On \((k, r)\)-gatherings on a Road

Toshihiro Akagi\(^\dagger\)  
Shin-ichi Nakano\(^\dagger\)

Abstract  Given two integers \(k\) and \(r\), a set of customer locations \(C\), and a set of potential facility locations \(F\), we wish to compute an assignment \(A\) of \(C\) to \(F\) such that (1) for each \(c \in C\) the distance between \(c\) and \(A(c) \in F\) is at most \(k\), and (2) for each \(f \in F\) the number of customers assigned to \(f\) is either zero or at least \(r\). Such an assignment is called a \((k, r)\)-gathering of \(C\) to \(F\). Intuitively we wish to assign customers to near “open” facilities so that each “open” facility has an enough number of customers, namely \(r\) or more customers.

In this paper we solve the problem in linear time when the customer locations and potential facility locations are on a line.

1 Introduction

Given two integers \(k\) and \(r\), a set of customer locations \(C\), and a set of potential facility locations \(F\), we wish to compute an assignment \(A\) of \(C\) to \(F\) such that (1) for each \(c \in C\) the distance between \(c\) and \(A(c) \in F\) is at most \(k\), and (2) for each \(f \in F\) the number of customers assigned to \(f\) is either zero or at least \(r\). Such an assignment is called a \((k, r)\)-gathering of \(C\) to \(F\). See some examples in Fig. 1 and 2. We say a facility \(f\) is open in an assignment if the number of customers assigned to \(f\) is at least \(r\). Intuitively we wish to assign customers to near open facilities so that each open facility has an enough number of customers, namely \(r\) or more customers. This is a variant of the \(r\)-gathering problem in [1]. However the graph version of the problem is NP-hard even for \(k = 1\) and \(r = 3\) [1].

In this paper we solve the problem in linear time when the customer locations and potential facility locations are on a line.

\[ \begin{array}{c}
\text{customer} & \text{facility} & k = 4 & r = 3 \\
\end{array} \]

Figure 1: An example of a non-overlapping \((k, r)\)-gathering.

\[ \begin{array}{c}
\text{customer} & \text{facility} & k = 4 & r = 3 \\
\end{array} \]

Figure 2: An example of an overlapping \((k, r)\)-gathering.

\(^\dagger\)Department of Computer Science, Gunma University
We regard $C$ as the set of distinct coordinates of the customers, and $F$ as the set of distinct coordinates of the potential facilities. In this paper we assume the elements in $C$ and $F$ are on the horizontal axis and sorted respectively.

If some customer has no facility within distance $k$ then no $(k, r)$-gathering exists, so we assume no such customer exists in $C$. If some facility has less than $r$ customers within distance $k$ then the facility is never open in any $(k, r)$-gathering, so we assume no such potential facility exists in $F$.

## 2 First Algorithm

In this section we design our first algorithm for the $(k, r)$-gathering problem, and all $c \in C$ and $f \in F$ are integers. The algorithm is a simple dynamic programming. First we define subproblems for our dynamic programming.

Given two integers $x$ and $s$, let $C(x, s)$ be the subset of $C$ consisting of the coordinates of customers with $x + k - s$ or less, and $F(x)$ be the subset of $F$ consisting of the coordinates of potential facilities with $x$ or less. An assignment $A$ of $C(x, s)$ to $F(x)$ is called a partial gathering of $C(x, s)$ to $F(x)$ if (1) $A$ is a $(k, r)$-gathering of $C(x, s)$ to $F(x)$, and (2) the facility $f$ at $x$ is open. Intuitively this is a $(k, r)$-gathering of customers locating “left” of open facility $f$ at $x$, in which the customers in the rightmost $s$ locations of $f$ are not assigned yet. A $(k, r)$-gathering of $C$ to $F$, a solution of the original problem, exists if and only if a partial gathering of $C(x, 0)$ to $F(x)$ exists for some facility at $x$ within distance $k$ from the rightmost customer. Let $PG(x, s)$ be the subproblem which ask for the existence of a partial gathering of $C(x, s)$ to $F(x)$.

We have the following fact.

**Fact 1** If $PG(x, s)$ has a solution with $s > 1$, then $PG(x, s - 1)$ has a solution.

We have two lemmas. We say a solution of $PG(x, s)$ is overlapping if for some pair of facilities the two intervals induced by the assigned customers overlap. The assignment in Fig. 1 is non-overlapping, while the assignment in Fig. 2 is overlapping.

**Lemma 1** If $PG(x, s)$ has a solution then $PG(x, s)$ has a non-overlapping solution.

**Proof** We say a pair of customers $(c_l, c_r)$ is a reverse pair in an assignment $A$ if $A(c_l) > A(c_r)$ and $c_l < c_r$. Assume $PG(x, s)$ has only overlapping solutions. Let $A$ be a solution with the minimum number of reverse pairs. Then by swapping the assignments of a reverse pair $c_l$ and $c_r$, a solution with less reverse pairs is obtained. A contradiction.

**Lemma 2** For each $x \in F$, $PG(x, s)$ has a solution if and only if either

(i) $|x - c| \leq k$ for each $c \in C(x, s)$ and $|C(x, s)| \geq r$, or

(ii) for some $x' < x$ and $s' \in [0, 2k]$, $PG(x', s')$ has a solution such that either

(a) $x' + k < x - k$ and $(x' + k, x - k)$ has no customer location, and $[x - k, x + k - s]$ contains $r$ or more customers,

(b) $x' + k \geq x - k$, $x' + k - s' < x - k$ and $[x - k, x + k - s]$ contains $r$ or more customers, or

(c) $x' + k - s' \geq x - k$ and $(x' + k - s', x + k - s]$ contains $r$ or more customers.
Proof Assume $PG(x, s)$ has a solution. Then $PG(x, s)$ has a non-overlapping solution $A$ by Lemma 1. We have two cases. If all customers are assigned to the facility at $x$ then (i) holds. Otherwise let $x'$ be the open facilities in $A$ with the maximum coordinate except $x$. Then (ii) holds.

Assume either (i) or (ii) holds. If (i) holds then $PG(x, s)$ has a simple $(k, r)$-gathering, in which all customers are assigned to the facility at $x$. If (ii) holds then by assigning the customers in either $[x - k, x + k - s]$ or $(x' + k - s', x + k - s]$ to the facility at $x$ we can extend a solution of $PG(x', s')$ to a solution of $PG(x, s)$. \[\Box\]

Intuitively above condition means $PG(x, s)$ has a solution if and only if either (i) the facility at $x$ can serve all customers and the number of customers is $r$ or more, or (ii) some smaller subproblem $PG(x', s')$ has a partial gathering $A'$ and an assignment of customers around the facility $f$ at $x$ to $f$ can be “patched” to $A'$ to construct a partial gathering of $PG(x, s)$, in which the customers in the rightmost $s$ locations of $f$ are reserved, and possibly to be assigned to some facility on the right.

Based on Lemma 2 we have the following algorithm. If subproblem $PG(x, s)$ has a solution then the algorithm sets $Ans(x, s) = “exists”$, otherwise sets $Ans(x, s) = “not exist”$. 

Figure 3: Illustrations for condition (ii) of Lemma 2.
**Algorithm find-gathering**(C, F, k, r)

01. let $c_{\text{min}}$ and $c_{\text{max}}$ be the minimum and the maximum in C
02. for each facility $x \in F$ with $|x - c_{\text{min}}| \leq k$ (in increasing order)
    03. for each $s \in [0, 2k]$
        04. if $|C(x, s)| \geq r$
            05. then $\text{Ans}(x, s) = \text{“exists”}$
            06. else $\text{Ans}(x, s) = \text{“not exist”}$
07. for each facility $x \in F$ with $|x - c_{\text{min}}| > k$ (in increasing order)
08. for each $s \in [0, 2k]$
    09. begin
        10. $\text{Ans}(x, s) = \text{“not exist”}$
        11. for each facility $x'$ with $x' < x$
            12. for each $s' \in [0, 2k]$
                13. if $\text{Ans}(x', s') = \text{“exists”}$ and either
                    (a) $x' + k < x - k$ and $(x' + k, x - k)$ has no customer and $[x - k, x + k - s']$ contains $r$ or more customers,
                    (b) $x' + k \geq x - k$ and $x' + k - s' < x - k$ and $[x - k, x + k - s']$ contains $r$ or more customers, or
                    (c) $x' + k - s' \geq x - k$ and $(x' + k - s', x + k - s]$ contains $r$ or more customers
                14. then $\text{Ans}(x, s) = \text{“exists”}$
            15. end
        16. $\text{Ans} = \text{“not exist”}$
        17. for each facility $x \in F$ with $|x - c_{\text{max}}| \leq k$
            18. if $\text{Ans}(x, 0) = \text{“exists”}$
                19. then $\text{Ans} = \text{“exists”}$
        20. return $\text{Ans}$

As a preprocessing we prepare the following data in three arrays $L$, $R$, and $r$. For each facility $f \in F$ we compute the total number $L(f)$ of customers less than $f - k$ and the total number $R(f)$ of customers less than or equal to $f + k$ Then $(x' + k, x - k)$ has no customer iff $R(x') = L(x)$, and we can check this in constant time. Also for each $f \in F$ we compute the location $r(f)$ of the $r$-th smallest customer in $[f - k, f + k]$, which always exists by the assumption in Section 1. Then $[x - k, x + k - s']$ contains $r$ or more customers iff $r(f) \leq x + k - s$, and we can check this in constant time. One can compute $L$ by the following algorithm as preprocessing, and also compute $R$ and $r$ similarly. Assume $C = \{c_0, c_1, \ldots, c_{|C| - 1}\}$. However in the algorithm above we need $O(k)$ time to check if $(x' + k - s', x + k - s]$ contains $r$ or more customers.

**Algorithm $L(C, F, k)$**

1. \( j \leftarrow 0 \)
2. for each $f_i \in F$ (in increasing order)
    3. begin
        4. while $c_j < f_i - k$
            5. \( j \leftarrow j + 1 \)
        6. $L(f_i) = j$
    7. end
Lemma 3 One can solve the \((k, r)\)-gathering problem in \(O(k^3|F|^2)\) time if all customers and facilities are on a line with integer coordinates.

3 Improvement I

In this section we design a faster dynamic programming algorithm. The algorithm in Section 2 computes the existence of a solution of \(PG(x, s)\) for each \(x\) and \(s\). Let \(s(x)\) be the maximum \(s\) such that \(PG(x, s)\) has a solution, and \(PG(x)\) be the subproblem which ask for \(s(x)\). For convenience let \(s(x) = -1\) if \(PG(x, s)\) has no solution for any \(s \in [0, 2k]\). By Fact 1, \(PG(x, s)\) has a solution for each \(s \leq s(x)\) and has no solution for \(s > s(x)\). Thus if we have just \(s(x)\) then we know whether each of \(PG(x, 0), PG(x, 1), \ldots, PG(x, 2k)\) has a solution or not.

We have the following algorithm.

<table>
<thead>
<tr>
<th>Algorithm find-gathering2((C, F, k, r))</th>
</tr>
</thead>
<tbody>
<tr>
<td>01. let (c_{min}) and (c_{max}) be the minimum and the maximum in (C)</td>
</tr>
<tr>
<td>02. let (c_r) be the (r)-th smallest customer in (C)</td>
</tr>
<tr>
<td>03. for each facility (x_i \in F) with (</td>
</tr>
<tr>
<td>(/ * [x_i - k, x_i + k] ) contains (r) or more customers */</td>
</tr>
<tr>
<td>04. (s(x_i) = x_i + k - c_r)</td>
</tr>
<tr>
<td>05. for each facility (x_i \in F) with (</td>
</tr>
<tr>
<td>06. (s(x_i) = -1)</td>
</tr>
<tr>
<td>07. (s(x_i)) if (s(x_i) \neq -1) and (x_j + k &lt; x_i - k) and ((x_j + k, x_i - k)) has no customer</td>
</tr>
<tr>
<td>08. (\text{then } s(x_i) = \max{x_i + k - c', s(x_i)}) where (c') is the (r)-th smallest customer in ([x_i - k, x_i + k])</td>
</tr>
<tr>
<td>(/ * [x_i - k, x_i + k] ) contains (r) or more customers */</td>
</tr>
<tr>
<td>09. else if (s(x_i) \neq -1) and (x_j + k \geq x_i - k) and (x_j + k - s(x_j) &lt; x_i - k)</td>
</tr>
<tr>
<td>10. (\text{then } s(x_i) = \max{x_i + k - c', s(x_i)}) where (c') is the (r)-th smallest customer in ([x_i - k, x_i + k])</td>
</tr>
<tr>
<td>11. else if (s(x_j) \neq -1) and (x_j + k - s(x_j) \geq x_i - k) and</td>
</tr>
<tr>
<td>((x_j + k - s(x_j), x_i + k)) contains (r) or more customers</td>
</tr>
<tr>
<td>12. (\text{then } s(x_i) = \max{x_j + k - c', s(x_j)})</td>
</tr>
<tr>
<td>(c') is the (r)-th smallest customer in ((x_j + k - s(x_j), x_i + k))</td>
</tr>
<tr>
<td>13. end</td>
</tr>
<tr>
<td>14. (\text{Ans = “not exist”})</td>
</tr>
<tr>
<td>15. (\text{for each facility } x_i \in F ) with (</td>
</tr>
<tr>
<td>16. (\text{if } s(x_i) \geq 0)</td>
</tr>
<tr>
<td>17. (\text{then } \text{Ans = “exists”})</td>
</tr>
<tr>
<td>18. (\text{return } \text{Ans})</td>
</tr>
</tbody>
</table>

In this algorithm all \(c \in C\) and \(x_i \in F\) are allowed to be real numbers, and not restricted to integers. We store the coordinate of \(x_i \in F\) and \(c_j \in C\) in an array respectively and access it with its index. For each facility \(x_i \in F\) we compute the index of the smallest customer \(c_j\) with \(x_i - k \leq c_j\), and the index of the largest customer \(c_j\) with \(c_j \leq x_i + k\). We additionally store the index \(t\) of the customer at \(x_j + k - s(x_j)\) for each \(x_j \in F\). Then we can compute the \(r\)-th smallest customer in \((x_j + k - s(x_j), x_i + k)\) in constant time, since it is \(c_{t+r}\).

Lemma 4 One can solve the \((k, r)\)-gathering problem in \(O(|F|^2)\) time if all customers and facilities are on a line.
4 Improvement II

In this section we design a linear time algorithm. Our idea is the following two lemmas. Let $c_{\text{min}}$ be the minimum in $C$.

**Lemma 5** For two facilities $x_l$ and $x_r$, if $x_l < x_r$, $s(x_l) \neq -1$ and $s(x_r) \neq -1$, then $x_l + k - s(x_l) \leq x_r + k - s(x_r)$.

**Proof** If $x_l + k \leq x_r - k$ then it is clear. Otherwise $x_l + k > x_r - k$. Assume for the contradiction that $x_l + k - s(x_l) > x_r + k - s(x_r)$ holds. Since $s(x_r) \neq -1$, a $(k, r)$-gathering $A_r$ of $C(x_r, s(x_r))$ to $F(x_r)$ exists. Modify $A_r$ so that the customers assigned to $x_r$ to be assigned to $x_l$. The resulting assignment is a $(k, r)$-gathering of $C(x_l, s')$ to $F(x_l)$ with some $s' > s(x_l)$. A contradiction.

**Lemma 6** $s(x_i) \neq -1$ if and only if either

(1) $|x_i - c_{\text{min}}| \leq k$,

(2) $|x_i - c_{\text{min}}| > k$ and $(x_c + k, x_i - k)$ has no customer

where $x_c$ is the facility having the maximum coordinate satisfying $x_c + k < x_i - k$ and $s(x_c) \neq -1$,

(3) $|x_i - c_{\text{min}}| > k$, $x_c + k > x_i - k$, $x_c + k - s(x_c) < x_i - k$, and $[x_i - k, x_i + k]$ contains $r$ or more customers where $x_c$ is the facility having the minimum coordinate satisfying $x_c + k \geq x_i - k$ and $s(x_c) \neq -1$, or

(4) $|x_i - c_{\text{min}}| > k$, $x_c + k - s(x_c) \geq x_i - k$ and $(x_c + k - s(x_c), x_i + k]$ contains $r$ or more customers where $x_c$ is the facility having the minimum coordinate satisfying $x_c + k \geq x_i - k$ and $s(x_c) \neq -1$.

**Proof** Assume $s(x_i) \neq -1$ then $\text{PG}(x_i)$ has some solution. Then $\text{PG}(x_i)$ has a non-overlapping solution $A$ by Lemma 1. We have two cases. If all customers are assigned to $x_i$ in $A$ then (i) holds. Otherwise $A$ has two or more open facilities and $|x_i - c_{\text{min}}| > k$ holds. Let $x'_c$ be the open facility in $A$ having the maximum coordinate except $x_i$. If $x'_c + k < x_i - k$ then, since $(x'_c + k, x_i - k)$ has no customer and $x_c \geq x'_c$, (ii) holds. Otherwise $x'_c + k \geq x_i - k$ holds. If $x'_c + k - s(x'_c) < x_i - k$ then, since $x_c \leq x'_c$ and Lemma 5 means $x_c + k - s(x_c) \leq x'_c + k - s(x'_c)$, (iii) holds. Otherwise $x'_c + k - s(x'_c) \geq x_i - k$ then similarly either (iii) or (iv) holds.

Assume either (i), (ii), (iii) or (iv) holds. If (i) holds then $\text{PG}(x_i)$ has a simple $(k, r)$-gathering, in which all customers are assigned to $x_i$. If either (ii), (iii) or (iv) holds then by assigning the customers in either $[x_i - k, x_i + k]$ or $(x_c + k - s(x_c), x_i + k]$ to $x_i$ we can extend a solution of $\text{PG}(x_c)$ to a solution of $\text{PG}(x_i)$. A contradiction.

Now by Lemma 6 we can skip many parts of Algorithm find-gathering2. We have the following algorithm.
Algorithm `find-gathering3(C, F, k, r)`

01. let $c_{\text{min}}$ and $c_{\text{max}}$ be the minimum and the maximum in $C$
02. let $c_r$ be the $r$-th smallest coordinate in $C$
03. for each facility $x_i \in F$ with $|x_i - c_{\text{min}}| \leq k$ (in increasing order) /* condition (i) */
04. /* $[x_i - k, x_i + k]$ contains $r$ or more customers */
05. $s(x_i) = x_i + k - c_r$
06. for each facility $x_i \in F$ with $|x_i - c_{\text{min}}| > k$ (in increasing order)
07. begin
08. $s(x_i) = -1$
09. if there is a facility $x_c$ satisfying $x_c + k < x_i - k$ and $s(x_c) \neq -1$ then /* condition (ii) */
10. begin
11. let $x_c$ be the maximum of such facilities
12. if $(x_c + k, x_i - k)$ has no customer
13. then $s(x_i) = x_i + k - c'$ where $c'$ is the $r$-th smallest customer in $[x_i - k, x_i + k]$
14. end
15. if there is a facility $x_c$ satisfying $x_c + k \geq x_i - k$ and $s(x_c) \neq -1$ then /* condition(iii), (iv) */
16. begin
17. let $x_c$ be the minimum of such facilities
18. if both $(x_c + k - s(x_c), x_i + k]$ and $[x_i - k, x_i + k]$ has $r$ or more customers then
19. if $x_c + k - s(x_c) \geq x_i - k$
20. then $s(x_i) = x_i + k - c''$ where $c''$ is the $r$-th smallest customer in $(x_c + k - s(x_c), x_i + k]$
21. else $s(x_i) = x_i + k - c''$ where $c''$ is the $r$-th smallest customer in $[x_i - k, x_i + k]$
22. end
23. end
24. Ans = “not exist”
25. for each facility $x_i \in F$ with $|x_i - c_{\text{max}}| \leq k$
26. if $s(x_i) > 0$
27. then Ans = “exists”
28. return Ans

To check the condition (ii) of Lemma 6 efficiently we maintain $x_c$ for the current $x_i$ so that $x_c$ is the facility having the maximum $x_c$ satisfying $(1) x_c + k < x_i - k$ and $(2) s(x_c) \neq -1$. We need $O(|C| + |F|)$ time in total for this maintenance. Also to find $x_c$ in line 17 efficiently we maintain the list $L$ of “useful facilities” for the current $x_i$ so that $L$ contains all facilities $x_j$ satisfying $(1) x_j + k \geq x_i - k$, $(2) s(x_j) \neq -1$. Then we can find $x_c$ in line 17 in constant time. We need $O(|C| + |F|)$ time in total for this maintenance. Thus we have the following theorem.

**Theorem 1** One can solve the $(k, r)$-gathering problem in $O(|C| + |F|)$ time if all customers and facilities are on a line.

By a simple dynamic programming algorithm on array $s(x_i)$ one can compute a $(k, r)$-gathering with the minimum number of open facilities.
5 Conclusion

In this paper we designed a linear time algorithm to solve the \((k, r)\)-gathering problem when the customers and facilities are on a line.

If each customer has a weight, and the sum of the weights of the customers assigned to each open facility should be \(r\) or more, then it is the weighted version of the \((k, r)\)-gathering problem. Unfortunately it is NP-complete even for \(|F| = 2\), as follows. Given a set \(S\) of integers, problem PARTITION asks for the existence of a subset \(S'\) with \(\sum_{a \in S'} a = \sum_{a \notin S'} a\). PARTITION is NP-complete [2]. We can transform PARTITION to the weighted version of the \((k, r)\)-gathering problem as follows. Let \(S = \{a_1, a_2, \ldots, a_n\}\). Locate each customer with weight \(a_i\) at \(i\) for each \(i = 1, 2, \ldots, n\). Set \(F = \{1, n\}\), \(k = n\) and \(r = (\sum_{i=1}^{n} a_i)/2\). Now a \((k, r)\)-gathering exists if and only if PARTITION has a solution.

If customers can gather at any place, not restricted at potential facility locations, then one can perturbate any solution of the problem so that every gathering place is at \(c_i + k\) for some \(c_i\). So, by locating possible facilities at \(c_i + k\) for each \(c_i \in C\), we can find a solution of this problem using our algorithm. The running time is \(O(|C| + |F|) = O(|C| + |C|) = O(|C|)\).

References
