Chapter 10
CLUSTERING

Mobile Computing
Winter 2005 / 2006

Overview

- Motivation
- Dominating Set
- Connected Dominating Set

- General Algorithms:
  - The “Greedy” Algorithm
  - The “Tree Growing” Algorithm
  - The “Marking” Algorithm
  - The “k-Local” Algorithm

- Algorithms for Special Models:
  - Unit Ball Graphs: The “Largest ID” Algorithm
  - Independence-Bounded Graphs: The “MIS” Algorithm
  - Unstructured Radio Network Model

Discussion

- We have seen: 10 Tricks → 2^10 routing algorithms
  - In reality there are almost that many!

- Q: How good are these routing algorithms?!? Any hard results?
  - A: Almost none! Method-of-choice is simulation…
  - Perkins: “if you simulate three times, you get three different results”

- Flooding is key component of (many) proposed algorithms, including most prominent ones (AODV, DSR)
  - At least flooding should be efficient

Finding a Destination by Flooding
Finding a Destination *Efficiently*

**Backbone**

- Idea: Some nodes become backbone nodes (gateways). Each node can access and be accessed by at least one backbone node.

- Routing:
  1. If source is not a gateway, transmit message to gateway.
  2. Gateway acts as proxy source and routes message on backbone to gateway of destination.
  3. Transmission gateway to destination.

**(Connected) Dominating Set**

- A **Dominating Set** $DS$ is a subset of nodes such that each node is either in $DS$ or has a neighbor in $DS$.

- A **Connected Dominating Set** $CDS$ is a connected $DS$, that is, there is a path between any two nodes in $CDS$ that does not use nodes that are not in $CDS$.

- A $CDS$ is a good choice for a backbone.

- It might be favorable to have few nodes in the $CDS$. This is known as the Minimum $CDS$ problem.

**Formal Problem Definition: M(C)DS**

- **Input**: We are given an (arbitrary) undirected graph.

- **Output**: Find a Minimum (Connected) Dominating Set, that is, a $(C)DS$ with a minimum number of nodes.

- **Problems**
  - $M(C)DS$ is NP-hard
  - Find a $(C)DS$ that is “close” to minimum (approximation)
  - The solution must be local (global solutions are impractical for mobile ad-hoc network) – topology of graph “far away” should not influence decision who belongs to $(C)DS$
Greedy Algorithm for Dominating Sets

- Idea: Greedy choose "good" nodes into the dominating set.
- Black nodes are in the DS
- Grey nodes are neighbors of nodes in the DS
- White nodes are not yet dominated, initially all nodes are white.

- Algorithm: Greedily choose a node that colors most white nodes.

One can show that this gives a \( \log \Delta \) approximation, if \( \Delta \) is the maximum node degree of the graph. (The proof is similar to the "Tree Growing" proof on 6/13ff.)

One can also show that there is no polynomial algorithm with better performance unless \( P \approx NP \).

CDS: The “too simple tree growing” algorithm

- Idea: start with the root, and then greedily choose a neighbor of the tree that dominates as many as possible new nodes
- Black nodes are in the CDS
- Grey nodes are neighbors of nodes in the CDS
- White nodes are not yet dominated, initially all nodes are white.

- Start: Choose a node with maximum degree, and make it the root of the CDS, that is, color it black (and its white neighbors grey).

- Step: Choose a grey node with a maximum number of white neighbors and color it black (and its white neighbors grey).

Example of the “too simple tree growing” algorithm

Graph with \( 2n+2 \) nodes; tree growing: \(|\text{CDS}|=n+2\); Minimum \(|\text{CDS}|=4\)

Tree Growing Algorithm

- Idea: Don’t scan one but two nodes!

- Alternative step: Choose a grey node and its white neighbor node with a maximum sum of white neighbors and color both black (and their white neighbors grey).
Analysis of the tree growing algorithm

- Theorem: The tree growing algorithm finds a connected set of size $|CDS| \leq 2(1+H(\Delta)) \cdot |DS_{OPT}|$.

- \(DS_{OPT}\) is a (not connected) minimum dominating set.
- \(\Delta\) is the maximum node degree in the graph.
- \(H\) is the harmonic function with \(H(n) \approx \log(n) + 0.7\).

- In other words, the connected dominating set of the tree growing algorithm is at most a \(O(\log(\Delta))\) factor worse than an optimum minimum dominating set (which is NP-hard to compute).

- With a lower bound argument (reduction to set cover) one can show that a better approximation factor is impossible, unless \(P = NP\).

Proof Sketch

- The proof is done with amortized analysis.

- Let \(S_u\) be the set of nodes dominated by \(u \in DS_{OPT}\), or \(u\) itself. If a node is dominated by more than one node, we put it in one of the sets.

- We charge the nodes in the graph for each node we color black. In particular we charge all the newly colored grey nodes. Since we color a node grey at most once, it is charged at most once.

- We show that the total charge on the vertices in an \(S_u\) is at most \(2(1+H(\Delta))\), for any \(u\).

Charge on \(S_u\)

- Initially \(|S_u| = u_0\).
- Whenever we color some nodes of \(S_u\), we call this a step.
- The number of white nodes in \(S_u\) after step \(i\) is \(u_i\).
- After step \(k\) there are no more white nodes in \(S_u\).
- In the first step \(u_0 - u_1\) nodes are colored (grey or black). Each vertex gets a charge of at most \(2/(u_0 - u_1)\).
- After the first step, node \(u\) becomes eligible to be colored (as part of a pair with one of the grey nodes in \(S_u\)). If \(u\) is not chosen in step \(i\) (with a potential to paint \(u\) nodes grey), then we have found a better (pair of) node. That is, the charge to any of the new grey nodes in step \(i\) in \(S_u\) is at most \(2/u_i\).

Adding up the charges in \(S_u\)

\[
C \leq \frac{2}{u_0 - u_1}(u_0 - u_1) + \sum_{i=1}^{k-1} \frac{2}{u_i}(u_i - u_{i+1})
\]

\[
= 2 + 2 \sum_{i=1}^{k-1} \frac{u_i - u_{i+1}}{u_i}
\]

\[
\leq 2 + 2 \sum_{i=1}^{k-1} (H(u_i) - H(u_{i+1}))
\]

\[
= 2 + 2(H(u_1) - H(u_k)) = 2(1 + H(u_1)) = 2(1 + H(\Delta))
\]
Discussion of the tree growing algorithm

- We have an extremely simple algorithm that is asymptotically optimal unless \( P \neq NP \). And even the constants are small.

- Are we happy?

- Not really. How do we implement this algorithm in a real mobile network? How do we figure out where the best grey/white pair of nodes is? How slow is this algorithm in a distributed setting?

- We need a fully distributed algorithm. Nodes should only consider local information.

The Marking Algorithm

- Idea: The connected dominating set CDS consists of the nodes that have two neighbors that are not neighboring.

1. Each node \( u \) compiles the set of neighbors \( N(u) \)
2. Each node \( u \) transmits \( N(u) \), and receives \( N(v) \) from all its neighbors
3. If node \( u \) has two neighbors \( v, w \) and \( w \) is not in \( N(v) \) (and since the graph is undirected \( v \) is not in \( N(w) \)), then \( u \) marks itself being in the set CDS.

+ Completely local; only exchange \( N(u) \) with all neighbors
+ Each node sends only 1 message, and receives at most \( \Delta \)
+ Messages have size \( O(\Delta) \)

- Is the marking algorithm really producing a connected dominating set? How good is the set?

Example for the Marking Algorithm

Correctness of Marking Algorithm

- We assume that the input graph \( G \) is connected but not complete.

- Note: If \( G \) was complete then constructing a CDS would not make sense. Note that in a complete graph, no node would be marked.

- We show:

  The set of marked nodes CDS is
  a) a dominating set
  b) connected
  c) a shortest path in \( G \) between two nodes of the CDS is in CDS
Proof of a) dominating set

• Proof: Assume for the sake of contradiction that node \( u \) is a node that is not in the dominating set, and also not dominated. Since no neighbor of \( u \) is in the dominating set, the nodes \( N^+(u) := u \cup N(u) \) form:

  • a complete graph
    – if there are two nodes in \( N(u) \) that are not connected, \( u \) must be in the dominating set by definition
  • no node \( v \in N(u) \) has a neighbor outside \( N(u) \)
    – or, also by definition, the node \( v \) is in the dominating set

• Since the graph \( G \) is connected it only consists of the complete graph \( N^+(u) \). We precluded this in the assumptions, therefore we have a contradiction

Proof of b) connected, c) shortest path in CDS

• Proof: Let \( p \) be any shortest path between the two nodes \( u \) and \( v \), with \( u,v \in CDS \).

• Assume for the sake of contradiction that there is a node \( w \) on this shortest path that is not in the connected dominating set.

• Then the two neighbors of \( w \) must be connected, which gives us a shorter path. This is a contradiction.

Improved Marking Algorithm

• If neighbors with larger ID are connected and cover all other neighbors, then don’t join CDS, else join CDS

Correctness of Improved Marking Algorithm

• Theorem: Algorithm computes a CDS \( S \)

• Proof (by induction of node IDs):
  – assume that initially all nodes are in \( S \)
  – look at nodes \( u \) in increasing ID order and remove from \( S \) if higher-ID neighbors of \( u \) are connected
  – \( S \) remains a DS at all times: (assume that \( u \) is removed from \( S \))
  – \( S \) remains connected: replace connection \( v-u-v' \) by \( v-n_1,...,n_k-v' \) (\( n_i \): higher-ID neighbors of \( u \))
Quality of the (Improved) Marking Algorithm

- Given an Euclidean chain of n homogeneous nodes
- The transmission range of each node is such that it is connected to the k left and right neighbors, the id’s of the nodes are ascending.

- An optimal algorithm (and also the tree growing algorithm) puts every k’th node into the CDS. Thus |CDS_{OPT}| \approx n/k; with k = n/c for some positive constant c we have |CDS_{OPT}| = O(1).

- The marking algorithm (also the improved version) does mark all the nodes (except the k leftmost ones). Thus |CDS_{Marking}| = n – k; with k = n/c we have |CDS_{Marking}| = \Omega(n).

- The worst-case quality of the marking algorithm is worst-case! 😊

Phase A is a Distributed Linear Program

- Nodes 1, ..., n: Each node u has variable x_u with x_u \geq 0
- Sum of x-values in each neighborhood at least 1 (local)
- Minimize sum of all x-values (global)

Linear Program

\[
\begin{align*}
\text{min} & \quad \sum_{i=1}^{n} x_i \\
\text{subject to} & \quad N \cdot x \geq 1 \\
& \quad x \geq 0
\end{align*}
\]

Adjoint matrix with 1’s in diagonal

- Linear Programs can be solved optimally in polynomial time
- But not in a distributed fashion! That's what we need here...

Phase A Algorithm

1. \( x_1 := 0; \)
2. for \( u = 2 \) to \( n \) do
3. \( \lambda := \text{min} \sum_{j \in N_u} x_j \) 
4. if \( \lambda \geq 1 \) then
5. \( x_u := x_u + \lambda \)
6. else if \( \lambda \geq 2 \) then
7. \( x_u := x_u + 2 \lambda \)
8. \( x_u := x_u + \lambda \)
9. end if
10. end for
11. procedure increase_duals():
12. if \( w_i \geq 1 \) then
13. \( w_i := w_i + \lambda \)
14. else if \( w_i \geq 2 \) then
15. \( w_i := w_i + 2 \lambda \)
16. end if
17. end procedure
18. procedure decrease_duals():
19. if \( w_i \leq 0 \) then
20. \( w_i := w_i - \lambda \)
21. else if \( w_i \leq -\lambda \) then
22. \( w_i := w_i - 2 \lambda \)
23. end if
24. end procedure

Phase A: Distributed linear program rel. high degree gives high value

Phase B: Probabilistic algorithm

Phase C: Connect DS by "tree" of "bridges"
Result after Phase A

- Distributed Approximation for Linear Program
- Instead of the optimal values $x_i^*$ at nodes, nodes have $x_i^{(a)}$, with
  \[
  \sum_{i=1}^{n} x_i^{(a)} \leq \alpha \cdot \sum_{i=1}^{n} x_i^*
  \]
- The value of $\alpha$ depends on the number of rounds $k$ (the locality)
  \[\alpha \leq (\Delta + 1)^{c/\sqrt{k}}\]
- The analysis is rather intricate… 😊

Phase B Algorithm

Each node applies the following algorithm:

1. Calculate $\delta_i^{(2)}$ (max degree of neighbors in distance 2)
2. Become a dominator (i.e. go to the dominating set) with probability
   \[p_i := \min\{1, x_i^{(a)} \cdot \ln(\delta_i^{(2)} + 1)\}\]
   - From phase A
   - Highest degree in distance 2
3. Send status (dominator or not) to all neighbors
4. If no neighbor is a dominator, become a dominator yourself

Result after Phase B

- Randomized rounding technique
- Expected number of nodes joining the dominating set in step 2 is bounded by $\alpha \log(\Delta + 1) \cdot |DS_{OPT}|$.
- Expected number of nodes joining the dominating set in step 4 is bounded by $|DS_{OPT}|$.

Theorem: $E[|DS|] = O((\Delta + 1)^{c/\sqrt{\Delta}} \log \Delta \cdot |DS_{OPT}|)$

- Phase C → essentially the same result for CDS

A better algorithm?

Unit Disk Graph
Better and faster algorithm

- Assume that graph is a unit disk graph (UDG)
- Assume that nodes know their positions (GPS)

Then...

Grid Algorithm

1. Beacon your position
2. If, in your virtual grid cell, you are the node closest to the center of the cell, then join the CDS, else do not join.
3. That’s it.

- 1 transmission per node, O(1) approximation.
- If you have mobility, then simply “loop” through algorithm, as fast as your application/mobility wants you to.

Comparison

**k-local algorithm**
- Algorithm computes DS
- $k^2 + O(1)$ transmissions/node
- $O(\Delta^{O(1)\times \log \Delta})$ approximation

**Grid algorithm**
- Algorithm computes DS
- 1 transmission/node
- O(1) approximation
- General graph
- No position information
- Unit disk graph (UDG)
- Position information (UDG)

The model determines the distributed complexity of clustering
Let’s talk about models…

- General Graph
  - Captures obstacles
  - Captures directional radios
  - Often too pessimistic

- UDG & GPS
  - UDG is not realistic
  - GPS not always available
    - Indoors
  - 2D → 3D?
  - Often too optimistic

Let’s look at models in between these extremes!

Models

- General Graph
- UDG & GPS
- UDG
- No GPS
- GPS

- Bounded Independence
- Unit Ball Graph
- Quasi UDG

Unit Ball Graphs

- ∃ metric \((V,d)\) describing distances between nodes \(u,v \in V\)
  such that:
  \[
  \begin{align*}
  d(u,v) &\leq 1 : (u,v) \in E \\
  d(u,v) &\geq 1 : (u,v) \not\in E
  \end{align*}
  \]

- Assume that doubling dimension of metric is constant

- Doubling Dimension: \(\log(\text{#balls of radius } r/2 \text{ to cover ball of radius } r)\)

Unit Ball Graph

UBG based on underlying doubling metric.
The “Largest-ID” Algorithm

- All nodes have unique IDs, chosen at random.
- Algorithm for each node:
  1. Send ID to all neighbors
  2. Tell node with largest ID in neighborhood that it has to join the DS
- Algorithm computes a DS in 2 rounds (extremely local!)

“Largest ID” Algorithm, Analysis I

- To simplify analysis: assume graph is UDG
  (same analysis works for UBG based on doubling metric)
- We look at a disk $S$ of diameter 1:

Nodes inside $S$ have distance at most 1.
$\rightarrow$ they form a clique

How many nodes in $S$ are selected for the DS?

“Largest ID” Algorithm, Analysis II

- Nodes which select nodes in $S$ are in disk of radius 3/2 which can be covered by $S$ and 20 other disks $S_i$ of diameter 1
  (UBG: number of small disks depends on doubling dimension)

“Largest ID” Algorithm: Analysis III

- How many nodes in $S$ are chosen by nodes in a disk $S_i$?

  $x = \#$ of nodes in $S$, $y = \#$ of nodes in $S_i$:

  A node $u \in S$ is only chosen by a node in $S_i$ if $\text{ID}(u) > \max\{\text{ID}(v)\}$
  (all nodes in $S_i$ see each other).

  The probability for this is: $\frac{1}{1+y}$

  Therefore, the expected number of nodes in $S$ chosen by nodes in $S_i$ is at most:

  $$\min\left\{y, \frac{x}{1+y}\right\}$$

  Because at most $y$ nodes in $S_i$ can choose nodes in $S$ and because of linearity of expectation.
“Largest ID” Algorithm, Analysis IV

- From \( x \leq n \) and \( y \leq n \), it follows that: \( \min \left\{ y, \frac{x}{1+y} \right\} \leq \sqrt{n} \)
- Hence, in expectation the DS contains at most \( 20\sqrt{n} \) nodes per disk with diameter 1.
- An optimal algorithm needs to choose at least 1 node in the disk with radius 1 around any node.
- This disk can be covered by a constant (9) number of disks of diameter 1.
- The algorithm chooses at most \( O(\sqrt{n}) \) times more disks than an optimal one.

“Largest ID” Algorithm, Remarks

- For typical settings, the “Largest ID” algorithm produces very good dominating sets (also for non-UDGs).
- There are UDGs where the “Largest ID” algorithm computes an \( \Theta(\sqrt{n}) \)-approximation (analysis is tight).

Iterative “Largest ID” Algorithm

- Assume that nodes know the distances to their neighbors:
  all nodes are active;
  for \( i := k \) to 1 do
    \( \forall \) act. nodes: select act. node with largest ID in dist. \( \leq 1/2^i \);
    selected nodes remain active
  od;
  DS = set of active nodes
- Set of active nodes is always a DS (computing CDS also possible)
- Number of rounds: \( k \)
- Approximation ratio \( n^{1/2^k} \)
- For \( k = O(\log \log n) \), approximation ratio = \( O(1) \)

Iterative “Largest ID” Algorithm, Remarks

- Possible to do everything in \( O(1) \) rounds (messages get larger, local computations more complicated)
- If we slightly change the algorithm such that largest radius is \( 1/4 \):
  - Sufficient to know IDs of all neighbors, distances to neighbors, and distances between adjacent neighbors
  - Every node can then locally simulate relevant part of algorithm to find out whether or not to join DS
- Doubling UBG: \( O(1) \) approximation in \( O(1) \) rounds
Models

- General Graph
- UDG
- UDG No GPS
- UDG GPS
- too pessimistic
- too optimistic
- Bounded Independence
- Number of independent neighbors is bounded (UDG: 5)
- Unit Ball Graph
- In a doubling metric:
- Quasi UDG

Real Networks

- Wireless Networks are not unit disk graphs, but:
  - No links between far-away nodes
  - Close nodes tend to be connected
  - In particular: Densely covered area → many connections

Bounded Independence

- Def.: A graph $G$ has bounded independence if there is a function $f(r)$ such that every $r$-neighborhood in $G$ contains at most $f(r)$ independent nodes.
  - Note: $f(r)$ does not depend on size of the graph!
  - Polynomially Bounded Independence: $f(r) = \text{poly}(r)$, e.g. $O(r^2)$

  1) A node can have many neighbors
  2) But not all of them can be independent!
  3) Can model obstacles, walls, ...

- $f(1) = 6$

Maximal Independent Set I

- Maximal Independent Set (MIS): (non-extendable set of pair-wise non-adjacent nodes)
  - An MIS is also a dominating set:
    - assume that there is a node $v$ which is not dominated
    - $\forall v \in \text{MIS}, (u,v) \in E \rightarrow u \notin \text{MIS}$
    - add $v$ to MIS
Maximal Independent Set II

- **Lemma:**
  On independence-bounded graphs: \( |\text{MIS}| \leq O(1) \cdot |\text{DS}_{\text{OPT}}| \)

- **Proof:**
  1. Assign every MIS node to an adjacent node of DS_{OPT}
  2. \( u \in \text{DS}_{\text{OPT}} \) has at most \( f(1) \) neighbors \( v \in \text{MIS} \)
  3. At most \( f(1) \) MIS nodes assigned to every node of DS_{OPT}
  \( \Rightarrow |\text{MIS}| \leq f(1) \cdot |\text{DS}_{\text{OPT}}| \)

- Time to compute MIS on independence-bounded graphs:
  \( O(\log \Delta \cdot \log^* n) \)

### MIS (DS) \( \rightarrow \) CDS

- MIS gives a dominating set.
- But it is not connected.
- Connect any two MIS nodes which can be connected by one additional node.
- Connect unconnected MIS nodes which can be conn. by two additional nodes.
- This gives a CDS!
- \( \#2\text{-hop connectors} \leq f(2) \cdot |\text{MIS}| \)
- \( \#3\text{-hop connectors} \leq 2(3) \cdot |\text{MIS}| \)
- \( |\text{CDS}| = O(|\text{MIS}|) \)

### Models

- **General Graph** too pessimistic
- **UDG No GPS** too realistic
- **UDG GPS** too simplistic

- **Bounded Independence** too pessimistic
- **Unit Ball Graph** too optimistic
- **Quasi UDG** too realistic
- **Message Passing Models** too pessimistic
- **Radio Network Model** too optimistic

- **Physical Signal Propagation** too pessimistic
- **Unstructured Radio Network Model** too realistic

### Unstructured Radio Network Model

- **Multi-Hop**
- **No collision detection**
  - Not even at the sender!
- **No knowledge** about (the number of) neighbors
- **Asynchronous Wake-Up**
  - Nodes are not woken up by messages!
- **Unit Disk Graph (UDG)** to model wireless multi-hop network
  - Two nodes can communicate if Euclidean distance is at most 1
- **Upper bound** \( n \) for number of nodes in network is known
  - This is necessary due to \( \Omega(n / \log n) \) lower bound
  [Jurdzinski, Stachowiak, ISAAC 2002]
Unstructured Radio Network Model

- Can MDS and MIS be solved efficiently in such a harsh model?

There is a MIS algorithm with running time $O(\log^2 n)$ with high probability.