A Proof of Theorem 1

We will first give a high-probability bound on the service cost paid during the phase corresponding to facility cost f.

Lemma 1. *Suppose that we have a facility cost of f along with some initial facilities, and we continue running online facility location on some set of weighted points until we open x additional facilities. Let C be the total service cost paid. Then $E[C] \leq xf$ and*

$$Pr[C > xf(1+\epsilon)] \leq (\frac{1+\epsilon}{e^{\epsilon}})^x$$

Proof. The statement about expectation is immediate from expected waiting time. Let p_y be the maximum possible probability that we pay at least yf before opening a facility. If δf is the cost of service for the first point to arrive for $\delta \leq 1$, then we have $p_y = \max_\delta (1-\delta)(p_{y-\delta})$ Since $1 - \delta \leq e^{-\delta}$, we can show by induction that $p_y = e^{-y}$.

Now consider the probability of exceeding cost yf for opening x facilities. We will prove by induction that this is at most $e^{x \ln \gamma - y(1-1/\gamma)}$, for any $\gamma > 1$. The base case $x = 1$ is immediate. There are two ways to exceed yf cost when opening x facilities. Either we pay more than yf already for the first facility, which happens with probability e^{-y}, or we pay some $zf \leq yf$ prior to the first facility, then pay at least $(y-z)f$ for the remaining facilities. The probability of paying zf for the first facility is bounded by $e^{-z}dz$, so we can write:

$$\begin{aligned}
Pr[C > yf] &\leq e^{-y} + \int_0^y e^{-z} e^{(x-1)\ln\gamma + (z-y)(1-1/\gamma)} dz \\
Pr[C > yf] &\leq e^{-y} + e^{(x-1)\ln\gamma - y(1-1/\gamma)} \int_0^y e^{-z/\gamma} dz \\
Pr[C > yf] &\leq e^{-y} + \gamma e^{(x-1)\ln\gamma - y(1-1/\gamma)} (1 - e^{-y/\gamma}) \\
Pr[C > yf] &\leq e^{x\ln\gamma - y(1-1/\gamma)} + e^{-y} - e^{x\ln\gamma - y}
\end{aligned}$$

The last two terms sum to at most zero, which completes the induction. We now wish to bound the probability of exceeding $1+\epsilon$ times expectation. We set $y = x(1+\epsilon)$ and $\gamma = 1+\epsilon$ to get the expression claimed in the lemma. □

We now bound the probability of exceeding the expectation for the overall algorithm of many phases.

Lemma 2. *Suppose we have some initial facility cost f_0 and initial facilities, and we run online facility location until we open x additional facilities. We then increase the facility cost to βf_0 and continue until we open x more facilities. This process continues until we open x facilities with facility cost some f each. Let C be the total service cost induced by this process. Then $E[C] \leq xf\frac{\beta}{\beta-1}$ and $Pr[C > xf(1+\epsilon)\frac{\beta}{\beta-1}]$ decreases exponentially with x.*

Proof. The expectation follows simply from applying linearity of expectation to the result of lemma 1. Consider the time period when our facility cost was $\frac{f}{\beta^i}$ for some i. Did we pay more than $\frac{xf}{\beta^i} + \epsilon\frac{xf(\beta-\beta^{1/2})}{(\beta^{i/2})(\beta-1)}$? If no time period involved paying more than this amount, we can sum the cost over time periods to obtain a bound of

$$C \leq \frac{xf\beta}{\beta-1} + \epsilon xf\frac{\beta-\beta^{1/2}}{\beta-1}\frac{\beta^{1/2}}{\beta^{1/2}-1} \leq \frac{xf\beta}{\beta-1}(1+\epsilon)$$

For the time period when the facility cost is $\frac{f}{\beta^i}$, exceeding this bound would mean that we exceeded the expected cost times $1 + \epsilon\frac{(\beta-\beta^{1/2})(\beta^{i/2})}{\beta-1}$. Now we can bound the probability that there exists *any* time period when we exceeded the bound by applying lemma 1 along with the union bound. □

The result of lemma 2 implies that the probability of substantially exceeding the expected cost will be exponentially small in the number of means. Since we will be selecting a number of means $x = \Theta(k \log n)$, this means the probability of "error" will be polynomial in $1/n$, for a "high probability" of success.

B Proof of Theorem 2

Here we will show that the algorithm is very likely to halt with $f \leq \frac{O(C^*)\beta}{\kappa}$. Combining this result with theorem 1 will give bounds on our approximation ratio (with high probability). We will first show that when the algorithm raises the facility cost to f, there is very likely to be a solution with bounded service cost.

Lemma 3. *Consider the set of weighted means and additional points from the stream to be read when the algorithm first increases the service cost to f. With high probability, there is a k-means solution on these points of cost at most $C^* + \frac{f\kappa}{\beta-1}$.*

Proof. Let $\hat{C}_i$ be the service cost of assigning all points read from the stream when the facility cost was at most β^i to their facilities at that time. Let K_i be the set of facilities of cost β^i. Let C_i be the service cost of the assignments made during the time when facilities cost β^i. We observe that $C_0 = \hat{C}_0$ trivially. For later phases, we observe that the cost of assigning a set of points to some mean y is at most the cost of assigning the same set of points to their center of mass, plus the distance from that center of mass to y (weighted by the number of points). For points read prior to phase i, the cost to assign them to their center of mass is at most $\hat{C}_{i-1}$. The term C_i accounts for the (weighted) cost of assigning these centers of mass to their new facilities, plus the cost of assigning newly read points to their facilities. We conclude that $\hat{C}_i \leq \hat{C}_{i-1} + C_i$ and thus by induction $\hat{C}_i \leq \Sigma_{j \leq i} C_j$. We now apply lemma 2 to see that the total cost of assigning points read while the facility cost is at most f/β to their facilities is at most $\frac{f\kappa}{\beta-1}$. Reassigning these points to the centers of mass of their clusters only reduces cost.

We now consider the cost of assigning these centers of mass, plus any unread points, to the optimum k-means. This cost should not exceed the total cost of mapping the original points to their centers of mass, plus the cost of clustering unread points. The former is at most $\frac{fk \log n}{\beta-1}$ from before, and the latter is at most C^* for obvious reasons. Combining these gives the bound claimed in the lemma. □

We now want to show that the algorithm actually halts at a reasonable time. Consider the algorithm running with facility cost f. We look at each optimum cluster and divide it into regions. The first region has the $1 - (1/\Delta)$ points of the optimum cluster closest to the facility, the next region has the next $(1/\Delta)(1 - 1/\Delta)$ points, and so forth. The total number of regions is thus at most $k \log_\Delta n$. We'd like to bound the number of facilities constructed here. The algorithm can construct at most $k \log_\Delta n$ "first" facilities for each region. Let $s^*_{x,i}$ be the total optimum service cost for the i'th closest region to optimum facility x. Once a facility has been opened in the i'th closest region to facility x, the remaining points from that region can be dealt with for service cost at most twice the sum of their optimum service cost and the maximum service cost in that region (by 2-approximate triangle inequality on k-means distance-squared cost). It follows that the total service cost paid by our algorithm for these points is $s_{x,i} \leq 2\Delta s^*_{x,i+1} + 2s^*_{x,i}$. The expected number of means generated from these points is just the total service cost divided by f. We conclude that the total expected number of means is at most:

$$k \log_\Delta n + 2(\Delta + 1)(\frac{C^*}{f} + \frac{\kappa}{\beta - 1})$$

Setting $f > zC^*/\kappa$ gives us an expected number of means which is at most:

$$\kappa(\frac{k \log n}{\kappa \log \Delta} + \frac{2\Delta + 2}{z} + \frac{2\Delta + 2}{\beta - 1})$$

We observe that as β and κ grow large, it becomes sufficient to have $z > 4$. For smaller values (i.e. $\kappa \approx k \log n, \beta > 2$) we will need a larger (but still constant) value of z. In any case, we should finish once $f > O(C^*/\kappa)$ as required. Note that if the expected number of means is a constant factor smaller than κ, it will follow from applying Chernoff bounds that the probability of exceeding the expectation is at most $1 - \frac{1}{poly(n)}$.

C Integration of Approximate Nearest-Neighbor

C.1 Algorithm for stronger result

We can use the result of [6], with the set of facilities are maintained through the data structure mentioned in [29]. We first select ζ hash functions, where $\zeta = \frac{\log \kappa}{\log(1/g)}$ (where g is a probability bound on collisions, as specified in [29]). We use the locality sensitive hash family that were developed based on the Leech Lattice hash functions mentioned in [6].

When a point x is added to K, we must compute $H(x) = (h_1(x), h_2(x), \ldots h_\zeta(x))$ and store x at $H(x)$ in a table (we store only non-empty locations). Since we will be performing many nearest-neighbor computations (and errors will be independent), we will not need to enforce a high-probability guarantee on each computation separately and can use a single table rather than the polylog(κ) such tables implicit in [6].

When a point x is first seen, we must determine some nearby point (or determine that none are within f of it). If we knew δ_x^* (the optimal distance from x to its nearest neighbor), we would pick $\tilde{O}(\kappa^\varrho)$ points v from $B(x, \delta_x^*)$ and search buckets $H(v)$, where $\varrho = \frac{\xi}{\log(1/g)}$, where ξ is the entropy of the hash values based on the query point. This takes $\tilde{O}(\kappa^\varrho)$ time. Alternately, we can use $\tilde{O}(\kappa^{\frac{1}{1-\varrho}})$ space for query time $\tilde{O}(d)$.

However, because we don't know δ_x^*, we must guess it. To solve this, we add the constraint that if $\delta_x < \frac{f}{n}$, we will always service the point (this can add only one f in total to the service cost). We then know we only need to find $\delta \in [f/n, f]$. This can be accomplished with a binary search in $\log n$ time; this replaces the $k \log n$ in the runtime with $(\log n)(\log k + \log \log n)$. For $k > \log \log n$, this does improve the running-time result.

C.2 Proof of Theorem 4

Suppose that at some point in the algorithm we have $f > zC^*/\kappa$. Then our expected number of means (as in the proof of theorem 2) is at most:

$$\kappa\left(\frac{k \log n}{\kappa \log \Delta} + \frac{(2\Delta + 2)\nu^2}{z} + \frac{(2\Delta + 2)\nu^2}{\beta - 1}\right)$$

With appropriately large constant settings of z, κ, and β, we can conclude that the expected number of means is a constant times smaller than κ. However, we can no longer apply Chernoff bounds because the errors in distance measurements are not independent. Instead we conclude that there is a constant probability to halt at this phase. Now consider a phase where $f > \beta^i zC^*/\kappa$; applying Markov's inequality, our probability of halting at this phase (if we have not halted prior to the phase) will be at least $1 - \frac{1}{2\beta^i}$. Combining these terms, we conclude that we should halt at the phase where $f > \beta^i zC^*/\kappa$ with probability $(1 - \frac{1}{2\beta^i})(\frac{1}{(2\beta)^{i-1}})$. In this case our expected cost is $\beta^i zC^* \frac{\beta}{\beta - 1}$, so we can combine terms and solve the geometric sum to bound our cost by $O(\beta)$. This is a potentially large constant, but our performance in practice should be better since we assumed worst-case correlations in the distance measurement errors.

D Further Modifications

D.1 Parallelism

We considered re-introducing parallel invocations of online facility location. Unlike [12], who did this for probability guarantees, we do this to minimize the chance of overshooting the best facility cost. Furthermore, we allow each to run independently, and with a different starting facility cost. We run these until the stream is exhausted, and keep the sketch belonging to the instance with smaller final facility cost. This should help us avoid the situation that hurt our approximation guarantee in Theorem 3. We tested this by having two invocations, one with the normal starting facility cost, and the other with $\sqrt{\beta}$ times that amount. The differences in cost due to this approach were minimal, suggesting that it doesn't help significantly in practice. The results of some experiments of this type are shown in Figures 13 and 14 (next section).

D.2 Value of β

To some extent, the value of the β parameter in our algorithm is the last remnant of [10]'s encoding of worst-case behavior. We ran some cases again with $\beta = 4$ (instead of their $\beta \approx 77.3$); this caused a larger amount of time to be used, although for small memory availability, it was also more accurate. The time increase is hardly surprising, as a smaller β leads to a smaller facility cost in each phase, and thus more facilities and more phase transitions. The improved accuracy implies that [10]'s value of β causes an overshoot of the best facility cost in small memory cases. The results of some experiments of this type are shown in Figures 13 and 14 (next section).

E Additional Experimental Results

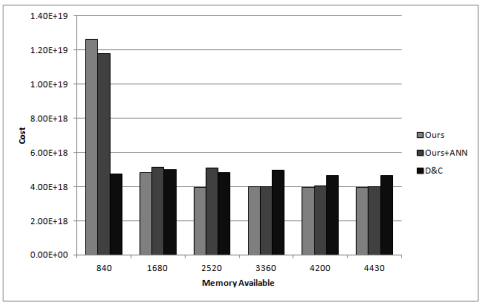

Figure 9: Census Data, k=8, cost

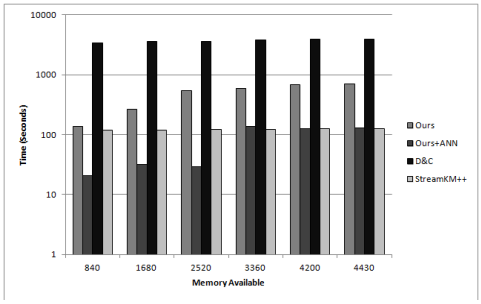

Figure 10: Census Data, k=8, time

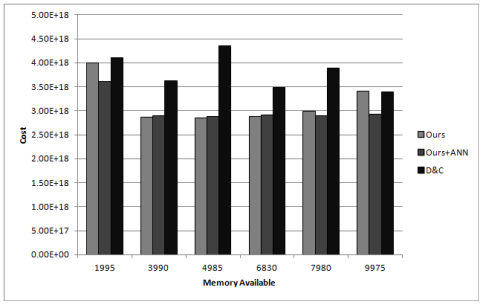

Figure 11: Census Data, k=19, cost

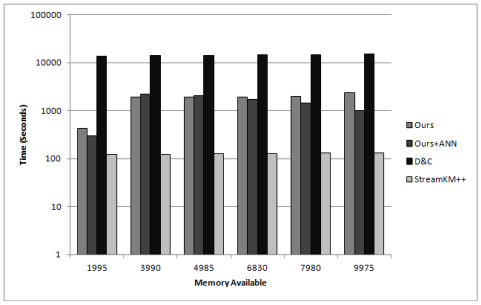

Figure 12: Census Data, k=19, time

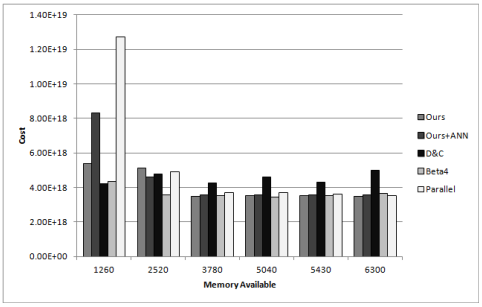

Figure 13: Census Data, k=12, cost; additional trials

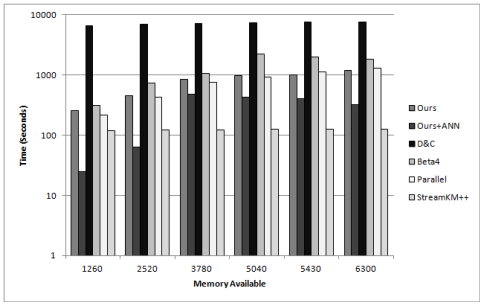

Figure 14: Census Data, k=12, time; additional trials