Computing the maximum overlap of a disk and a piecewise circular domain under translation

Narcís Coll* Marta Fort* J. Antoni Sellarès*

Abstract

We present a GPU parallel algorithm for approximately computing the maximum overlap of a disk and a piecewise circular domain under translation. We also provide initial experimental results obtained with the implementation of our algorithm.

1 Introduction

The continuous maximal coverage problem consists in siting facilities in the continuous space to maximize coverage of regional demand. We study the one-facility case, with the assumption of uniformly distributed demand and a disk-like service area for the facility (see Figure 1 for a motivational example).

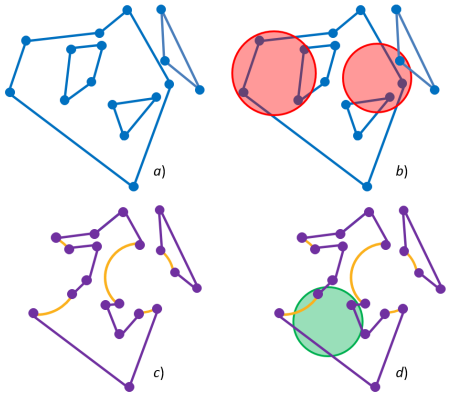

Figure 1: a) Polygonal domain to be partially covered by circular sensors; b) Domain partially covered by two circular sensors; c) Piecewise circular domain not yet covered; d) New sensor partially covering the piecewise circular domain.

A piecewise circular curve is a finite ordered list of connected circular arcs and line segments (considered as circular arcs with infinite radius). The arcs and line segments are the edges, and the points where these edges intersect are the vertices of the piecewise circular curve. A piecewise circular curve is closed if its first and last vertices coincide, and weakly simple if some pair of non-adjacent edges may intersect but the edges do not cross. A piecewise circular region is a set whose boundary is a closed weakly simple piecewise circular curve. A piecewise circular region with holes is a piecewise circular region from which the union of the interiors of a finite number of enclosed piecewise circular regions, which define the holes, has been removed. The boundaries of the enclosing piecewise circular region and the holes are pairwise disjoint, and the holes are empty. A piecewise circular domain is the union of a finite collection of non overlapping piecewise circular regions with holes.

Next, we formally define the problem to be solved. Let P be a given piecewise circular domain. For any point $q \in \mathbb{R}^2$, denote $D_r(q)$ the disk of center q and radius r. The goal is to find a location $q_0 \in \mathbb{R}^2$ which maximizes the area $A(q)$ of the overlap of $D_r(q)$ with P.

There has been some related work on this problem. Given two simple polygons P and Q with n and m vertices, respectively, Mount et al. [4] gave an algorithm to compute their maximum overlap under translation in $O(n^2m^2)$ time. Cheong et al. [1] proposed an algorithm to approximate the maximum overlap using random sampling techniques. With high probability the additive error is $\varepsilon \cdot \min\{\text{area}(P), \text{area}(Q)\}$ and the running time is $O(n + (m^2\varepsilon^{-4}\log_2 m))$. More recently, Cheng and Lam [2] presented an algorithm to approximate the maximum overlap of two polygons P and Q, built upon the framework of Cheong et al. [1]. Polygons P and Q may have multiple holes. If n denotes the total number of vertices in P and Q, the running time of the algorithm is $O(n^2\varepsilon^{-3}\log^{1.5} n \log(n/\varepsilon))$. If one of the two polygons is convex, the additive error with high probability is $\varepsilon \cdot \text{area}(P)$ and the running time can be improved to $O(n\log n + \varepsilon^{-3}\log^{2.5} n \log((\log n)/\varepsilon))$.

The prohibitive running times of the existing approximation algorithms, mainly for small ε values, motivated us to design a GPU parallel approach to efficiently find a set of approximate solutions. An initial version of this paper dealing with polygonal domains was presented in [3].

2 Overlap area computation

To solve the problem we need an efficient way to compute the area of the overlap between a disk and a piecewise circular domain. It can be computed as the area of the overlap between the disk and the outer components of the domain minus the area of overlap between the disk and each one of the holes. Along this section we provide a way to exactly and efficiently compute the area $A(q) = \text{area}(D_r(q) \bigcap R)$ of the over-

*Departament d'Informàtica, Matemàtica Aplicada i Estadística. Universitat de Girona, Spain, {coll,mfort,sellares}@ima.udg.edu.

lap of $D_r(q)$ with a piecewise circular region R without holes. This area $A(q)$ can be computed in time proportional to the number n of the vertices of R as follows. The area $A(q)$ is equal to the area of overlap between $D_r(O)$ and R', where O represents the origin and R' is the region R translated by the vector $-\overrightarrow{Oq}$. Taking into account that a circular arc or a segment intersects a circle in at most two points, the boundary of R' can be expressed as a closed piecewise curve $B = \bigcup_{i=0}^{m-1} B_i$ ($m \leq 3n$). Each curve B_i connects the points p'_i and p'_{i+1} which are vertices of R' or intersection points between $\partial D_r(O)$ and the boundary of R'. Moreover, each B_i satisfies one of the next cases:

Case 1: B_i is a piecewise circular curve exterior to $D_r(O)$ and its endpoints p'_i and p'_{i+1} are intersection points.

Case 2: B_i is a segment contained in $D_r(O)$.

Case 3: B_i is a circular arc contained in $D_r(O)$.

Consider now the curve $\overline{B} = \bigcup_{i=0}^{m-1} \overline{B}_i$ where $\overline{B}_i$ is the radial projection of B_i onto $D_r(O)$ when B_i is exterior to $D_r(O)$ and B_i otherwise. Observe that the polycurve $\overline{B}$: 1) is weakly simple; 2) can be continuously approximated by simple closed curves (see Figure 2); 3) encloses a region whose area equals $A(q)$.

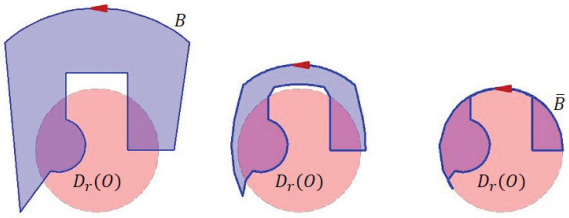

Figure 2: Approximation of $\overline{B}$ from B.

Thus, the area $A(q)$ can be computed by using Greens's theorem as follows:

$$A(q) = \frac{1}{2}\int_{\overline{B}} -ydx + xdy = \frac{1}{2}\sum_{i=0}^{m-1} I_i\,,$$

where $I_i = \int_{\overline{B}_i} -ydx + xdy$. Next, we explain how to compute the value of I_i according to the case where B_i belongs to:

Case 1: B_i is a piecewise circular curve exterior to $D_r(O)$. Let A_i be the shortest oriented arc on $D_r(O)$ that connects the point p'_{i+1} with the point p'_i and let $B_{i,i}$ be the closed simple curve determined by $B_i \cup A_i$. Then, I_i can be computed by:

$$I_i = I_{i,i} - \int_{A_i} -ydx + xdy = I_{i,i} - r^2\alpha_i\,,$$

where
$$I_{i,i} = \int_{\overline{B}_{i,i}} -ydx + xdy\,,$$

and α_i denotes the oriented angle between the vectors $\overrightarrow{Op'_{i+1}}$ and $\overrightarrow{Op'_i}$.

The result of the integral $I_{i,i}$ depends on the orientation of $B_{i,i}$ and on whether the disk $D_r(O)$ is interior or exterior to $B_{i,i}$. Let n_i be the number of intersections between B_i and the half-line with origin O in the direction of the vector $-\overrightarrow{Op'_i}$. According to n_i and α_i, there are three cases to consider (Figure 3):

1. n_i is odd and $\alpha_i > 0$. Then, $D_r(O)$ is interior to $B_{i,i}$ and $B_{i,i}$ is oriented counterclockwise. Consequently, $I_{i,i} = -2\pi r^2$ and $I_i = r^2(2\pi - \alpha_i)$.
2. n_i is odd and $\alpha_i < 0$. Then, $D_r(O)$ is interior to $B_{i,i}$ and $B_{i,i}$ is oriented clockwise. Consequently, $I_{i,i} = -2\pi r^2$ and $I_i = -r^2(2\pi + \alpha_i)$.
3. n_i is even. Then, $D_r(O)$ is exterior to $B_{i,i}$. Consequently, $I_{i,i} = 0$ and $I_i = -r^2\alpha_i$.

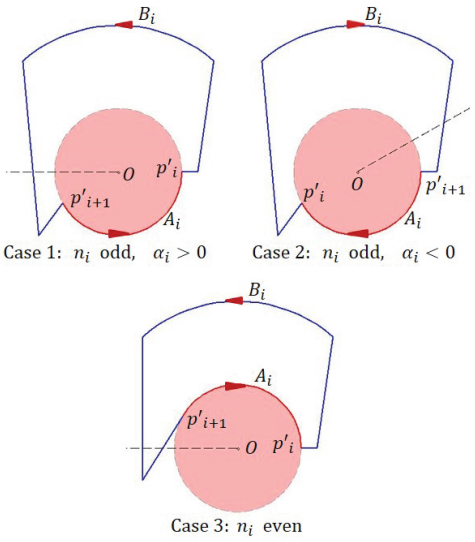

Figure 3: Exterior chain cases.

Case 2: B_i is a segment contained in $D_r(O)$. The segment $\overline{B}_i = B_i$ can be parameterized by:

$$p'_i + t(p'_{i+1} - p'_i), \quad t \in [0,1]\,.$$

Then, it holds:

$$I_i = \det(p'_i, p'_{i+1})\,.$$

Case 3: B_i is a circular arc contained in $D_r(O)$. The arc $\overline{B}_i = B_i$ can be parameterized by:

$$c_i + \cos(t)\overrightarrow{c_ip'_i} + \sin(t)\overrightarrow{c_ip'_i}^{\perp}, \quad t \in [0, \beta_i]\,,$$

where c_i is the center of the arc B_i and β_i is the oriented angle between the vectors $\overrightarrow{c_ip'_i}$ and $\overrightarrow{c_ip'_{i+1}}$. Then, it holds:

$$I_i = \det(c_i, p'_{i+1} - p'_i) + r^2\beta_i\,.$$

3 Computing the maximum overlap

In this section we describe our strategy to approximately obtain the maximum overlap of a disk and a

piecewise circular domain under translation with an ϵ-absolute error. In subsection 3.1, we present a basic approach that samples the minimum bounding box of the piece circular domain by using a uniform grid. We improve the approach locally refining the grid in subsection 3.2.

3.1 Uniform grid solution

Lemma 1 *At least a location optimizing $A(q)$ belongs to the region delimited by the convex hull $CH(P)$ of the piecewise circular domain P and, consequently, it also belongs to the minimum bounding box of P.*

The proof of Lemma 1 can be found in [3].

Lemma 1 allows us to reduce the search space for finding the maximum area of overlap. First, we sample the minimum bounding box of the piecewise circular domain P by using a uniform grid composed of square cells of side length d. We choose $d \leq \sqrt{2}r$, so that for each grid cell center c the disk $D_r(c)$ covers the cell of center c. Then, we compute the area $A(c)$ for each grid cell center c and pick a center c_0 with maximum area. To ensure that the absolute error between areas when choosing as optimal grid center c_0 instead of the optimal point q_0 is smaller than ε, we need to choose an appropriate side length d smaller than a threshold value d_ε. Let us determine d_ε.

The largest value of the absolute error $A(q_0)-A(c_0)$ occurs when the point q_0 coincides with a vertex of a square grid cell. It is bounded by the area of the lune $L(q_0) = D_r(q_0) \setminus D_r(c)$, where c is an arbitrary grid cell center, because the lune $L(q_0)$ is a subregion of $D_r(q_0) \bigcap P$ but the lune $L(c) = D_r(c) \setminus D_r(q_0)$ does not intersect $D_r(q_0) \bigcap P$ (see Figure 4a). Thus:

$$A(q_0) - A(c_0) \leq A(q_0) - A(c) \leq \text{area}(L(q_0)) \,. \quad (1)$$

In the particular worst-case in which q_0 is a vertex of the square grid cell of center c_0, taking $h = d(c_0, q_0) = d/\sqrt{(2)} \leq r$ and by using Taylor expansion, we have:

$$\text{area}(L(q_0)) = 2r^2 \arcsin\left(\frac{h}{2r}\right) + \frac{h}{2}\sqrt{4r^2 - h^2} \leq$$
$$\leq 2rh = \sqrt{2}dr \,.$$

Fixed r and $\varepsilon \in (0, 1]$, and according to (1), an ε absolute error can be guaranteed by choosing the side length d_ε of a square grid cell as:

$$d_\varepsilon = \min\left(\sqrt{2}r, \frac{\varepsilon}{\sqrt{2}r}\right) ,$$

because, in such a case:

$$A(q_0) - A(c_0) \leq \text{area}(L(q_0)) \leq \sqrt{2}d_\varepsilon r = \varepsilon \,.$$

Thus, if the bounding box of the polygon P has dimension $a \times b$, the number of vertices of the regular grid providing an ε absolute error is $(\lceil a/d_\varepsilon \rceil + 1)(\lceil b/d_\varepsilon \rceil + 1) \in O(ab(r/\varepsilon)^2)$.

3.2 Adaptive local grid refinement

If we use an initial grid of size d_ϵ over the entire bounding box of P we may waste a lot of grid cells in areas where they are not necessary. It may be computationally demanding in time and memory requirements. Using a global refinement method would have the same problems. Thus, we propose a local grid refinement method starting with a coarser grid of side length d with $\sqrt{2}r \geq d \geq d_\varepsilon$. The local grid refinement strategy identifies the cells, called parent cells, to be refined according to a two-way filtering criterium that allows us to quickly detect grid cells where it is not necessary to apply the refinement process because their points can not be optimal, Lemma 2, or are all optimal, Lemma 3. After detecting the parent cells, we determine a new smaller value for d, according to the desired ε or the maximum number of desired grid cells used per refinement step. We construct a new regular grid of child cells on each selected parent cell and we perform, for each child cell, the process we followed for the initial grid. This local refinement process can be repeated as many times as required until $d \leq d_\varepsilon$ and the desired accuracy is obtained.

Next, we give the mentioned Lemmas:

Lemma 2 *If $d \leq 2\sqrt{2}r$ and $A(c_0) - A(c) > \sqrt{2}dr$ for the center c of a grid cell g, then the optimal point q_0 does not belong to the cell g.*

Proof. From (1) we know that if the cell g of center c contains q_0 the following inequalities are fulfilled:

$$A(q_0) - A(c) \leq \text{area}(L(q_0)) \leq \sqrt{2}dr \,. \quad (2)$$

If $A(c_0) - A(c) > \sqrt{2}dr$, then also

$$A(q_0) - A(c) \geq A(c_0) - A(c) > \sqrt{2}dr \,, \quad (3)$$

thus q_0 cannot belong to g because in this case inequalities (2) and (3) are in contradiction. □

Lemma 3 *Assume that $A(c) = area(D_r(c) \cap P) = area(D_r(c)) = \pi r^2$, thus $D_r(c) \subseteq P$ and c is an optimal point. If for the center c' of each one of the eight cells adjacent to cell g we have $A(c') = \pi r^2$, then any point of the cell g is optimal.*

Proof. The union F of the set of all disks of radius r whose center belongs to the cell g is the r-offset of the cell g. Consequently, if $F \subseteq P$, then, $D_r(c') \subseteq P$ for any point c' belonging to the cell g, thus c' is an optimal point. Observe that the union of the eight disks of radius r centered in the center of the grid cells adjacent to cell g together with the disk $D_r(c)$ covers the r-offset F (see Figure 4b)). Thus, in order to have $F \subseteq P$ it suffices that $D_r(c') \subseteq P$, or equivalently $D_r(c') = \pi r^2$, for the center c' of each of the eight adjacent cells to g. □

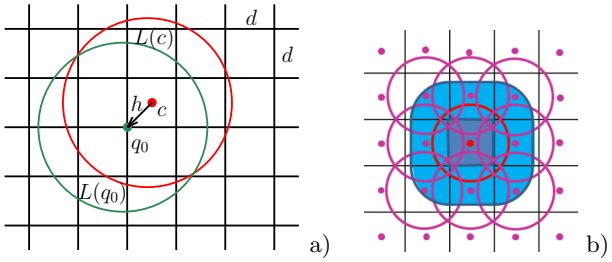

Figure 4: a) The maximum error occurs when the optimal point q_0 is a vertex of a cell. b) The union of the nine disks covers the cell offset.

4 GPU Implementation

To solve the problem in the GPU we have to transfer the piecewise circular domain to the GPU. We use: a float array storing the vertex coordinates; three integer values with the number of vertices, of enclosed components and of holes; an integer array p in which $p[i] = -1$ if the edge from vertex v_i to vertex v_{i+1} is a line segment and $p[i] = j \geq 0$ otherwise; and two extra float arrays with the radii and centers of the circular arcs, their jth element is the radius and center of the edge from v_i to v_{i+1} whenever $p[i] = j \geq 0$.

The initial overlap areas, corresponding to the circles centered at a $a \times b$ grid with $d \leq \sqrt{2}r$, are computed in parallel by finding one area per thread, the current $A(c_0)$ value is estimated using an atomic max operation. The threads in a block cooperate to store the polygon vertices information to shared memory. If a refinement step is required, we analyze the cells in parallel using the filtering criteria, parent cells are marked with a 1 and terminal cells with a 2 or a 0 depending on whether they are or cannot be optimal. It is done by using squared $B \times B$ blocks, whose threads cooperate to transfer the $(B+2) \times (B+2)$ potentially required areas to shared memory. After being marked, the N parent cells are extracted and a $k \times k$ new grid is placed in each one. We take $k = \min(32, a, b, k_\varepsilon)$ in the first step, where the value 32 is due to GPU reasons and $k_\varepsilon = d/d_\varepsilon$. In the subsequent steps $k = \min(32, k, k_\varepsilon)$. Note that at each refinement step d is divided by k becoming d/k. We add two extra rows and columns surrounding the $k \times k$ grid, that will never be refined, to be able to properly use the stop-refining criterium of Lemma 3 at this refinement step. Thus, we compute $(k+2) \times (k+2)$ areas per parent cell. Finally, the $N(k+2)^2$ areas are computed in parallel and $k \times k$ blocks are used in the next filtering step, if it is required.

5 Experimental results

We have implemented our algorithms in C++ and Cuda C and run the experiments using a Inter(R) Core(TM) i7-4790CPU with a Tesla k40 active GPU.

We have used the piece-wise circular polygon with holes of 3569 edges and considered the orange circle of $r = 3.6$ (km) that are shown in Figure 5.

We can see, represented by colored points, the cell centers analyzed during the process. The green ones are the centers of the cells having maximum area, in this case $\pi r^2 \approx 41$ (km^2). The blue points are the other analyzed centers, they are painted in a blue color gradation according to the overlap area of the circle centered on them. The darker the point the smaller the overlap area. The gradual refinement is clearly seen in the figure as well as some rounding errors in the points surrounding the regions with maximal area.

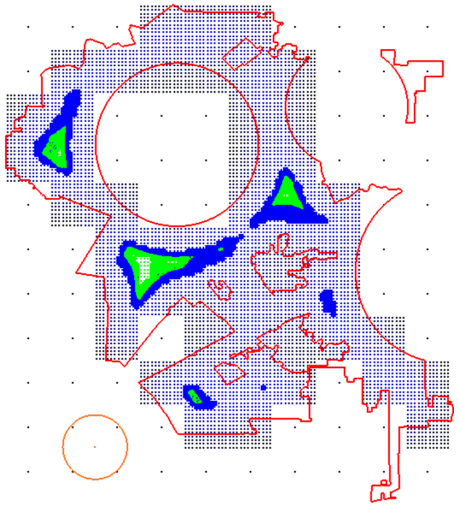

Figure 5: Piecewise circular domain with the analyzed grid cells centers, the optimal locations are marked in green. Initial grid size=11 × 10.

Considering $\varepsilon = 0.144$ (km^2) and a 11×10 initial grid, which corresponds to an initial grid cell size of $d = 4.9$ (km), three refinement steps are needed until a grid with cell size of 17 (m) is obtained, the algorithm takes 0.14 (s). By using a finer 352×285 initial grid, which has $d = 211$ (m), one refinement step and 0.19 (s) are needed, now the smallest d is of 26 (m).

Acknowledgments

Work partially funded by the TIN2014-52211-C2-2-R project from MEC, Spain. We also acknowledge NVIDIA Corporation for the donation of the Tesla K40 GPU.

References

[1] O. Cheong, A. Efrat, S. Har-Peled, Finding a guard that sees most and a shop that sells most, *Discrete & Computational Geometry* **37(4)** (2007) 545-563.

[2] S.W. Cheng, C.K. Lam, Shape matching under rigid motion, *Comput. Geom. Theory Appl.* **46(6)** (2013) 591–603.

[3] N. Coll, M. Fort, J.A. Sellarès, Computing the maximum overlap of a disk and a polygon with holes under translation, *XVI Encuentros de Geometría Computacional* (2015) 57–60.

[4] D.M. Mount, R. Silverman, A.Y. Wu, On the area of overlap of translated polygons, *Computer Vision and Image Understanding* **64(1)** (1996) 53–61.