Discrete Mobile Centers

André Bayer

Seminar of Distributed Computing
WS 03/04

Overview

- Introduction
- Previous Work
- Basic Algorithm
- Analysis 2-D
- Hierarchical Algorithm
- Kinetic Discrete Clustering
- Summary

Paper

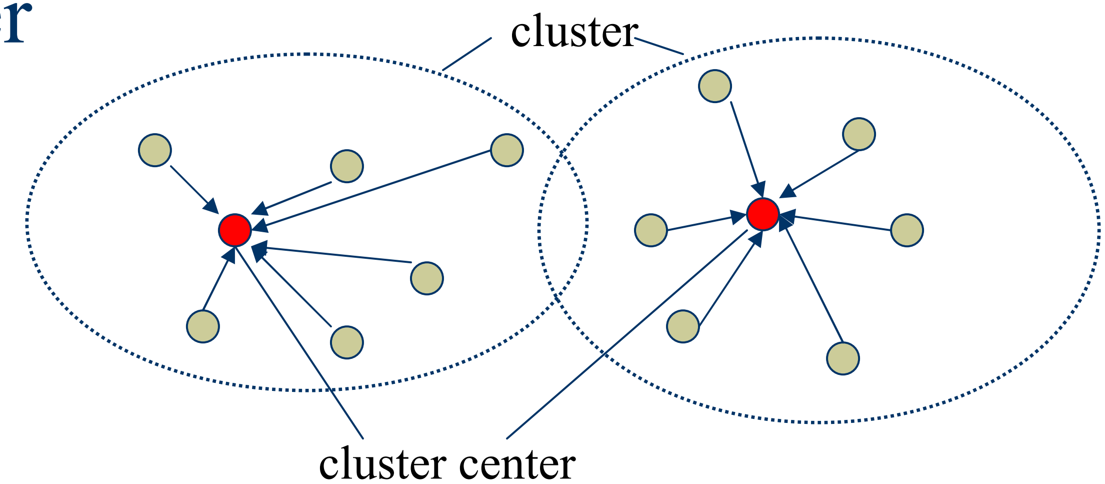
- Paper: Discrete Mobile Centers
  Jie Gao, Leonidas J. Guibas, John Hershberger, Li Zhang, An Zhu
  Published: 2001

Introduction

- Nodes in the plane
- Nodes are mobile, can switch off and on
- Short range
- For ad-hoc multi-hop networks
- Example: Bluetooth, WLAN

Problem

- Maintaining a clustering for a set of n moving points in the plane
- In communication range => visible to each other



Goal

- Minimal subset of the n nodes, the centers
- Every node is visible to at least one of the centers
- O(1)-approximation with high probability
- Smooth cluster changes
- Don‘t need the exact position
- Can be implemented in a distributed way

Previous Work

- Clustering problem = minimum dominating set
- Static version of the problem is NP-complete

- Dominating set in an intersection graph: Greedy Algorithm with const approximation

- Connected dominating set: extra condition, the subgraph must be connected.
- Marking Algorithm solves the problem presented in Mobile Computing Course SS02
  - Idea: a node is in CDS if it has two neighbors which are itself not neighbors
- Worst case: O(n)-approximation, but works well in simulation

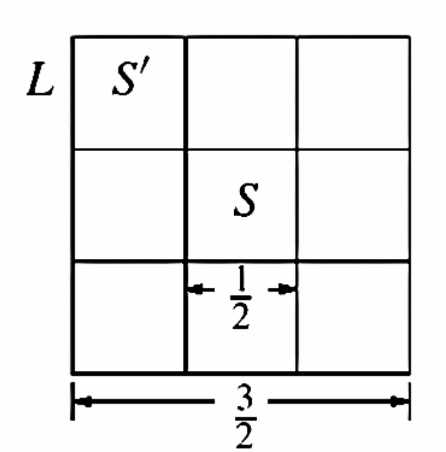
Basic Algorithm

- n points $P=\{p_1,p_2,\ldots,p_n\}$ in the plane
- Visible range: square with side length 1
- Each point can cover all points in its visible range
- Unique identifier (random number)

Description of the basic algorithm

- Each point p_i nominates the largest indexed point in its visible range to be a center (maybe itself)
- All points nominated are the centers in our solution
- A cluster is formed by a selected center and all the points that nominated it

Analysis 2-D



- 9 sub-squares with sidelength ½
- L: visible range of S
- Suppose |L| = m
- Bound for each sub-square S′ of L

Lemma: The number of centers nominated inside S is $O(\sqrt{m})$

Proof

$S' = S$ The nodes are mutually visible to each other

(complete Graph) => at most one point nominated

$S' \neq S$ Suppose x = |S| and y = |S′|

Point p ∈ S can be nominated by a point q ∈ S′ if q finds that p has the largest index in its visible range

p must have rank higher than all the points in S′

Probability that p can be nominated is at most $\frac{1}{1+y}$

At most $\frac{x}{1+y}$ points nominated

Only y points in S′

At most y centers nominated by points in S′

The expected total number of centers is no more than

$$min(y, \frac{x}{1+y}) \leq \sqrt{x+y+1} - 1 < \sqrt{m}$$

Summing over all the 9 sub-squares, the expected number of centers nominated in S is bounded by $O(\sqrt{m})$

Theorem

Theorem: The algorithm has an approximation factor of $O(\sqrt{n})$ in expectation

Proof: consider an optimal covering U_i, $1 \leq i \leq k$

Partition each U_i in the optimal solution into 4 quadrant sub-squares

Apply previous Lemma to each sub-square

$$4kc\sqrt{n} = O(\sqrt{n}) - approximation$$

High probability

- Probability that there are more than $\sqrt{n}\, log\, n \cdot k\ \ centers\ \ is\ \ O(\frac{1}{n^{log\, n - 1}})$
- k is the optimal number of centers
- High probability result

If the points are uniformly distributed, then we get a O(1)-approximation. Good performance observed in practice

Hierarchical Algorithm

- The basic algorithm is simple
- Constant approximation
- Use a hierarchical algorithm
- Proceed a number of rounds
- At each round we apply the basic algorithm to the centers produced by the previous round
- Use a larger covering ball

Details

- P_i is a cover in round i, P is the input set
- lg lg n rounds (lg n = $\log_2$ n)
- Squares with side length $\delta_i = 2^i/\lg n$
- i^{th} step, for $1 \le i < \lg \lg n$,
  apply the algorithm with squares of side length δ_i to the set P_{i-1} and let P_i be the output
- Final output is $P' = P_{\lg \lg n-1}$

Lemma

- $\alpha(x) \leq 4/x^2$
- $\alpha(x)$ the number of centers of an optimal covering of P
- x the side length of the squares

Proof: a unit square covers all the points in P

Divide the unit square into $4/x^2$ small squares of size x/2

Pick one point from each non-empty small square

This gives a covering with $4/x^2$ centers

Constant approximation

The expected size of P_{i+1} is at most $c\sqrt{|P_i|}\alpha(\delta_i)$

For some constant $c > 0$

From Theorem in 2-D analysis $O(\sqrt{n}) - approx \Leftrightarrow c\sqrt{n}k$

n_i the size of P_i

Recursive relation: $n_0 = n, \quad n_{i+1} \leq c\sqrt{n_i}\alpha(\delta_i)$

$\delta_i = 2^i/\lg n$, $\alpha(x) \leq 4/x^2$

Last round: $i = \lg \lg n-1$

We have $|P'| = n_{\lg \lg n-1} \leq c^2 2^{13} = O(1)$

Kinetic Discrete Clustering

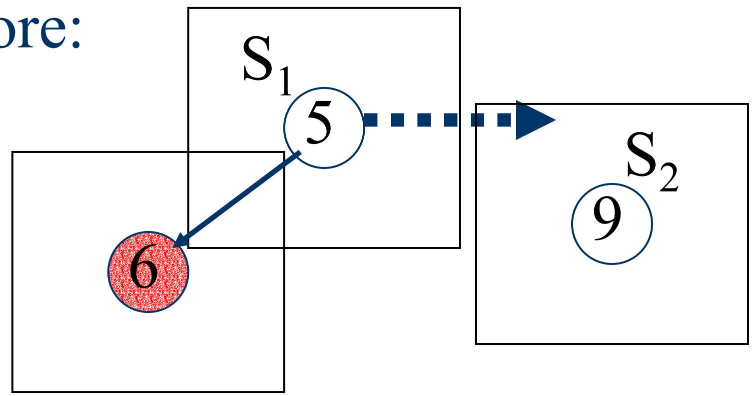
- Half-size square centered over each point
- Two squares intersect, the points are mutually visible
- Left and right extremes in x-sorted order
- Top and bottom extremes in y-sorted order
- Lists for each level of the hierarchy

Event

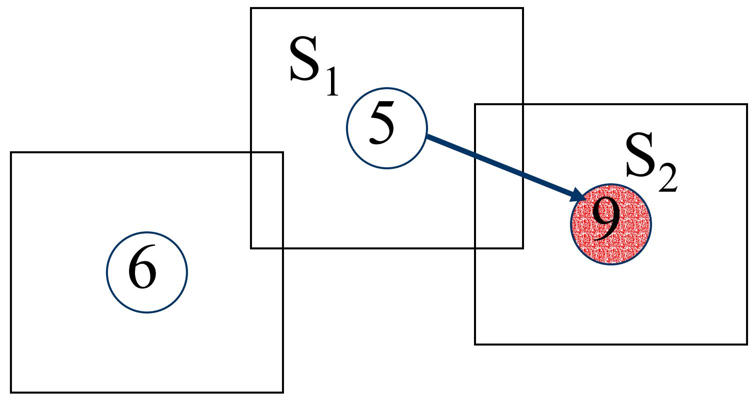
- An event is if two extremes of squares change x- or y-order
- Two cases: start or stop intersecting
- S_1, S_2 start intersecting:
  - Need to check the square with the lower rank
  - Say S_1, look at its nomination
  - If it has a lower rank than S_2, we need to change S_1 to point to S_2

Example S_1, S_2 start intersecting

before:



after:



S_1, S_2 stop intersecting

- Say S_1 lower rank than S_2
- Check if S_1 nominated S_2
- If so, find another overlapping square with the highest rank
- Data structure: standard range search tree
  - Binary tree
  - Each leaf stores a range of the interval
  - Find a point in O(log n)

- Two dimensions: find the point with the highest rank in $O(\log^2 n)$ time
- For the hierarchical algorithm, we need this structure for each level

Kinetic Properties

- Assume the points have bounded-degree algebraic motion
- Points move continuously
- Simplification to analyse the efficiency
- => each pair of points can cause O(1) events

Kinetic Properties

- The number of events in the basic algorithm is $O(n^2)$
- The number of events in the hierarchical algorithm is $O(n^2 \log \log n)$
- O(1)-approximate covering with high probability

Distributed implementation

- Each node keeps track of its neighborhood, with „who is there“ messages
- For the hierarchical algorithm, nodes broadcast with different power for each level
- New nominated centers cause updates in higher levels
- Only local operations

http://www.stanford.edu/~jgao/mobile_centers.html

Summary

- Moving points in the plane
- Given cluster radius
- Algorithm: variable subset of the nodes as cluster centers
  - Property: chosen nodes cover all the others
  - The number of centers selected is a constant-factor approximation of the minimum possible
- Use for applications in ad-hoc networks

Comment

- Hierarchical Algorithm is theoretically very interesting
- In practice?
- Linear motion realistic approach?

Questions?