$. Let $L_k = \sum_{i=s_{k+1}}^{s_k-1} \alpha_{p(i)}A_{p(i)}(T^*_{p(i)})$ be the amount these contribute to L. Since it moves backwards through pipes the indices of new pipes are not fixed yet, but as labeled at the end, round k ensures $\sigma'_j < \gamma\sigma'_{j+1}$ and creates a term $\alpha'_{p'(j)}$ where $j = K' - k$.

The rotation step reduces both $\alpha'_{p'(j)}$ and $p'(j)$ which can only help in this step, and we have already bounded the increase in $\alpha'_{p'(j+1)}$ due to rotation, so we assume that no rotation is needed. This implies $\alpha'_{p'(j)} = \delta_{s_{k+1}} - \delta_{s_k} = \sum_{i=s_{k+1}}^{s_k-1} \alpha_{p(i)}$. As in Lemma 4.3 we need to ensure that too much α-mass does not move too high.

Let $c_\sigma < \frac{2\gamma}{5}$ be a small constant. We need to consider two cases again: either $\sigma_{s_{k+1}} < c_\sigma\sigma'_{j+1}$ or $\sigma_{s_{k+1}} \geq c_\sigma\sigma'_{j+1}$.

Case 1: $\sigma_{s_{k+1}} < c_\sigma\sigma'_{j+1}$.

Intuitively, this means $\sigma_{s_{k+1}+1}$ is much larger than $\sigma_{s_{k+1}}$ because $\sigma_{s_{k+1}+1} \geq \frac{2\gamma}{5}\sigma'_{j+1}$, so by the time pipe s_{k+1} catches up with pipe $s_{k+1}+1$ or any later pipe, it has already covered an $O(1)$-fraction

of the distance to $2^{p'(j)}$. Therefore, pushing mass from up to $A_{p'(j)}(T^*_{p'(j)})$ increases L by only a constant factor.

We bound $2^{p'(j)}$ by bounding the cost to which pipe s_{k+1} must grow before switching pipes. Before removal the old pipe $s_k - 1$ crossed the new $j+1$ at $2^{p(s_k-1)} = \frac{\sigma'_{j+1} - \sigma_{s_k-1}}{\delta_{s_k-1} - \delta'_{j+1}} \leq \frac{\sigma'_{j+1}}{\frac{1}{\gamma}\delta'_{j+1} - \delta'_{j+1}} \leq \frac{\sigma'_{j+1}}{\delta'_{j+1}}$, so $\sigma'_{j+1} + \delta'_{j+1} 2^{p(s_k-1)} \leq 2\sigma'_{j+1}$. Pipe s_{k+1}'s cost increases faster than $s_k - 1$'s and surpasses s_k's cost before $2^{p(s_k-1)}$. Therefore $\sigma'_{j+1} + \delta'_{j+1} g \leq 2\sigma'_{j+1}$.

We know $\sigma_{s_{k+1}+1} \geq \frac{2\gamma}{5}\sigma'_{j+1}$ or else it would not have been removed. When s_{k+1} intersects $s_{k+1}+1$ at $2^{p(s_{k+1})}$ it has grown from $\sigma_{s_{k+1}}$ to at least $\sigma_{s_{k+1}+1}$ and therefore has covered at least

$$\frac{\sigma_{s_{k+1}+1} - \sigma_{s_{k+1}}}{2\sigma'_{j+1}} \geq \frac{\frac{2\gamma}{5}\sigma'_{j+1} - c_\sigma \sigma'_{j+1}}{2\sigma'_{j+1}} = \frac{2\gamma - 5c_\sigma}{10}$$

fraction of the distance to the indifference point between $s_{k+1}+1$ and s_k. Therefore

$$2^{p(s_{k+1})} \geq \frac{2\gamma - 5c_\sigma}{10} 2^{p'(j)} \Rightarrow A_{p'(j)}(T^*_{p'(j)}) \leq \frac{10}{2\gamma - 5c_\sigma} A_{p(s_{k+1})}(T^*_{p(s_{k+1})})$$

Every other affected $\alpha_{p(i)}$ is pushed up less than $\alpha_{p(s_{k+1})}$, so

$$\begin{aligned}
\alpha'_j A_{p'(j)}(T^*_{p'(j)}) &= \sum_{i=s_{k+1}}^{s_k-1} \alpha_{p(i)} A_{p'(j)}(T^*_{p'(j)}) \\
&\leq \sum_{i=s_{k+1}}^{s_k-1} \alpha_{p(i)} \left(\frac{10}{2\gamma - 5c_\sigma} A_{p(i)}(T^*_{p(i)}) \right) = \frac{10}{2\gamma - 5c_\sigma} L_k
\end{aligned}$$

Case 2: $\sigma_{s_{k+1}} \geq c_\sigma \sigma'_{j+1}$.

In this case pipes s_{k+1} and $s_{k+1}+1$ may meet very early, and $A_{p'(j)}(T^*_{p'(j)})$ could be much bigger than $A_{p(s_{k+1})}(T^*_{p(s_{k+1})})$. Note that we are never in this case when $\sigma_{s_{k+1}} = 0$. We have that

$$\begin{aligned}
\sigma'_{j+1} + \delta_{j+1} 2^{p'(j)} &= \sigma_{s_{k+1}} + \delta_{s_{k+1}} 2^{p'(j)} \\
\Rightarrow \alpha'_{p'(j)} = \delta_{s_{k+1}} - \delta'_{j+1} &= \frac{\sigma'_{j+1} - \sigma_{s_{k+1}}}{2^{p'(j)}} \leq \left(\frac{1}{c_\sigma} - 1 \right) \frac{\sigma_{s_{k+1}}}{2^{p'(j)}}
\end{aligned}$$

After the next round—which we know occurs because $\sigma_{s_{k+1}} \neq 0$— $\sigma_{s_{k+2}}$ will be the pipe preceding $\sigma_{s_{k+1}}$ (which is σ'_j). Using $\sigma_{s_{k+2}} < \gamma \sigma_{s_{k+1}}$, it is easy to see that $\sigma_{s_{k+1}} < \frac{\sigma_{s_{k+1}} - \sigma_{s_{k+2}}}{1-\gamma}$ and from the formula for $\sigma_{s_{k+2}}$ we have $\sigma_{s_{k+1}} - \sigma_{s_{k+2}} = \sum_{i=s_{k+2}}^{s_{k+1}-1} \alpha_{p(i)} 2^{p(i)}$

Combining the previous inequalities,

$$\begin{aligned}
\alpha'_{p'(j)} &\leq \left(\frac{1 - c_\sigma}{c_\sigma} \right) \frac{\sigma_{s_{k+1}}}{2^{p'(j)}} \leq \left(\frac{1 - c_\sigma}{c_\sigma} \right) \left(\frac{\sigma_{s_{k+1}} - \sigma_{s_{k+2}}}{1 - \gamma} \right) \frac{1}{2^{p'(j)}} \\
&\leq \frac{1 - c_\sigma}{c_\sigma (1 - \gamma)} \sum_{i=s_{k+2}}^{s_{k+1}-1} \alpha_{p(i)} 2^{p(i) - p'(j)}
\end{aligned}$$

Now we can apply Lemma 4.1 to finish the bound:

$$\begin{aligned}\alpha'_{p'(j)}A_{p'(j)}(T^*_{p'(j)}) \leq & \frac{1-c_\sigma}{c_\sigma(1-\gamma)}\sum_{i=s_{k+2}}^{s_{k+1}-1}\alpha_{p(i)}2^{p(i)-p'(j)}A_{p'(j)}(T^*_{p'(j)}) \\ \leq & \frac{1-c_\sigma}{c_\sigma(1-\gamma)}\sum_{i=s_{k+2}}^{s_{k+1}-1}\alpha_{p(i)}A_{p(i)}(T^*_{p(i)}) = \frac{1-c_\sigma}{c_\sigma(1-\gamma)}L_{k+1}\end{aligned}$$

Therefore we can charge the increase in L_k this iteration to L_{k+1} used in the next iteration.

For a particular chunk L_k of L, round k's increase may be bounded by a $\frac{10}{2\gamma-5c_\sigma}$-factor increase and round $k-1$ may be bounded by a $\frac{1-c_\sigma}{c_\sigma(1-\gamma)}$-factor increase. Each charge only occurs once. The rotation step adds another factor of $\frac{8-5\gamma}{5(1-\gamma)}$ on top of this. Therefore, the total growth of L is at most

$$\frac{8-5\gamma}{5(1-\gamma)}\left(\frac{1-c_\sigma}{c_\sigma(1-\gamma)}+\frac{10}{2\gamma-5c_\sigma}\right)$$

□