GPU-BASED INFLUENCE REGIONS OPTIMIZATION

Marta Fort
J.Antoni Sellares
Objective

Given a set of restaurants, open a new restaurant so that it takes over as many customers as possible from the existing competitors.
Competitive facility location problem

Motivation – Example

Objective

Given a set of restaurants, open a new restaurant so that it takes over as many customers as possible from the existing competitors.

Assumptions

Clients are not equally distributed (domain partition)
Competitive facility location problem

Motivation – Example

Objective

Given a set of restaurants, open a new restaurant so that it takes over as many customers as possible from the existing competitors.

Assumptions

Clients are not equally distributed (domain partition)
Clients eat at one of their 4-nearest restaurants
Competitive facility location problem

Motivation – Example

Objective

Given a set of restaurants, open a new restaurant so that it takes over as many customers as possible from the existing competitors.

Assumptions

Clients are not equally distributed (domain partition)
Clients eat at one of their 4-nearest restaurants

Solution

We locate the new restaurant so that the number of people living/working in the region having the new restaurant among their 4-nearest restaurants is maximum.
Competitive facility location problem

Motivation – Another Example

Given a set of garbage dumps, locate a new garbage plant so that the number of people living/working in the region having the plant among their 3-farthest garbage dumps is maximum.
**Nearest influence region - \( NI_k(s, S) \)**

Set of points of the domain \( D \) having the given facility among their \( k \)-nearest neighbors.

- It is a **star shape polygon** having the facility \( s \) in its kernel.
- \( NI_k(s, S) \subset NI_{k+1}(s, S) \).
Farthest influence region - $F_{I_k}(s, S)$

Set of points of the domain $D$ having the given facility among their $k$-farthest neighbors

- $F_{I_k}(s, S)$ and $N_{I_{n-k}}(s, S)$ are complementary with respect the domain.
- $F_{I_{k+1}}(s, S) \subset F_{I_k}(s, S)$. 

$|S| = 11$
Weighed area of a region

Given a weighted polygonal partition of the domain $D$, the weighed area of a region $R$, $w(R)$, is the sum of the weighted areas of the subregions obtained intersecting $R$ with the polygons of the partition of $D$. 
Problem definition

From now on, we denote by $I_k(s, S)$ an order-$k$ influence region, without specifying the nearest or farthest criterion.

**Competitive facility location problem**

Given a set of existent facilities $S$ and a weighted partition of the domain $D$, we want to find the location of a new facility $s$ in $D - S$ whose influence region $I_k(s, S)$ has maximal weight.
Previous work

• Maximizing the Voronoi region of a new facility (case k=1)
  - Pixel based solutions (Denny) [3]
  - Newton approximation (Dehne et al.) [2]
  - Near-linear time (1+ε)-approximated algorithm (Cheong et al.) [1]


Competitive facility location problem

The **difficulty of maximizing**, analytically or by a numerical optimization procedure, the **weighted area of the $k$-influence region**, has motivated us to use a GPU parallel approach.
GPU – Graphics Processing Units

The programmable **Graphic Processor Unit** or **GPU** is especially well-suited to address problems for which the same program is executed on many data elements in parallel.
GPU – Graphics Processing Units

All the threads execute the instructions contained in the kernel in parallel.
Atomic instructions

Meanwhile a thread reads, modifies and writes to a location any other thread can access there.
GPU Solution

To find the point \( s \in D - S \) maximizing \( w(l_k(s, S \cup \{s\}) \) we proceed as follows:

1. **Discretize** \( D \) into a grid of size \( G=H\times W \) obtaining the set \( Q \) of candidate points.

2. For each \( q \in Q \) estimate \( w(l_k(q, S \cup \{q\}) \) as

   \[
   \sum_{q' \in Q \cap l_k(q, S \cup \{q\})} w(q')
   \]

3. Find the point \( r \in Q \) whose estimated \( w(l_k(r, S \cup \{r\}) \) is maximum
GPU Solution

To find the point $s \in D - S$ maximizing $w(I_k(s, S \cup \{s\})$ we proceed as follows:

1. Discretize $D$ into a grid of size $G=HxW$ obtaining the set $Q$ of candidate points.

2. For each $q \in Q$ estimate $w(I_k(q, S \cup \{q\})$ as

   $$\sum w(q') \text{ for each } q' \in Q \cap I_k(q, S \cup \{q\})$$

3. Find the point $r \in Q$ whose estimated $w(I_k(r, S \cup \{r\})$ is maximum

Using parallel algorithms in the GPU
Approximated solution
Estimated \( w(l_k(q, S \cup \{q\})) \) computation

Store S.
Discretize $D$ into a grid of size $G=H \times W$, obtaining the set $Q$ of candidate points.

**Estimated $w(I_k(q, S \cup \{q\})$ computation**
Estimated \( w(I_k(q, S \cup \{q\}) \) computation

Store the weights of the points of \( Q \) according to the weighted partition of the domain.
Estimated $w(I_k(q, S \cup \{q\})$ computation

Fix $k$. 
For each grid point \( q \), run a thread that computes and stores the distance from \( q \) to its \( k \)-nearest point in \( S \).
For each grid point $q'$, run a thread that reads its weight and $k$-nearest distance.
Estimated \( w(I_k(q, S \cup \{q\})) \) computation

For each grid point \( q' \), run a thread that reads its weight and \( k \)-nearest distance, next explores the grid points \( q \) whose influence region \( I_k(p, S \cup \{q\}) \) contains \( q' \) (in the nearest / farthest case, those such that: \( d(q', q) \leq d_k(q', S) \) / \( d(q', q) \geq d_k(q', S) \))
Estimated $w(I_k(q, S \cup \{q\})$ computation

For each grid point $q'$, run a thread that reads its weight and $k$-nearest distance, next explores the grid points $q$ whose influence region $I_k(p, S \cup \{q\})$ contains $q'$, and increments its weighted area in $w$. 
Maximal $w(I_k(q, S \cup \{q\})$ obtention

The maximal $k$-influence weighted area among the obtained ones is computed in parallel using a reduction type algorithm.
# Complexity analysis

<table>
<thead>
<tr>
<th>k-nearest/farthest neighbor:</th>
<th>Weighted area estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>G threads with $O(n)$ work</td>
<td>G threads with $O(G)$ work</td>
</tr>
</tbody>
</table>

**Total work:** $O(nG + G^2)$

**Space complexity:** $O(n + G)$

- $n$: number of sites
- $G = H \times W$: discretization size
Experimental results

\[ |S| = 100 \quad k = 10 \]

Colored from dark to light green according to increasing region weight

Colored from dark to light red according to increasing \( w(NI_{10}(q, S \cup \{q\})) \)

Colored from dark to light blue according to increasing \( w(FI_{10}(q, S \cup \{q\})) \)
Experimental results

<table>
<thead>
<tr>
<th>$G$</th>
<th>100 $\times$ 100</th>
<th>500 $\times$ 500</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n \backslash k$</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>0.018</td>
<td>0.05</td>
</tr>
<tr>
<td>100</td>
<td>0.008</td>
<td>0.02</td>
</tr>
<tr>
<td>1000</td>
<td>0.006</td>
<td>0.01</td>
</tr>
<tr>
<td>10000</td>
<td>0.008</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Nearest case computational times in $seconds$.

<table>
<thead>
<tr>
<th>$G$</th>
<th>100 $\times$ 100</th>
<th>500 $\times$ 500</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n \backslash k$</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>0.064</td>
<td>0.09</td>
</tr>
<tr>
<td>100</td>
<td>0.056</td>
<td>0.07</td>
</tr>
<tr>
<td>1000</td>
<td>0.056</td>
<td>0.06</td>
</tr>
<tr>
<td>10000</td>
<td>0.089</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Farthest case computational times in $seconds$.

Strategy

if $k < n/2$ we maximize the $k$-nearest/farthest influence region weighted area
if $k > n/2$ we minimize the $(n-k)$-farthest/nearest influence region weighted area

OpenCL implementation executed in an Intel Core 2 CPU 2.13GHz, 2GB RAM and a GPU NVidia GeForce GTX 480.
THANK-YOU FOR YOUR ATTENTION!
Future work

• Error approximation analysis
• Adding weights to the points of S