Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 23, no. 5, pp. 837–845 (2019)
DOI: 10.7155/jgaa.00514

A Simple Algorithm for r-gatherings on the Line

Shin-ichi Nakano[1]

[1]Gunma University, Kiryu 376-8515, Japan

Abstract

In this paper we study a recently proposed variant of the facility location problem called the r-gathering problem.

Given sets C and F of points on the plane and distance $d(c, f)$ for each $c \in C$ and $f \in F$, an r-gathering of C to F is an assignment A of C to facilities $F^{'} \subset F$ such that r or more customers are assigned to each facility in $F^{'}$. A facility is open in A if at least one customer is assigned to it. The cost of an r-gathering is the maximum distance $d(c, f)$ between $c \in C$ and $A(c) \in F'$ among the assignment, which is $\max_{c \in C}\{d(c, A(c))\}$. The r-gathering problem finds the r-gathering that minimizes the cost. When all points of C and F are on the line, an $O((|C|+|F|)\log(|C|+|F|))$-time algorithm and an $O(|C| + |F|\log^2 r + |F|\log|F|)$-time algorithm to solve the r-gathering problem are known. In this paper we give a simple $O(|C| + r^2|F|)$-time algorithm to solve the r-gathering problem. Since $r << |F| << |C|$ holds in a typical case, say evacuation planning, our new algorithm is $O(\log|F|)$ factor faster than the known algorithms.

We also give an algorithm to solve a simpler problem, called the r-gather-clustering problem, defined as follows. Given a set C of n points on the plane and distance for each pair of points in C, an r-gather-clustering is a partition of the points into clusters such that each cluster has at least r points. The cost of an r-gather-clustering is the maximum radius among the clusters, where the radius of a cluster is the minimum radius of the disk which can cover the points in the cluster. The r-gather-clustering problem is the problem to find the r-gather-clustering that minimizes the cost. In this paper we give an $O(rn)$-time simple algorithm to solve the problem when all points of C are on the line.

Submitted:	Reviewed:	Revised:	Reviewed:
March 2018	November 2018	December 2018	February 2019
Revised:	Accepted:	Final:	Published:
March 2019	April 2019	September 2019	October 2019
Article type:		Communicated by:	
Regular Paper		M.S. Rahman, W.-K. Sung, R. Uehara	

E-mail address: nakano@cs.gunma-u.ac.jp (Shin-ichi Nakano)

1 Introduction

The facility location problem and many of its variants are studied [6, 7]. In this paper we study a recently proposed variant of the problem, the r-gathering problem [2, 5].

We start with a rather simpler problem. Given a set C of n points on the plane an *r-gather-clustering* is a partition of the points into clusters such that each cluster has at least r points. The cost of an r-gather-clustering is the maximum radius among the clusters, where the radius of a cluster is the minimum radius of the disk which can cover the points in the cluster. The r-gather-clustering problem [2] is the problem to find the r-gather-clustering that minimizes the cost. The problem is NP-complete in general, however a polynomial time 2-approximation algorithm for the problem is known [2]. When all points of C are on the line, an $O(n \log n)$-time algorithm, based on the matrix search method [8, 1], for the problem is known [4].

In this paper we give an $O(rn)$-time simple algorithm to solve the problem when all points of C are on the line, by reducing the problem to the min-max path problem [9] in a weighted directed graph.

Assume that C is a set of residents on a street and we wish to locate emergency shelters for the residents so that each shelter serves r or more residents. Then r-gather-clustering computes optimal locations for shelters which minimizes the evacuation time span, where each shelter for a cluster is located at the center of the minimum disk which can cover the residents in the cluster.

Then we consider the r-gathering problem. Given two sets C and F of points on the plane and distance $d(c, f)$ for each $c \in C$ and $f \in F$, an *r-gathering* of C to F is an assignment A of C to $F^{'} \subset F$ such that r or more customers are assigned to each facility in $F^{'}$. A facility is *open* in A if at least one customer is assigned to it. Note that no customer is assigned to each facility in $F \setminus F^{'}$. The cost of an r-gathering is the maximum distance $d(c, f)$ between $c \in C$ and $A(c) \in F'$ among the assignment, which is $\max_{c \in C}\{d(c, A(c))\}$. The r-gathering problem finds the r-gathering that minimizes the cost.

Armon [5] provides a simple 3-approximation algorithm for the r-gathering problem and proves that with the assumption $P \neq NP$ the problem cannot be approximated within a factor less than 3 for any $r \geq 3$. When all points of C and F are on the line, an $O((|C| + |F|) \log(|C| + |F|))$-time algorithm [4] and an $O(|C| + |F| \log^2 r + |F| \log |F|)$-time algorithm [10] to solve the r-gathering problem are known.

In this paper we give an $O(|C| + r^2|F|)$-time algorithm to solve the r-gathering problem when all points of C and F are on the line. Since $r << |F| << |C|$ holds in a typical case, say evacuation planning, our new algorithm is $O(\log |F|)$ factor faster than the known algorithms.

Assume that we are planning an evacuation plan for the residents on a street, F is a set of possible locations for emergency shelters, and $d(c, f)$ is the time needed for a person $c \in C$ to reach a shelter $f \in F$. Then, an r-gathering (when all points of C and F are on the line) corresponds to an evacuation assignment such that each open shelter serves r or more people, and the r-gathering problem finds an evacuation plan that minimizes the evacuation time span.

The remainder of this paper is organized as follows. In Section 2 we consider the r-gather-clustering problem and give an algorithm when all points in C are on the line. The idea of the algorithm is to reduce the problem to the min-max path problem for a weighted directed graph. Then in Section 3 we give our algorithm for the r-gathering problem when all points in C and F are on the line. The idea of our algorithm is (1) to reduce the problem to the min-max path problem for a weighted directed graph, and (2) carefully bounding the number of edges in the graph. Finally, Section 4 is a conclusion.

2 r-gather-clustering on the line

In this section we consider the r-gather-clustering problem, and give an algorithm when all points in C are on the line. Let $C = \{c_1, c_2, \cdots, c_n\}$ be points on the horizontal line and we assume they are sorted from left to right. Our idea is to reduce the r-gather-clustering problem to the min-max path problem in a weighted directed (acyclic) graph. First we have the following two lemmas.

Lemma 1 *There exists a solution in which the points in each cluster are consecutive in C.*

Proof: We call three points $c_a, c_b, c_c \in C$ *crossing triple* if (1) $a < b < c$, (2) $c_a, c_c \in C_x$ and (3) $c_b \in C_y$ where C_x and C_y are clusters.

We provide a proof by contradiction. We assume any solution has some crossing triple. Let S be a solution of the r-gather-clustering problem with the minimum number of crossing triples, and S has a crossing triple c_a, c_b, c_c with $c_a, c_c \in C_x$ and $c_b \in C_y$. Let $S^{'}$ be the r-gather-clustering derived from S by replacing C_x and C_y by $C_x^{'}$ and $C_y^{'}$ so that $C_x^{'}$ are the leftmost $|C_x|$ points in $C_x \cup C_y$ and $C_y^{'}$ are the rightmost $|C_y|$ points in $C_x \cup C_y$. Now, the cost of $S^{'}$ does not increase and $S^{'}$ has less crossing triples than S, which is a contradiction. □

Thus we can assume each cluster in a solution consists of consecutive points $\{c_i, c_{i+1}, \cdots, c_j\}$ for some i and j.

Lemma 2 *There exists a solution in which the number of points in each cluster is at most $2r - 1$.*

Proof: We provide a proof by contradiction. Assume that a solution contains a set of clusters $C_1, C_2, \cdots, C_k$ where each cluster C_i has more than $2r$ points. We divide every cluster C_i into two (or more) clusters, respectively, so that each of the divided clusters has r or more points, but at most $2r - 1$ points. Since this modification does not increase the cost, the resulting clustering is also a solution. □

Then we define the directed (acyclic) graph $D(V, E)$ and the weight of each edge, as follows.

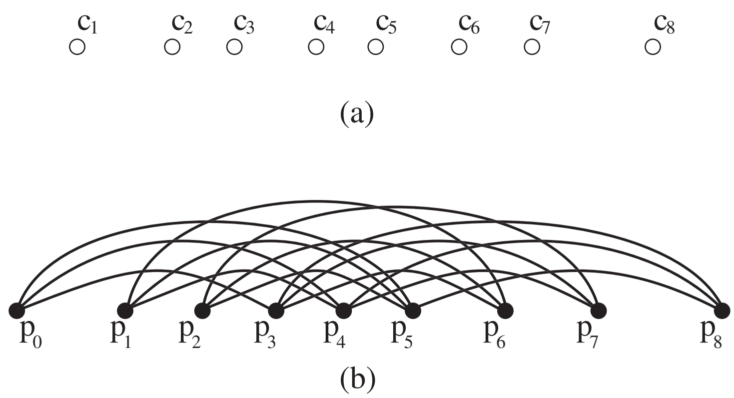

Figure 1: (a) A point set C and (b) the weighted directed graph D with $r = 3$.

$$V = \{p_0, p_1, p_2, \cdots, p_n\}$$

$$E = \{(p_i, p_j) | i + r \leq j \leq i + 2r - 1\}$$

See an example with $r = 3$ in Figure 1. Note that $|V| = |C| + 1$ because of p_0. Each edge is directed from left to right. Also note that the number $|E|$ of edges is at most rn. The weight of an edge (p_i, p_j) is the half of the distance between c_{i+1} and c_j, and denoted by $w(p_i, p_j)$.

The cost of a directed path from p_0 to p_n is defined by the weight of the edge having the maximum weight in the directed path. *The min-max path* from p_0 to p_n is the directed path from p_0 to p_n with the minimum cost.

Now C has an r-gather-clustering with cost k iff $D(V, E)$ has a directed path from p_0 to p_n with cost k. See Figure 2. Intuitively, each (directed) edge in the min-max path corresponds to a cluster of an r-gather-clustering. Edge (p_i, p_j) in the directed path corresponds to cluster $\{c_{i+1}, c_{i+2}, \cdots, c_j\}$ in the r-gather-clustering. If $|C| \geq r$ we can partition C into subsets so that each subset consists of consecutive points with at most $2r - 1$ points and also at least r points, so there always exists a path from p_0 to p_n.

Thus if we can compute the min-max path in D then it corresponds to the solution of the r-gather-clustering problem.

We can construct the $D(V, E)$ in $O(rn)$ time. An $O(|E| \log^* |V|)$ time algorithm for the min-max path problem for a directed graph $D = (V, E)$ is known [9]. However, since $D(V, E)$ is a DAG (directed acyclic graph) we can compute the min-max path from p_0 to p_n in $O(|E|)$ time by a simple dynamic programming algorithm. (Let w_i be the cost of the min-max path from p_0 to p_i. For each p_i, we can compute w_i by checking each incoming edge (p_x, p_i) to p_i and the cost w_x of the min-max path from p_0 to p_x.)

Thus we have the following theorem.

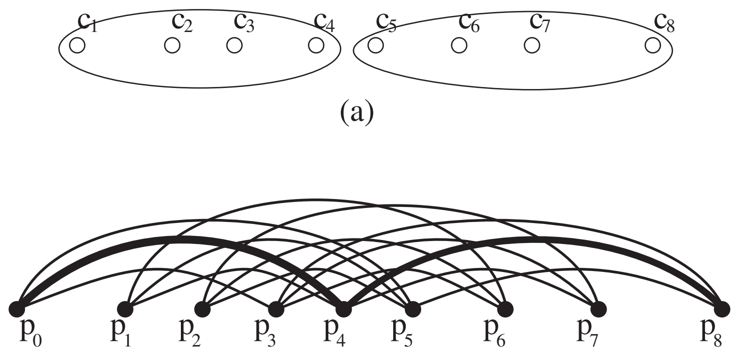

Figure 2: (a) An r-gather-clustering and (b) its corresponding min-max path of D.

Theorem 1 *One can solve the r-gather-clustering problem in $O(rn)$ time, when all points in C are on the line.*

3 r-gathering on the line

In this section we give an algorithm for the r-gathering problem when all points in C and F are on the line, by reducing the problem to the min-max path problem for a weighted directed graph, and bounding the number of edges in the graph.

Let $C = \{c_1, c_2, \cdots, c_n\}$ and $F = \{f_1, f_2, \cdots, f_m\}$ be points on the horizontal line and we assume they are sorted from left to right, respectively. Similar to Lemma 1 we can assume the set of points assigned to a facility are consecutive in a solution.

Lemma 3 *There exists a solution in which the set of points assigned to a facility are consecutive in C.*

Proof: We call three points $c_a, c_b, c_c \in C$ *crossing triple* if (1) $a < b < c$, (2) $A(c_a) = A(c_c) = f_x \in F'$ and (3) $A(c_b) = f_y \in F'$.

We provide a proof by contradiction. We assume any solution has some crossing triple. Let A be the solution of the r-gathering problem with the minimum number of crossing triples, and A has a crossing triple c_a, c_b, c_c with $A(c_a) = A(c_c) = f_x$ and $A(c_b) = f_y$. Without loss of generality we can assume $x < y$.

Let C_x be the set of points in C assigned to f_x in A, and C_y the set of points in C assigned to f_y. Let C_x' be the set of leftmost $|C_x|$ points in $C_x \cup C_y$, and C_y' the set of rightmost $|C_y|$ points in $C_x \cup C_y$. Let A' be the r-gathering derived from A by slightly modifying A so that reassign C_x' to f_x and C_y' to f_y. Now we show $\max_{c \in C_y'}\{d(c, A'(c))\} \leq \max_{c \in C_y}\{d(c, A(c))\}$ holds as follows.

Let c_ℓ be the leftmost point in C assigned to f_y in A', and c_r the rightmost point in C assigned to f_y. We consider two cases. In each case the cost of A' is less than or equal to A. A contradiction.

Case 1: $d(c_\ell, f_y) \leq d(c_r, f_y)$.

If $A(c_r) = f_x$ then $d(c_r, f_y) \leq d(c_r, f_x)$ holds. Otherwise $A(c_r) = A'(c_r) = f_y$. Thus the cost of A' is less than or equal to A.

Case 2: $d(c_\ell, f_y) > d(c_r, f_y)$.

Let c_z be the leftmost point assigned to f_y in A. Now $z < \ell$ holds so $d(c_\ell, f_y) < d(c_z, f_y)$ holds. Thus the cost of A' is less than or equal to A.

Similarly we can show $\max_{c \in C_x'}\{d(c, A'(c))\} \leq \max_{c \in C_x}\{d(c, A(c))\}$ holds. □

For consecutive three facilities f_{j-1}, f_j and f_{j+1} in F let m_L be the midpoint of f_{j-1} and f_j, and m_R the midpoint of f_j and f_{j+1}. We have the following two lemmas.

Lemma 4 *Assume that C has $2r$ or more points on the left of m_L. Let c_i be the $2r$-th point from right in C' where C' is the set of points in C on or left of m_L. There exists a solution in which $c_{i'}$ with $i' < i$ is never assigned to f_j.*

Proof: Assume for a contradiction such $c_{i'}$ is assigned to f_j. We have two cases.

If the rightmost point assigned to f_j is on the left of m_L then reassigning the points assigned to f_j to f_{j-1} results in a new r-gathering and since it does not increase the cost the resulting r-gathering is also a solution of the given r-gathering problem.

Otherwise, the rightmost point assigned to f_j is on or right of m_L. Then at least $2r$ points on or left of m_L are assigned to f_j by Lemma 3, with other points on the right of m_L. Let C' be the subset of C consisting of the points (1) assigned to f_j, (2) on or left of m_L, and (3) but not the rightmost r points on or left of m_L. Note that $|C'| \geq r$ holds and C' contains $c_{i'}$. Reassigning the points in C' to f_{j-1} results in a new r-gathering and the resulting r-gathering is also a solution since it does not increase the cost. □

Intuitively if $c_{i'}$ is too far from f_j then $c_{i'}$ is never assigned to f_j. Symmetrically we have the following lemma.

Lemma 5 *If C has $2r$ or more points on the right of m_R, then $c_{i'}$ with $i' > i$ is never assigned to f_j, where c_i is the $2r$-th point in C on or right of m_R.*

We have more lemmas. Let C' be the set of points between m_L and m_R except the leftmost $2r$ points and the rightmost $2r$ points.

Lemma 6 *If C has $5r$ or more points between m_L and m_R, then the customers in C' are assigned to f_j in a solution of the r-gathering problem.*

Proof: Immediate from the two lemmas above. □

Thus if we can compute the solution for $C - C'$, then appending the assignment from the points in C' to f_j results in the solution for C. From now on assume that we have removed every such C' from C.

We have more lemmas for the boundary case. Let m be the midpoints of f_1 and f_2 in F.

Lemma 7 *If C has $2r$ or more points on the left of m, then each $c_{i'}$ with $i' < i$ is assigned to f_1 in a solution of the r-gathering problem, where c_i is the $2r$-th customer in C on the left of m.*

Proof: Immediate from Lemma 4. □

Let m be the midpoints of f_{m-1} and f_m in F.

Lemma 8 *If C has $2r$ or more points on the right of m, then each $c_{i'}$ with $i' > i$ is assigned to f_m in a solution of the r-gathering problem, where c_i is the $2r$-th customer in C on the right of m.*

Thus we have the following lemma.

Lemma 9 *The number of points in C possibly assigning to each facility $f \in F$ is at most $9r$.*

Proof: For each f_j with $1 < j < m$ define m_L and m_R as above. The number of points possibly assigning to f_j is (1) at most $2r$ on the left of m_L, (2) at most $2r$ on the right of m_R, and (3) at most $5r$ between m_L and m_R, by the lemmas above. Similar for f_1 and f_m. □

Now we are going to define a weighted directed graph $D(V, E)$ for F and C, and the weight of each edge.

The set of vertices is defined as follows.

$$V = \{p_0, p_1, p_2, \cdots, p_n\}$$

For each facility f_h with $h = 2, 3, \cdots, m-1$ and its possible cluster consisting of points $\{c_{i+1}, c_{i+2}, \cdots, c_j\}$ we define an edge (p_i, p_j). So (p_i, p_j) is an edge iff
(1) $i + r \leq j \leq i + 2r - 1$
(2) $i \geq i'$ where $c_{i'}$ is the $2r$-th customer on the left of m_L, and
(3) $j \leq j'$ where $c_{j'}$ is the $2r$-th customer on the right of m_R,
where m_L and m_R are defined for f_h as above. Let E_h be the set of edges consisting of edges defined for f_h above. Similary we define E_1 and E_m.

Finally,

$$E = E_1 \cup E_2 \cup \cdots \cup E_m$$

Note that E may contain many multi-edges.

The weight of an edge (p_i, p_j) for f_h is the maximum of (1) the distance between c_{i+1} and f_h, and (2) the distance between c_j and f_h.

The cost of a directed path from p_0 to p_n is defined by the weight of the edge having the maximum weight in the directed path. *The min-max path* from p_0 to p_n is the directed path from p_0 to p_n with the minimum cost.

We need to compute for each f_h the $2r$-th customer on the left of m_L and the $2r$-th customer on the right of m_R. By scanning the line we can compute them for all f_h in $O(|F|+|C|)$ time in total. Note that each edge in E corresponds to a pair of customers possibly assigning to a common facility. Thus the number of the edges in E is at most $81r^2|F|$ by Lemma 9. Thus we can construct $D(V,E)$ in $O(|F|+|C|+r^2|F|)$ time in total.

Now, similar to Section 2, there is an r-gathering with cost k iff $D(V,E)$ has a directed path from p_0 to p_n with cost k.

Theorem 2 *When both C and F are on the line one can solve the r-gathering problem in $O(n+r^2m)$ time, where $n=|C|$ and $m=|F|$.*

4 Conclusion

In this paper we have presented an algorithm to solve the r-gather-clustering problem when all points of C are on the line. The running time of the algorithm is $O(rn)$, where $n=|C|$. We also presented an algorithm to solve the r-gathering problem, which runs in time $O(n+r^2m)$, where $n=|C|$ and $m=|F|<n$.

Can we solve the problem more efficiently or can we solve the problem for more general input or cost?

Recently $O(n+m)$-time algorithm to solve the problem is reported [11]. Also an algorithm to solve the problem for a star is reported [3].

Acknowledgements

This work is partially supported by Grant-in-Aid for Scientific Research (KAKENHI) 17K00003.

References

[1] P. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *Computing Surveys*, 30:412–458, 1998. doi:10.1145/299917.299918.

[2] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu. Achieving anonymity via clustering. *Transactions on Algorithms*, 6:Article No.49, 2010. doi:10.1145/1798596.1798602.

[3] S. Ahmed, S. Nakano, and M. S. Rahman. r-gatherings on a star. *Proc. of WALCOM2019*, pages 31–42, 2019. doi:10.1007/978-3-030-10564-8_3.

[4] T. Akagi and S. Nakano. On r-gatherings on the line. *Proc.of FAW 2015*, LNCS 9130:25–32, 2015. doi:10.1007/978-3-319-19647-3_3.

[5] A. Armon. On min-max r-gatherings. *Theoretical Computer Science*, 412:573–582, 2011. doi:10.1016/j.tcs.2010.04.040.

[6] Z. Drezner. *Facility Location: A Survey of Applications and Methods.* Springer, 1995.

[7] Z. Drezner and H. Hamacher. *Facility Location: Applications and Theory.* Springer, 2004.

[8] G. Frederickson and D. Johnson. Generalized selection and ranking: Sorted matrices. *SIAM Journal on Computing*, 13:14–30, 1984. doi:10.1137/0213002.

[9] H. Gabow and R. Tarjan. Algorithms for two bottleneck optimization problems. *J. of Algorithms*, 9:411–417, 1988. doi:10.1016/0196-6774(88)90031-4.

[10] Y. Han and S. Nakano. On r-gatherings on the line. *Proc. of FCS2016*, pages 99–104, 2016.

[11] A. Sarker, W. Sung, and M. S. Rahman. A linear time algorithm for the r-gathering problem on the line. *Proc. of WALCOM2019*, pages 56–66, 2019. doi:10.1007/978-3-030-10564-8_5.