On r-gatherings on the Line

Yijie Han[1] **and Shin-ichi Nakano**[2]
[1]School of Computing and Engineering, University of Missouri at Kansas City, Kansas City, MO 64110, USA
[2]Department of Computer Science, Gunma University, City, Kiryu 376-8515, Japan
hanyij@umkc.edu, nakano@cs.gunma-u.ac.jp

Abstract— *Given an integer r, a set C of customers, a set F of facilities, and a connecting cost $co(c, f)$ for each pair of $c \in C$ and $f \in F$, an r-gathering of customers C to facilities F is an assignment A of C to open facilities $F^{'} \subset F$ such that r or more customers are assigned to each open facility. We wish to find an r-gathering with the minimum cost, where the cost is $\max_{c_i \in C}\{co(c_i, A(c_i))\}$. When all C and F are on a line an algorithm to find such an r-gathering is known. In this paper we give a faster algorithm with time complexity $O(|C| + |F| \log^2 r + |F| \log |F|)$.*

Keywords: algorithm, facility location, gathering

1. Introduction

The facility location problem and many of its variants are studied[7], [8]. In the basic facility location problem we are given (1) a set C of customers, (2) a set F of facilities, (3) an opening cost $op(f)$ for each $f \in F$, and (4) a connecting cost $co(c, f)$ for each pair of $c \in C$ and $f \in F$, then we open a subset $F' \subset F$ of facilities and find an assignment A of C to F' so that a designated cost is minimized.

An *r-gathering*[6] of customers C to facilities F is an assignment A of C to *open facilities* $F^{'} \subset F$ such that r or more customers are assigned to each open facility. (This means each open facility has enough number of customers.) We assume $|C| >> r$ holds. Then we define the cost of (the max version of) a gathering as $\max_{c_i \in C}\{co(c_i, A(c_i))\}$. We assume $op(f_j) = 0$ for each $f_j \in F$ in this paper, as in [4]. The min-max version of the *r-gathering problem* finds an r-gathering having the minimum cost. For the min-sum version see the brief survey in [6].

Assume that F is a set of locations for emergency shelters, and $co(c, f)$ is the time needed for a person $c \in C$ to reach a shelter $f \in F$. Then an r-gathering corresponds to an evacuation assignment such that each opened shelter serves r or more people, and the r-gathering problem finds an evacuation plan minimizing the evacuation time span.

Armon[6] gave a 3-approximation algorithm for the r-gathering problem and proves that with assumption $P \neq NP$ the problem cannot be approximated within a factor less than 3 for any $r \geq 3$. Akagi and Nakano[4] gave an $O((|C| + |F|) \log(|C| + |F|))$ time algorithm to solve the r-gathering problem when all C and F are on a line. In this paper we give a faster $O(|C|+|F| \log^2 r+|F| \log |F|)$ time algorithm. Since we can assume in general $|F| << |C|$ and $r << |C|$ our algorithm is faster than the one in[4].

The remainder of this paper is organized as follows. Section 2 gives an algorithm to solve a decision version of the r-gathering problem, which is used as a subroutine in our main algorithm in Section 4. In Section 3 we describe the computation of left and right boundaries. Section 4 contains our main algorithm for the r-gathering problem. Section 5 analyze the running time of the algorithm tightly. Finally Section 6 is a conclusion.

2. (k,r)-gathering on the line

In this section we give an algorithm to solve a decision version of the r-gathering problem.

Given customers $C = \{c_0, c_1, \cdots, c_{|C|-1}\}$ and facilities $F = \{f_0, f_1, \cdots, f_{|F|-1}\}$ on a line (we assume they are distinct points and appear in those order from left to right) and two numbers k and r, (k, r)-gathering is an r-gathering such that $\max_{c_i \in C}\{co(c_i, A(c_i))\} \leq k$. Because there are $|C||F|$ possible $co(c_i, A(c_i))$ values we can do $\log(|C||F|)$ binary searches using (k, r)-gathering algorithms to find the $\min_A \max_{c_i \in C}\{co(c_i, A(c_i))\}$ (the min-max value). In [4] Akagi and Nakano observed that the number of binary searches can be reduced to $O(\log(|C| + |F|))$.

For a facility f, the index of its left boundary is $l(f) = \min\{i||f - c_i| \leq k\}$ and its left boundary is $c_{l(f)}$ and the index of its right boundary is $r(f) = \max\{i||f - c_i| \leq k\}$ and its right boundary is $c_{r(f)}$. Two facilities $f_a < f_b$ are intersecting if $r(f_a) \geq l(f_b) - 1$.

To find out whether a (k, r)-gathering exits we first compute the (indices of) left and right boundaries for every facility. The algorithm for computing these will be explained in the next section. In this section we just assume we can have them. For a facility f, if $r(f) - l(f) + 1 < r$ then we close it.

We can assume that the customers assigned to a facility is consecutive. A consecutive r' customers going to a facility are called a complete interval if $r' \geq r$. If $r' < r$ then they are called an incomplete interval.

We will use the Left-to-Right Maximal Scan and the Right-to-Left Minimal Scan. The Left-to-Right Maximal Scan is shown below:

Left-to-Right Maximal Scan

1. Find the rightmost non-closing facility f_a with $|c_0 - f_a| \leq k$. Set $i = a$. Set $border = 0$.

2. Find the rightmost non-closing intersecting facility f_b to the right of f_i.
if f_b does not exist then there is no solution; exit;
if $l(f_b) > border + r - 1$ then
begin
Mark $c_{border}, c_{border+1}, ..., c_{l(f_b)-1}$ as a complete interval of customers going to f_i;
Set $border = l(f_b)$;
end
else
begin
if $r(f_b) \geq border + 2r - 1$ then
begin
Mark $c_{border}, c_{border+1}, ..., c_{border+r-1}$ as a complete interval of customers going to f_i;
Set $border = border + r$;
end
else goto Step 3;
end
if $r(f_b) = |C| - 1$ then
begin
Mark $c_{border}, c_{border+1}, ..., c_{|C|-1}$ as a complete interval of customers going to f_b;
(k, r)-gathering found; exit;
end
else
begin
$i = b$; goto Step 2;
end
3 /*Here we reached a breakpoint because f_i and f_b cannot have $2r$ customers going to them.*/
if $r(f_b) = |C| - 1$ then
begin
Mark $c_{border}, c_{border+1}, ..., c_{border+r-1}$ as a complete interval of customers going to f_i and mark $c_{border+r}, c_{border+r+1}, ..., c_{|C|-1}$ as an incomplete interval of customers going to f_b;
exit;
end
else
begin
Let f_c be the immediate next non-closing facility right to f_b;
Mark $c_{border}, c_{border+1}, ..., c_{border+r-1}$ as a complete interval of customers going to f_i;
If f_b is not intersecting f_c then there is no solution for a (k, r)-gathering and we exit;
Mark $c_{border+r}, c_{border+r+1}, ..., c_{l(f_c)-1}$ as an incomplete interval of customers going to f_b;
Treat $c_{l(f_c)}$ through $c_{|C|-1}$ and $f_c, f_{c+1}, ..., f_{|F|-1}$ as a separate problem using divide-and-conquer;
/* Here we say that we break between $c_{l(f_c)-1}$ and $c_{l(f_c)}$ and between f_{c-1} and f_c.*/
exit;
end

Note that the Left-to-Right Maximal Scan for all facilities takes $O(|F|)$ time after the left and right boundaries are computed. If the Scan results in no breakpoints then we obtained a (k, r)-gathering. We will say that such a Scan is a successful Scan. If there is only one breakpoint then this breakpoint results in one incomplete interval and it is at the rightmost position among all formed intervals. In the case there is only one breakpoint we will say that the Scan is a complete Scan.

Now the Right-to-Left Minimal Scan:

Right-to-Left Minimal Scan

1. Find the rightmost non-closing facility f_a with $|f_a - c_{|C|-1}| \leq k$. Set $i = a$. Set $border = |C| - 1$.
2. Find the rightmost intersecting neighbor f_b to the left of f_i such that $border - l(f_b) + 1 \geq 2r$.
if f_b does not exist then goto Step 3.
if $r(f_b) \leq border - r$ then
begin
Mark $c_{r(f_b)+1}, c_{r(f_b)+2}, ..., c_{border}$ as a complete interval of customers going to f_i;
Set $border = r(f_b)$;
end
else
begin
Mark $c_{border-r+1}, c_{border-r+2}, ..., c_{border}$ as a complete interval of customers going to f_i;
Set $border = border - r$;
end
if $l(f_b) = 0$ then
begin
Mark $c_0, c_1, ..., c_{border}$ as a complete interval of customers going to f_b;
(k, r)-gathering found; exit;
end
else
begin
$i = b$; goto Step 2;
end
3 /*We reached a breakpoint because f_b cannot have r customers going to it.*/
Let f_b be the leftmost facility left to f_i and intersects with f_i.
if $l(f_b) = 0$ then
begin
Mark $c_{border-r+1}, c_{border-r+2}, ..., c_{border}$ as a complete interval of customers going to f_i and mark $c_0, c_1, ..., c_{border-r}$ as an incomplete interval of customers going to f_b;
exit;
end
else
begin

Let f_c be the immediate next non-closing facility left to f_b;

Mark $c_{border-r+1}, c_{border-r+2}, ..., c_{border}$ as a complete interval of customers going to f_i;

If $|c_{r(f_c)+1} - f_b| > k$ then there is no solution for a (k,r)-gathering and we exit;

Mark $c_{r(f_c)+1}, c_{r(f_c)+2}, ..., c_{border-r}$ as an incomplete interval of customers going to f_b;

Treat c_0 through $c_{r(f_c)}$ and $f_0, f_1, ..., f_c$ as a separate problem using divide-and-conquer;

/* We say that we break between $c_{r(f_c)}$ and $c_{r(f_c)+1}$ and between f_c and f_{c+1}.*/

exit;

end

Note that the Right-to-Left Minimal Scan for all facilities takes $O(|F|)$ time after the left and right boundaries are computed. If the Scan results in no breakpoints then we obtained a (k,r)-gathering. We will say that such a Scan is a successful Scan. If there is only one breakpoint then this breakpoint results in one incomplete interval and it is at the leftmost position among all formed intervals. In the case there is only one breakpoint we will say that the Scan is a complete Scan.

Let i be an interval of customers going to f. The extended interval of i is $\{c_{l(f)}, c_{l(f)+1}, ...c_{r(f)}\}$.

Lemma 1: If a complete Left-to-Right Maximal Scan S results in a set $S(S)$ of I intervals then at least I facilities has to open for a (k,r)-gathering.

Proof: Let A be any set of intervals and we will use $E(A)$ to denote the set of extended intervals in A. Assume that a (k,r)-gathering G has a set $S(G)$ of $I' < I$ complete intervals. Then there is an extended interval i_1 in $E(G)$ that proper contains an extended interval i_2 in $E(S)$ and the rightmost customer in i_1 is to the right of the rightmost customer in i_2. This can be seen by starting from the left side and going to the right, comparing extended intervals one in $E(S)$ against one in $E(G)$. This says that S is not a maximal scan as in the Left-to-Right Maximal Scan we always find the rightmost intersecting neighbor in Step 2. □

Lemma 2: If a complete Right-to-Left Minimal Scan S results in a set $S(S)$ of I intervals then at most $I-1$ facilities can open for a (k,r)-gathering.

Proof: Assume a (k,r)-gathering G has a set $S(G)$ of $I' > I-1$ complete intervals. Let $E(G)$ be the set of extended intervals of $S(G)$. Let $E(S)$ be the set of extended intervals of $S(S)$. Then there is an extended interval i_1 in $E(S)$ that proper contains an extended interval i_2 in $E(G)$ and the leftmost customer in i_1 is to the left of the leftmost customer in i_2. This can be seen by starting from the right side and going to the left, comparing extended intervals one in $E(S)$ against one in $E(G)$. This says that S is not a minimal scan as in the Right-to-Left Minimal Scan we always find the rightmost intersecting neighbor in Step 2. □

Lemmas 1 and 2 explains why the Left-to-Right Maximal Scan is called a maximal scan and why the Right-to-Left Minimal Scan is called a minimal scan.

Lemma 3: If a complete Left-to-Right Maximal Scan has I_{max} intervals and a complete Right-to-Left Minimal Scan has I_{min} intervals then $I_{min} \geq I_{max}$.

Proof: From Lemmas 1 and 2. □

Theorem 1: Assume we have a complete Left-to-Right Maximal Scan S_{max} with I_{max} intervals and a complete Right-to-Left Minimal Scan S_{min} with I_{min} intervals. If $I_{max} = I_{min}$ then there is no solution for a (k,r)-gathering. If $I_{max} < I_{min}$ then the two Scans can be combined into a solution for (k,r)-gathering.

Proof: If $I_{max} = I_{min}$ then Lemma 1 says that any (k,r)-gathering has $\geq I_{max}$ facilities open while Lemma 2 says that any (k,r)-gathering has $< I_{min}$ facilities open. Thus it is impossible to have a (k,r)-gathering.

If $I_{max} < I_{min}$ then there is a complete interval i_{min} created in S_{min} that is contained in a complete interval i_{max} created in S_{max}. Let $c_{min,l}$ be the leftmost customer in i_{min}, $c_{min,r}$ be the rightmost customer in i_{min}, $c_{max,l}$ be the leftmost customer in i_{max} and $c_{max,r}$ be the rightmost customer in i_{max}. Let i_{min} be the j_{min}-th interval counting from right to left created by S_{min} and i_{max} be the j_{max}-th interval counting from left to right created by S_{max}. We create a (k,r)-gathering by using the 0th through $(j_{max}-1)$-th intervals created by S_{max} and the 0th through $(j_{min}-1)$-th intervals created by S_{min}. We then add a complete interval for $c_{max,l}$ through $c_{min,r}$ and let them go to the facility opened in S_{max} for $c_{max,l}$ through $c_{max,r}$. This creates a (k,r)-gathering. We say that we combined S_{max} with S_{min} at i_{max} and i_{min}. □

Now we consider the situation where we have multiple breakpoints in the Scans. We use the following Fix procedure:

Fix

1. Start with the Right-to-Left Minimal Scan S_{min}.
2. if S_{min} is successful then we obtained a (k,r)-gathering and we exit;

else we stop when we reach the first breakpoint. This breakpoint partitions the customers into two sets $\{c_0, ..., c_{a-1}\}$ and $\{c_a, ..., c_{|C|-1}\}$ and partitions the facilities into two sets $\{f_0, ..., f_{b-1}\}$ and $\{f_b, ..., f_{|F|-1}\}$. $\{c_a, ..., c_{|C|-1}\}$ has been put into $I(S_{min})$ intervals (with one incomplete interval at the leftmost position and other $I(S_{min})-1$ complete intervals).

3. Now we start the Left-to-Right Maximal Scan S_{max} for $\{c_a, ..., c_{|C|-1}\}$.
4. /* If S_{max} is successful then S_{max} created $\leq I(S_{min})-1$ complete intervals by Lemma 2.*/
5. (Case 1) If S_{max} is successful or is complete with $\leq I(S_{min}) - 1$ intervals then we find the *leftmost* (complete) interval i_{max} created by S_{max} that contains a (complete) interval i_{min} created by S_{min} and combine

the intervals created by S_{max} and S_{min} at i_{max} and i_{min} to get a (minimal) solution for the (k, r)-gathering for $\{c_a, ..., c_{|C|-1}\}$. If there are more than one complete intervals created by I_{min} that are contained in i_{max} then we will pick the *leftmost* one to be combined with i_{max}. It is a minimal solution because we used "*leftmost*".
6. (Case 2) If S_{max} is complete with $I(S_{min})$ intervals then by Theorem 1 there is no solution for a (k, r)-gathering for $\{c_a, ..., c_{|C|-1}\}$. If there is a solution S for a (k, r)-gathering for $\{c_0, c_1, ..., c_{|C|-1}\}$ then let f be the facility c_a goes to in S. Let I be the number of open facilities to the right of and including f opened by S. Because c_a goes to f in S therefore f is to the right of f_c where $c_{r(f_c)} = c_{a-1}$ and f_c would be the first open facility if we would resume the Right-to-Left Minimal Scan after the first breakpoint. Thus the extended interval of f intersects with the extended interval of the last complete interval of S_{min}. However, S_{min} reached a breakpoint and thus $I \leq I(S_{min}) - 1$ by Lemma 2 (Note here that we may move the breakpoint to between $c_{l(f)-1}$ and $c_{l(f)}$). However, if we take the set F_o of open facilities right to and include f opened by S, then $|F_o| \leq I(S_{min}) - 1$. On the other hand S_{max} has $I(S_{min})$ intervals and thus the number of intervals in F_o has to be $\geq I(S_{min})$. This contradiction says that there is no solution for (k, r)-gathering for $\{c_0, c_1, ..., c_{|C|-1}\}$. Exit.
7. (Case 3) If S_{max} is not complete. Then stop at the first breakpoint of S_{max}. This breakpoint will partition $\{c_a, ..., c_{|C|-1}\}$ into $P = \{c_a, ...c_{a_1}\}$ and $\{c_{a_1+1}, ..., c_{|C|-1}\}$ and partition $\{f_b, ..., f_{|F|-1}\}$ into $\{f_b, ..., f_{b_1}\}$ and $\{f_{b_1+1}, ..., f_{|F|-1}\}$. Let c_{a_1} be a member of a complete interval i created by S_{min}. If c_{a_1} is not the rightmost customer in i then we add $c_{a_1+1}, c_{a_1+2}, ..., c_{a_2}$ to P, where c_{a_2} is the rightmost customer in i. The situation for customers $\{c_a, .., c_{a_2}\}$ can be analyzed in the same way as we analyzed in Steps 5 and 6.

Theorem 2: After the left and right boundaries have been computed we can find whether a solution for a (k, r)-gathering exists in $O(|F|)$ time.

Alternatively after computing the boundaries we can use the $O(|F|)$ time decision algorithm in [4].

3. Computing Left and Right Boundaries

For two neighboring facilities f_a and f_{a+1}, let $2r$ customers $F_{a,a+1} = \{c_b, c_{b+1}, ..., c_{b+2r-1}\}$ be such that $|f_a - c_{b+2r-2}| < |f_{a+1} - c_{b-1}|$ and $|f_a - c_{b+2r}| > |f_{a+1} - c_{b+1}|$. $F_{a,a+1}$ is called the boundary set of customers between f_a and f_{a+1}.

Lemma 4: Let $F_{a,a+1} = \{c_b, c_{b+1}, ..., c_{b+2r-1}\}$ be the boundary set of f_a and f_{a+1}, then in an optimal r-gathering or an (k, r)-gathering c_d, $d > b + 2r - 1$, will not go to facility f_a and c_e, $e < b$, will not go to facility f_{a+1}.

Proof: Suppose in an optimal r-gathering or a (k, r)-gathering c_{b+2r} goes to facility f_a. Let the leftmost customer going to f_a be c_t. If $t \geq b + 1$ then we can delete f_a and let all customers going to f_a now go to f_{a+1}. If $t \leq b$ then we can let $c_{b+r}, c_{b+r+1}, ..., c_l$ go to f_{a+1}, where c_l was the rightmost customer of f_a.

The other situation can be proved similarly. □

In order to use Lemma 4 we need place a dummy customer d_l at the left of f_0 and a dummy customer d_r at the right of $f_{|F|-1}$ and let $|f_0 - d_l|$ and $|f_{|F|-1} - d_r|$ larger than $\max\{|f_0 - c_{|C|-1}|, |f_{|F|-1} - c_0|\}$.

We will let $ll(f_{a+1}) = rl(f_a) = c_b$ and $lr(f_{a+1}) = rr(f_a) = c_{b+2r-1}$.

Lemma 4 says that for computing an optimal r-gathering or a (k, r)-gathering we need consider no more than $4r$ distances corresponding to customers in $[ll(f_a), lr(f_a)]$ and $[rl(f_a), rr(f_a)]$ for each facility. Thus the total number of distances to be considered is $4|F|r$. We may collect all these $4|F|r$ distances and then do binary search $\log(4|F|r)$ times to find the minimum k value for an optimal r-gathering. This will result in $O(|C|+|F|r\log(|F|r)+|F|(\log r)(\log(|F|r)))$ time for r-gathering by (1) preprocess them in $O(|C| + |F|)$ time to compute the boundary sets $F_{a,a+1}$, (2) sort $4|F|r$ distances in $O(|F|r\log(|F|r))$ time, (3) binary search $\log(4|F|r)$ rounds among the $4|F|r$ possible minimum distances where each round consists of computing the left and right boundaries in $O(|F|\log r)$ time and Right-to-Left Minimal Scan and Left-to-Right Maximal Scan in $O(F)$ time.

We maintain the set M of possible minimum costs, then repeatedly compute the median k of possible minimum costs, then compute left and right boundaries and call Right-to-Left Minimal Scan and Left-to-Right Maximal Scan for value k to find whether a (k, r)-gathering exits. If it returns YES then the minimum cost is less than or equal to k and we can remove the larger half of costs from M. If it returns NO then the minimum cost is larger than k and we can remove the smaller half of costs from M. After $\log(4|F|r)$ rounds we can find the minimum cost k^*. In later sections we show we can do better than this.

4. r-gathering on the line

If all C and F are on the line, an $O((|C|+|F|)\log(|C|+|F|))$ time algorithm to solve the r-gathering problem is known[4]. In this section we give a faster algorithm. Our algorithm runs in $O(|C|+|F|\log^3 r+|F|\log|F|\log r)$ time. Since $C >> F$ and $C >> r$ holds in general, or if we can assume r as a constant, our algorithm is faster.

We can observe that the minimum cost k^* of a solution of an r-gathering problem is $co(c, f)$ for some $c \in C$ and some $f \in F$. Since the number of possible minimum cost, say some $co(c, f)$, is at most $4|F|r$ by Lemma 4, one can find the minimum cost in $O(|C| + |F|r\log(|F|r) + |F|(\log r)(\log(|F|r))$ time as we explained before.

However we can design a faster algorithm which runs in $O(|C| + |F|\log^3 r + |F|\log|F|\log r)$ time. Our algorithm maintains a set M of possible minimum costs, then repeatedly computes the "median of medians" k, defined below, then call Right-to-Left Minimal Scan and Left-to-Right Maximal Scan for k. Depending whether a (k,r)-gathering exists the algorithm removes some subset of possible minimum costs from M. After $O(\log^2 r)$ rounds M has at most $2|F|$ distances remaining, then we can find the minimum cost k^* by an ordinary binary search. Now we explain the detail.

Set initially $M\ell(f_j) = \{co(c_i, f_j)|c_i \in [ll(f_j), lr(f_j)]\}$, and $Mr(f_j) = \{co(c_i, f_j)|c_i \in [rl(f_j), rr(f_j)]\}$. We are going to repeatedly remove the half of distances from some $M\ell(f_j)$ and/or $Mr(f_j)$. M is the set of all $M\ell(f_j)$ and $Mr(f_j)$, $j = 1, 2, \cdots, |F|$, however if $M\ell(f_j)$ has exactly one distance then $M\ell(f_j)$ is removed from M. Similar for $Mr(f_j)$.

We will use a weighing scheme similar to the one used in [5]. If $M\ell(f_j)$ has $2r/2^x$ customers we define the weight $w\ell(f_j)$ of $M\ell(f_j)$ as $(1+\log r - x)$. The weight $wr(f_j)$ of $Mr(f_j)$ is defined similarly. The weight of M is the sum of the weights of $M\ell(f_j)$ and $Mr(f_j)$ in M.

Initially each $M\ell(f_j)$ has exactly $2r$ customers, so $x = 0$, and its weight is $1 + \log r$. So initially the weight of M is $2|F|(1 + \log r)$.

Say there are $N \le 2|F|$ $M\ell(f_j)$'s and $Mr(f_j)$'s with more than one distance remaining and the total weights in them is T. In each round we find the median of $M\ell(f_j)$ and the median of $Mr(f_j)$ and this gives us N medians. This takes constant time for each facility. We then find the median k of these N medians and this takes $O(|F|)$ time. Say that k is the median of $M\ell(f_i)$ then we place all $M\ell(f_j)$'s and $Mr(f_j)$'s whose median is $< k$ above $M\ell(f_i)$ and all $M\ell(f_j)$'s and $Mr(f_j)$'s whose median is $> k$ below $M\ell(f_i)$. Because k is the median of the medians we have put half $(N/2)$ of $M\ell(f_j)$'s and $Mr(f_j)$'s above $M\ell(f_i)$ and the other half $(N/2)$ of $M\ell(f_j)$'s and $Mr(f_j)$'s below $M\ell(f_i)$. If a $(k.r)$-gathering exists then we remove half of the distances in each of the $M\ell(f_j)$'s and $Mr(f_j)$'s below $M\ell(f_i)$. If a (k,r)-gathering does not exists then we remove half of the distances in each of the $M\ell(f_j)$'s and $Mr(f_j)$'s above $M\ell(f_i)$'s. Thus in any case we remove half of the distances from half of the $M\ell(f_j)$'s and $Mr(f_j)$'s. Thus we remove total $N/2$ weights with one weight from each of the $M\ell(f_j)$'s or $Mr(f_j)$'s from which we removed half of the distances. Let us say that $M\ell(f_j)$ has $2r/2^x$ distances remaining and thus has weight $1+\log r - x$ and we removed half of distances in it and thus removed one weight. Then we removed $(1/(1+\log r - x))$-th $\ge (1/(1+\log r))$-th weight from it. If we pair one $M\ell(f_j)$ from which we removed half of the distances and one $M\ell(f_t)$ from which we did not remove half of the distances and say that $M\ell(f_j)$ has $2r/2^x$ distances and $\log r + 1 - x$ weights and $M\ell(f_t)$ has $2r/2^y$ distances and $\log r + 1 - y$ weights then the one weight we removed from $M\ell(f_j)$ is at least $1/(2(\log r + 1))$-th of the sum of the weights of $M\ell(f_j)$ and $M\ell(f_t)$. This says that in one round we reduce weights from T to at most $T(1 - 1/(2(\log r + 1))$. Initially we have $2|F|(1 + \log r)$ weights. So after $4(1 + \log r)\log r$ rounds the weights are at most

$$2|F|(1+\log r)(1-1/(2(1+\log r)))^{(2(1+\log r))2\log r} = 2|F|(1+\log r)(1/e$$

$$\le 2|F|(1+\log r)(1/2)^{(2\log r)} = 2|F|(1+\log r)/r^2 \le |F|/r.$$

After $4(1 + \log r)\log r$ rounds, as explained above, the weight T is at most $|F|/r$. Since each weight accounts for $2r/2^x$ customers for some $1 \le x \le \log r + 1$, one weight always account for at most r customers. Thus the number of remaining distances is at most $|F|$ because weights $T \le |F|/r$. Note that we have to place back the the last remaining distance in $M\ell(f_j)$'s and $Mr(f_j)$'s where all distances except one have been removed.There are iat most $2|F|$ of them. Thus we have at most $3|F|$ distances remaining.

Finally sort the remaining $3|F|$ remaining distances in $O(|F|\log|F|)$ time, then binary search them $\log(3|F|)$ rounds each of which takes $O(|F|\log r)$ time for computing the left and right boundaries and $O(|F|)$ time for Right-to-Left Minimal Scan and Left-to-Right Maximal Scan. Then we find the minimum cost.

Theorem 3: One can solve the r-gathering problem in $O(|C| + |F|\log^3 r + |F|\log|F|\log r)$ time when all C and F are on the real line. □

5. Tighter Analysis

In this section we analyze the running time of our algorithm in the preceding section more tightly.

We analyze again the running time to compute the boundaries in Section 3, in which we find some indices from $[ll(f_j), lr(f_j)]$ and $[rl(f_j), rr(f_j)]$ for each $f_j \in F$ by binary search. We repeat this in $O(\log^2 r)$ rounds.

For the first round we find the boundaries by binary search from the $2r$ distances. However for later round the number of distances from which we find the boundary is smaller.

Assume that for the first round the number of computation to compute the boundaries is at most $c|F|\log r$ for some constant c. For the second round the number of computation for the boundaries is at most

$$\begin{aligned} & c|F|\log r/2 + c|F|(\log r - 1)/2) && (1)\\ = & c|F|\log r(1/2 + 1/2 - 1/(2\log r)) && (2)\\ = & c|F|\log r(1 - 1/(2\log r)). && (3)\end{aligned}$$

So for the x-th round the number of computation for the boundaries is at most $c|F|\log r(1 - 1/(2\log r))^{x-1}$ Thus the total number of computation for the boundaries for all round is at most

$$c|F|\log r+c|F|\log r(1-1/(2\log r))+\cdots+c|F|\log r(1-1/(2\log r))^{x-1}$$

Except for the computation for the boundaries above and the computation for the weighted median, which runs in $O(|F|)$ time for each round and $O(|F|\log^2 r)$ time in total, the algorithm consists of $O(\log^2 r)$ rounds, in which each round call Right-to-Left Minimal Scan and Left-to-Right Maximal Scan, which runs in $O(|F|)$ time. This will accounts for $O(|C| + |F|\log^2 r)$ time. After that there are $3|F|$ distances remaining, and we use $O(\log |F|)$ rounds in which each x-th round computes the median of $3|F|/2^{x-1}$ distances, computes the left and right boundaries and call Right-to-Left Minimal Scan and Left-to-Right Maximal Scan, which runs in $O(|F|\log r)$ time.

Thus the running time of the algorithm is $O(|C| + |F|\log^2 r + |F|\log|F|\log r)$.

Note that after $M \leq 3|F|$ distances remaining, each round consists of finding the median (value k) in $O(|M|)$ time, compute left and right boundaries and this takes $O(|M|)$ time as follows. Assume that m_i distances are from f_i, that is $co(c, f_i)$ for some c. We have $\sum_i \log m_i = O(M)$ since $\sum_i m_i = M$. Thus we need $O(|F|)$ time for each round and $O(|F|\log|F|)$ time over all rounds.

Theorem 4: Optimal r-gathering of $|C|$ customers and $|F|$ facilities can be found in $O(|C| + |F|\log^2 r + |F|\log|F|)$ time. $\square$

6. Conclusion

In this paper we have given an algorithm to solve the r-gathering problem when all C and F are on the real line. The running time of the algorithm is $O(|C| + |F|\log^2 r + |F|\log|F|)$ and faster than the known algorithm in [4].

References

[1] G. Aggarwal, T. Feder, K. G. Aggarwal, T. ederS, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas and A. Zhu, Achieving anonymity via clustering, Transactions on Algorithms, 6, Article No.49 (2010).

[2] P. Agarwal and M. Sharir, Efficient Algorithms for Geometric Optimization, Computing Surveys, 30, pp.412-458 (1998).

[3] T. Akagi and S. Nakano, On (k, r)-gatherings on a Road, Proc. of Forum on Information Technology, FIT 2013, RA-001 (2013).

[4] T. Akagi and S. Nakano, On r-gatherings on the Line, Proc.of FAW 2015, Guilin, Guangxi, China LNCS 9130, pp.25-32 (2015).

[5] R. J. Anderson, G. L. Miller. Deterministic parallel list ranking. Algorithmica, Vol. 6, pp. 859-868 (1991).

[6] A Armon, On min-max r-gatherings, Theoretical Computer Science, 412, pp.573-582 (2011).

[7] Z. Drezner, Facility Location: A Survey of Applications and Methods, Springer (1995).

[8] Z. Drezner and H.W. Hamacher, Facility Location: Applications and Theory, Springer (2004).

[9] G. Frederickson and D. Johnson, Generalized Selection and Ranking: Sorted Matrices, SIAM Journal on Computing, 13, pp.14-30 (1984).

[10] H. Fournier, and A. Vigneron, Fitting a Step Function to a Point Set, Proc of ESA 2008, Lecture Notes in Computer Science, 5193, pp.442-453 (2008).

[11] J. Y. Liu, A Randomized Algorithm for Weighted Approximation of Points by a Step Function, Proc. of COCOA 2010, Lecture Notes in Computer Science, 6508, pp.300-308 (2010).