Shooter Location through Piercing Sets

Matthew J. Katz[1*] Frank Nielsen[2] Michael Segal[3†]

[1]Department of Mathematics and Computer Science
Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel
[2]SONY Computer Science Laboratories Inc., FRL
3-14-13 Higashi Gotanda, Shinagawa-Ku, Tokyo 141-0022, Japan
[3]Department of Computer Science
University of British Columbia, Vancouver, B.C. V6T 1Z4, Canada

Abstract

We present efficient approximate and exact solutions to the shooter location problem, using cuttings and a new data structure for maintaining a minimum (alternatively, a nearly minimum) piercing set for a set $\mathcal{S}$ of intervals on the line.

1 Introduction

The shooter location problem. Given a set of n disjoint segments in the plane, find a location p in the plane, for which the number of shots needed to hit all segments is minimal, where a shot is a ray emanating from p. This problem was first introduced by Nandy et al. [5], who observed that solving the problem for a given location p is equivalent to finding a minimum piercing set for a set of n arcs on a circle. The latter problem can be solved in time $O(n \log n)$ (see, e.g., [3, 4]). They also presented an $O(n^3)$-time algorithm for the case where the shooter is allowed to move along a given line, and left open the general problem. Wang and Zhu [6] obtained an $O(n^5 \log n)$-time solution for the general problem. They also gave an $O(n^5)$-time algorithm for computing a 2-approximation, that is, a location for which the number of required shots is at most twice the optimal number of shots. Recently, Chaudhuri and Nandy [1] presented an improved solution for the general problem; its worst-case running time is $O(n^5)$, but it is expected to perform better in practice. Actually, the general problem can be solved by applying the solution for a shooter on a line to the $O(n^2)$ lines defined by the endpoints of the segments, thus obtaining an alternative $O(n^5)$-time solution.

Here we obtain an $O(\frac{1}{\varepsilon} n^4 \log n)$-time algorithm for computing a $(1+\varepsilon)$-approximation for the general problem (with possibly one extra shot), significantly improving the $O(n^5)$ 2-approximation of [6]. We can also find a location for which the number of shots is at most r^*+1, where r^* is the optimal number of shots, in (output-sensitive) $O(n^4 r^* \log n)$ time. Finally, we describe another, more complicated, method for computing a $(1+\varepsilon)$-approximation. This method uses cuttings to

*Supported by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities, and by an Intel research grant.

†Supported by the Pacific Institute for Mathematical Studies and by the NSERC research grant.

compute a $(1+\varepsilon)$-approximation in $O(\frac{1}{\varepsilon^3}\frac{n^4\log n}{r^*})$ time, thus breaking the $O(n^4)$ barrier for most values of r^*, assuming ε is a constant.

All our solutions use a new data structure for a set $\mathcal{S}$ of intervals on the line [3]. The data structure, which is of size $O(|S|)$, enables us to either (i) maintain a minimum piercing set for $\mathcal{S}$ in $O(c(\mathcal{S})\log|\mathcal{S}|)$ time per update, where $c(\mathcal{S})$ is the size of the current piercing set, or (ii) maintain a piercing set for $\mathcal{S}$ which is of size $(1+\varepsilon)c(\mathcal{S})$, for $0<\varepsilon\le 1$, in $O(\frac{\log|\mathcal{S}|}{\varepsilon})$ amortized time per update.

1.1 Algorithms

In the *Shooter Location Problem* (SLP for short), we are given a set $\mathcal{S}=\{s_1,\ldots,s_n\}$ of n disjoint segments in the plane, and we seek a point p from which the number of shots needed to hit all segments in $\mathcal{S}$ is minimal, where a shot is a ray emanating from p.

A $(1+\varepsilon)$-approximation. Let $\mathcal{L}$ be the set of $O(n^2)$ lines defined by the endpoints of the segments in $\mathcal{S}$. Consider any cell f of the arrangement $\mathcal{A}(\mathcal{L})$, and let p be a point in the interior of f. The number of shots from p needed to hit all segments in $\mathcal{S}$ is equal to the size of a minimum piercing set for the set of circular arcs obtained by projecting each of the segments in $\mathcal{S}$ on a circle enclosing all the segments in $\mathcal{S}$ and centered at p. For any other point p' in the interior of f, the number of shots from p' is equal to the number of shots from p, since the circular-arc graphs for p and for p' are identical. Moving from one cell of $\mathcal{A}(\mathcal{L})$ to an adjacent cell corresponds to a swap in the locations of two adjacent arc endpoints.

We traverse the arrangement $\mathcal{A}(\mathcal{L})$, dynamically maintaining an approximation of r^*, the minimum number of rays required to intersect all the segments. At each cell of $\mathcal{A}(\mathcal{L})$, we shoot a vertical ray directed upwards, allowing us to deal with the interval graph obtained by unrolling the cell's circular-arc graph (after removing the arcs that are intersected by the vertical ray). We use the recent data structure of Katz et al. [3] (mentioned above) to maintain in amortized time $O(\frac{\log n}{\varepsilon})$ a $(1+\varepsilon)$-approximation of the minimum piercing set for this interval graph. At the end, we choose the cell for which the number computed is the smallest. (Actually, this scheme will also work for segments that are not necessarily disjoint.)

Theorem 1 *For any fixed $0<\varepsilon\le 1$, a $(1+\varepsilon)$-approximation (with possibly one extra shot) for the shooter location problem can be found in $O(\frac{1}{\varepsilon}n^4\log n)$ time.*

Towards an exact solution. We showed how to obtain a $(1+\varepsilon)$-approximation, that is, how to find a number r such that $r^*\le r\le(1+\varepsilon)r^*+1$. Therefore, if $\varepsilon r^*<1$, we obtain a location for which the number of rays is either optimal or optimal plus one. Since we need to choose $\varepsilon<\frac{1}{r^*}$ without knowing r^*, we first run the algorithm with, say, $\varepsilon=1$, and obtain a number of rays $r^*\le r'\le 2r^*+1$. Then we choose $\varepsilon=\frac{1}{r'}<\frac{1}{r^*}$ and run the algorithm to obtain the optimal, or optimal with one extra shot, solution in $O(n^4r^*\log n)$ output-sensitive time (single bootstrapping).

Theorem 2 *The optimal number of shots r^* (with possibly one extra shot) of the shooter location problem can be computed in $O(n^4r^*\log n)$-time.*

Avoiding the complete arrangement traversal. We now describe another method for obtaining a $(1+\varepsilon)$-approximation, which is often more efficient than the method described above. Let $\mathcal{L}'$ be a $\frac{1}{c}$-cutting of $\mathcal{L}$ (see [2]). That is, $\mathcal{L}'$ is a set of $O(c)$ lines, and each cell of the (vertical decomposition of the) arrangement $\mathcal{A}(\mathcal{L}')$ is cut by at most $\frac{|\mathcal{L}|}{c} \leq \frac{2n^2}{c}$ lines of $\mathcal{L}$. For each of these lines l, using the data structure of [3] we dynamically maintain a $(1+\delta)$-approximation for a shooter moving along l (in the original environment S). The total computation time is $O(n^2 c \frac{\log n}{\delta})$. Let r_{min} be the best score obtained during the computation. We have $r^* \leq r_{min} \leq (1+\delta)(r^* + \frac{2n^2}{c})$. (The right inequality holds since if $C \in \mathcal{A}(\mathcal{L})$ is the cell from which only r^* shots are needed, then there exists a cell $C' \in \mathcal{A}(\mathcal{L})$ that is supported by a line in $\mathcal{L}'$, such that, C can be reached from C' by passing through at most $\frac{2n^2}{c}$ cells of $\mathcal{A}(\mathcal{L})$.) By setting $c = \frac{2n^2}{\gamma r^*}$, for some $0 < \gamma < 1$, we obtain $r^* \leq r_{min} \leq (1+\delta)(1+\gamma)r^*$ in $O(\frac{1}{\delta\gamma}\frac{n^4 \log n}{r^*})$ time. We choose $\delta = \gamma = \frac{\varepsilon}{3}$ to ensure a $(1+\varepsilon)$-approximation scheme in $O(\frac{1}{\varepsilon^2}\frac{n^4 \log n}{r^*})$ time.

However, we do not know r^*, the size of the optimal solution, beforehand. We are going to approximate it by r' as follows. We first demonstrate the method for the special case where a 4-approximation is desired, and then present it for the general case.

For a 4-approximation, assume $\delta = \gamma = \frac{1}{4}$, and set $r' \leftarrow \frac{n}{2}$. Let $c = \frac{2n^2}{\gamma r'} = 16n$, and, as above, first compute a $\frac{1}{c}$-cutting of $\mathcal{L}$ and then, for each of the $O(c)$ lines in the cutting, compute a $(1+\delta)$-approximation for a shooter moving along the line. The total computation time is $O(n^3 \log n)$. By taking the minimum score r_{min} along the lines of the cutting, we have $r^* \leq r_{min} \leq (1+\frac{1}{4})(r^* + \frac{n}{8})$. Therefore, if $r_{min} \geq \frac{n}{2}$, then $r^* \geq \frac{11n}{40} \geq \frac{n}{4}$ and we return r_{min} and stop. This gives $\frac{r_{min}}{r^*} \leq \frac{n}{n/4} = 4$. Otherwise, we set $r' \leftarrow \frac{r'}{2}$ and repeat. We continue halving r' until at some stage $r_{min} \geq r'$ (and $r_{min} < 2r'$). At this stage we have $r^* \geq \frac{11r'}{20} \geq \frac{r'}{2}$, and $\frac{r_{min}}{r^*} \leq \frac{2r'}{r'/2} = 4$. The overall cost of this algorithm is bounded by $O(n^4 \log n) \sum_{r'} \frac{1}{r'}$ with $r' = \frac{n}{2^i}$ for $i \leq \log \frac{n}{r^*}$. Thus we end up with a 4-approximation in $O(\frac{n^4 \log n}{r^*})$ time.

For the general case, where a $(1+\varepsilon)$-approximation is desired, we set $r' = \beta n$, for an appropriate $0 < \beta < 1$, as our current estimate of r^*, and let $c = \frac{2n^2}{\gamma r'} = \frac{2n^2}{\gamma \beta n}$. After computing a $\frac{1}{c}$-cutting and r_{min} as before, we have

$$r^* \leq r_{min} \leq (1+\delta)(r^* + \frac{2n^2}{c}) = (1+\delta)(r^* + \gamma\beta n) \ .$$

Now, if $r_{min} \geq r'$ (i.e., if $r_{min} \geq \beta n$), then $(1+\delta)(r^* + \gamma\beta n) \geq \beta n$, which implies that

$$r^* \geq \frac{\beta n(1-(1+\delta)\gamma)}{1+\delta} \ .$$

Thus,

$$\frac{r_{min}}{r^*} \leq \frac{1+\delta}{\beta(1-(1+\delta)\gamma)} \ , \qquad (*)$$

since $r_{min} \leq n$.

If, however, $r_{min} < r'$, we set $r' \leftarrow \beta r'$, and repeat until at some stage $r_{min} \geq r'$. At this stage we have $\beta^i n \leq r_{min} < \beta^{i-1} n$, for some $i \geq 2$, and the ratio between r_{min} and r^* is as in the first stage (Equation $(*)$), this time using $r_{min} < \beta^{i-1} n$.

Therefore, we must pick δ, γ and β such that

$$\frac{1+\delta}{\beta(1-(1+\delta)\gamma)} \leq 1+\varepsilon \ . \qquad (**)$$

The running time is

$$\frac{n^4 \log n}{\gamma\delta} \Sigma_i \frac{1}{\beta^i n} \ ,$$

with i ranging from 1 to $\log_{\frac{1}{\beta}} \frac{n}{r^*}$. That is,

$$\frac{n^3 \log n}{\gamma\delta} \sum_{i=1}^{\log_{\frac{1}{\beta}} \frac{n}{r^*}} (\frac{1}{\beta})^i \ .$$

But $\sum_{i=1}^{\log_{\frac{1}{\beta}} \frac{n}{r^*}} (\frac{1}{\beta})^i$ is less than $\frac{n}{(1-\beta)r^*}$. Therefore the running time for a $(1+\varepsilon)$-approximation is $O(\frac{n^4 \log n}{\delta\gamma(1-\beta)r^*})$. It is easy to verify that by picking $\gamma = \delta = \frac{\varepsilon}{5}$ and $\beta = 1 - \frac{\varepsilon}{5}$, Equation $(**)$ is satisfied (assuming $\varepsilon \leq 1$), and thus the running time becomes $O(\frac{1}{\varepsilon^3} \frac{n^4 \log n}{r^*})$.

Comparing this method with the first method, we see that this method is more efficient than the first method whenever $r^* \geq \frac{1}{\varepsilon^2}$.

Theorem 3 *A $(1+\varepsilon)$-approximation for the shooter location problem can be found in $O(\frac{1}{\varepsilon^3} \frac{n^4 \log n}{r^*})$ time.*

References

[1] J. Chaudhri and S.C. Nandy "Generalized shooter location problem", in *Lecture Notes in Computer Science* 1627, pp. 389–401, 1999.

[2] B. Chazelle "Cutting hyperplanes for divide-and-conquer", *Discrete Comput. Geom.* 9 (1993), pp. 145–158.

[3] M. J. Katz, F. Nielsen and M. Segal "Maintenance of piercing sets with applications", manuscript.

[4] D. T. Lee, M. Sarrafzadeh and Y. F. Wu "Minimum cuts for circular-arc graphs", *SIAM J. Computing* 19(6) (1990), pp. 1041–1050.

[5] S. C. Nandy and K. Mukhopadhyaya and B. B. Bhattacharya "Shooter location problem", in *Proc. 8th Canad. Conf. Comput. Geom.*, pp. 93–98, 1996.

[6] C. A. Wang and B. Zhu "Shooter location problems revisited", in *Proc. 9th Canad. Conf. Comput. Geom.*, pp. 223–228, 1997.