

# Clustering

## Chapter 7



## Rating

- Area maturity

First steps

Text book

- Practical importance

No apps

Mission critical

- Theoretical importance

Not really

Must have

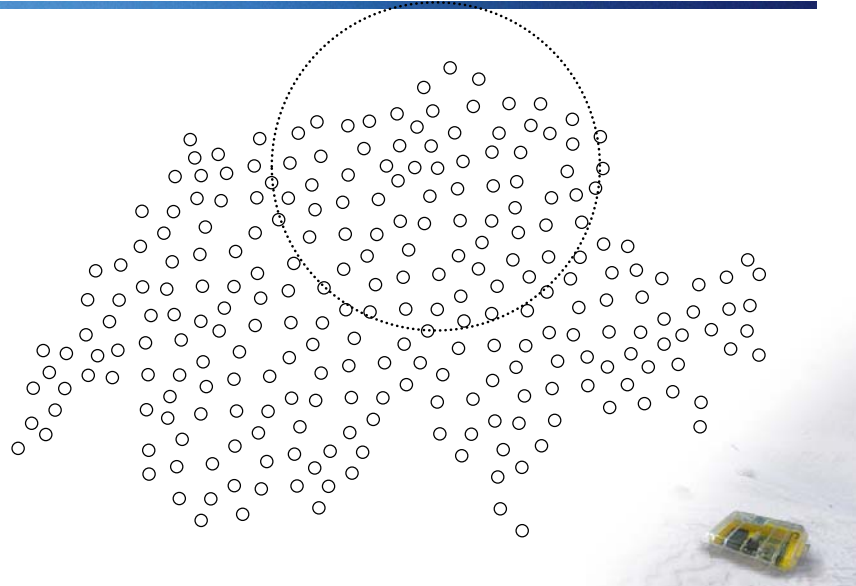
## Overview

- 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
  - Bounded Independence Graphs: The “MIS” Algorithm
  - Unstructured Radio Network Model

## Motivation

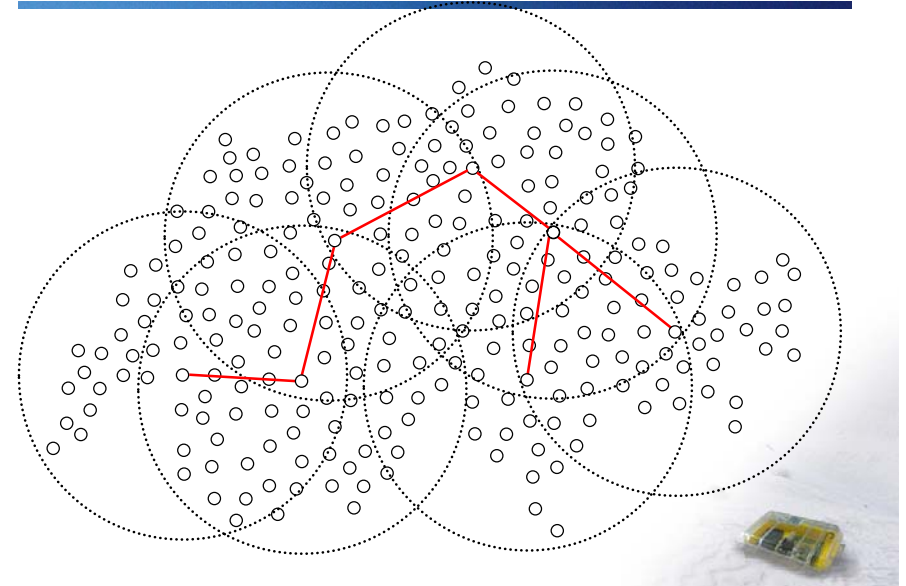
- In theory clustering is the solution to almost any problem in ad hoc and sensor networks. It improves almost any algorithm, e.g. in data gathering it selects cluster heads which do the work while other nodes can save energy by sleeping. Here, however, we motivate clustering with routing:
- There are thousands of routing algorithms...
- Q: How good are these routing algorithms?!? **Any hard results?**
- A: Almost none! Method-of-choice is simulation...
- **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



Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/5

## Finding a Destination Efficiently



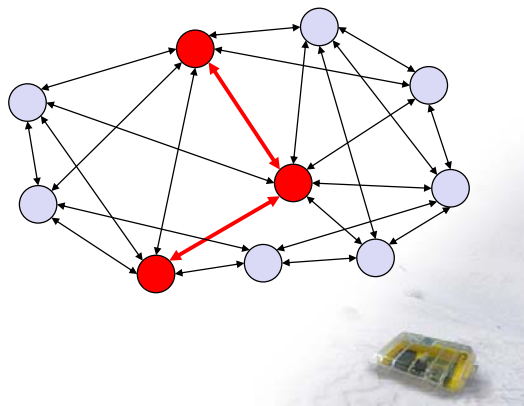
Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/6

## Backbone

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

### Routing:

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

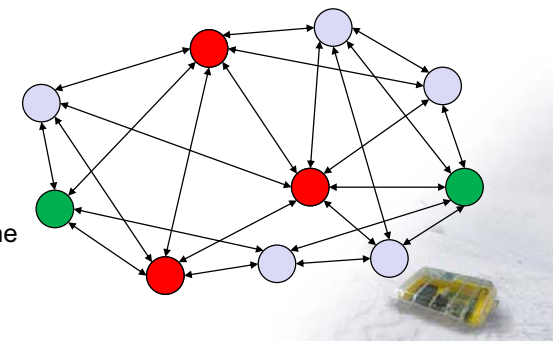


Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/7

## (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**

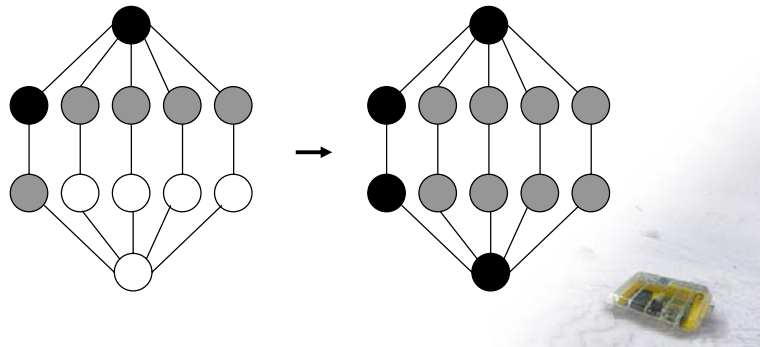


Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/8



## 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).



Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/13

## 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  $P \approx NP$ .

Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/14

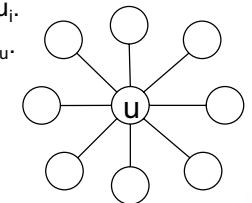
## Proof Sketch

- 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  $u$ .

Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/15

## 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  $u_i$  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  $2/u_i$ .

Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/16

## Adding up the charges in $S_u$

$$\begin{aligned}C &\leq \frac{2}{u_0 - u_1}(u_0 - u_1) + \sum_{i=1}^{k-1} \frac{2}{u_i}(u_i - u_{i+1}) \\ &= 2 + 2 \sum_{i=1}^{k-1} \frac{u_i - u_{i+1}}{u_i} \\ &\leq 2 + 2 \sum_{i=1}^{k-1} (H(u_i) - H(u_{i+1})) \\ &= 2 + 2(H(u_1) - H(u_k)) = 2(1 + H(u_1)) = 2(1 + H(\Delta))\end{aligned}$$

## Discussion of the tree growing algorithm

- We have an extremely simple algorithm that is asymptotically optimal unless  $P \approx NP$ . 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.

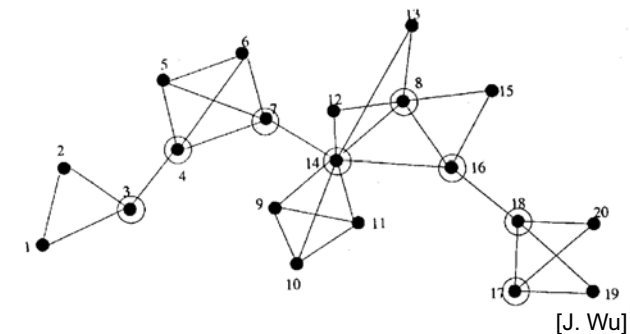


## The Marking Algorithm

- 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



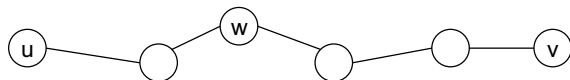
## Proof of a) dominating set

- 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  $u, v \in \text{CDS}$ .
- Assume for the sake of contradiction that there is a node  $w$  on this shortest path that is not in the connected dominating set.

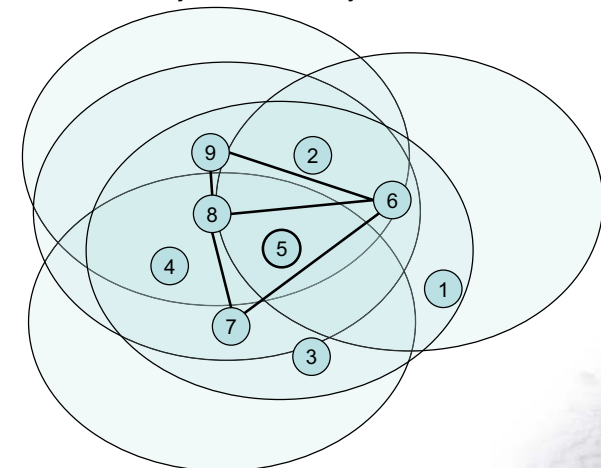


- Then the two neighbors of  $w$  must be connected, which gives us a shorter path. This is a contradiction.



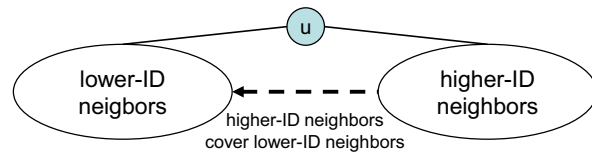
## Improved Marking Algorithm

- 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  $S$ )

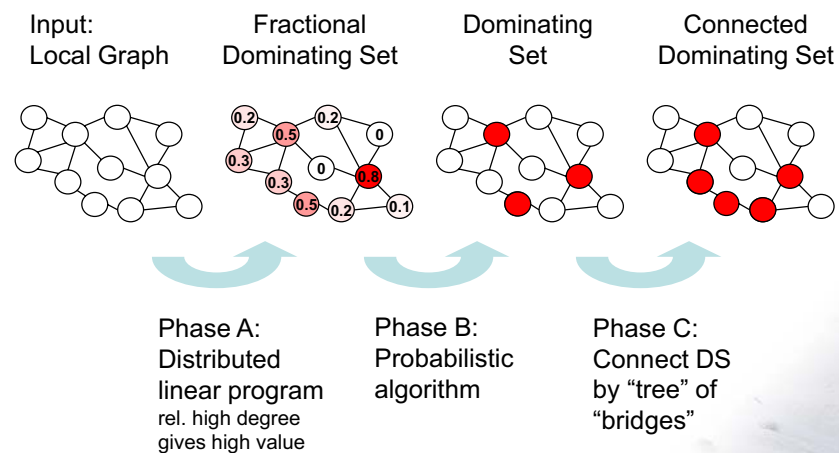


- $S$  remains connected:  
replace connection  $v-u-v'$  by  $v-n_1, \dots, n_k-v'$  ( $n_i$ : higher-ID neighbors of  $u$ )

## Quality of the (Improved) Marking Algorithm

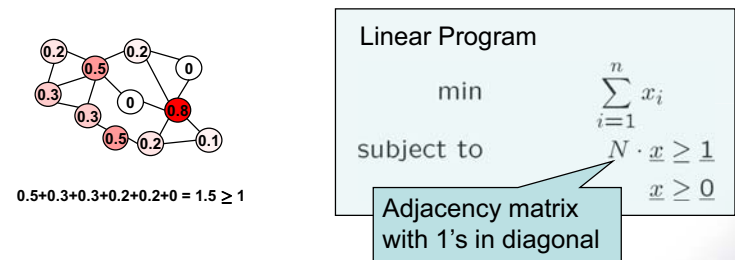
- Given an Euclidean chain of  $n$  homogeneous nodes
- The transmission range of each node is such that it is connected to the  $k$  left and right neighbors, the id's of the nodes are ascending.
  - $\circ \circ$
- An optimal algorithm (and also the tree growing algorithm) puts every  $k$ 'th node 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! ☺

## Algorithm Overview



## Phase A is a Distributed Linear Program

- Nodes  $1, \dots, n$ : 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 Programs can be solved optimally in polynomial time
- But **not in a distributed fashion!** That's what we need here...

## Phase A Algorithm

### LP Approximation Algorithm for Primal Node $v_i^{(p)}$ :

```

1:  $x_i := 0$ ;
2: for  $e_p := k_p - 2$  to  $-f - 1$  by  $-1$  do
3:   for 1 to  $h$  do
4:     (*  $\gamma_i := \frac{\sum_{j \in N_i^{(p)}} a_{ij} r_j^*$  *)
5:     for  $e_d := k_d - 1$  to 0 by  $-1$  do
6:        $\tilde{\gamma}_i := \frac{\sum_{j \in N_i^{(p)}} a_{ij} r_j^*}{c_i}$ ;
7:       if  $\tilde{\gamma}_i \geq 1/\Gamma_p^{e_p/k_p}$  then
8:          $x_i^+ := 1/\Gamma_p^{e_d/k_d}$ ;  $x_i := x_i + x_i^+$ ;
9:       fi;
10:      send  $x_i^+$ ,  $\tilde{\gamma}_i$  to dual neighbors;
11:    od;
12:  od;
13:  receive  $\tilde{r}_j$  from dual neighbors
14:  od;
15:  receive  $r_j$  from dual neighbors
16:  od;
17:  od;
18:  receive  $r_j$  from dual neighbors
19:  od;
20:  od;
21:  $x_i := x_i / \min_{j \in N_i^{(p)}} \sum_{\ell} a_{j\ell} x_\ell$ 

```

### LP Approximation Algorithm for Dual Node $v_i^{(d)}$ :

```

1:  $y_i := \tilde{y}_i^+ := w_i := f_i := 0$ ;  $r_i := 1$ ;
2: for  $e_p := k_p - 2$  to  $-f - 1$  by  $-1$  do
3:   for 1 to  $h$  do
4:      $\tilde{r}_i := r_i$ ;
5:     for  $e_d := k_d - 1$  to 0 by  $-1$  do
6:       procedure increase_duals():
7:         1: if  $w_i \geq 1$  then
8:           2: if  $f_i \geq f$  then
9:             3:  $y_i := y_i + y_i^+$ ;  $y_i^+ := 0$ ;
10:            4:  $r_i := 0$ ;  $w_i := 0$ 
11:           5: else if  $w_i \geq 2$  then
12:             6:  $y_i := y_i + y_i^+$ ;  $y_i^+ := 0$ ;
13:            7:  $r_i := r_i / \Gamma_p^{-\lceil w_i \rceil / k_p}$ 
14:           8: else
15:             9:  $\lambda := \max\{\Gamma_d^{1/k_d}, \Gamma_p^{1/k_p}\}$ ;
16:            10:  $y_i := y_i + \min\{y_i^+, r_i \lambda / \Gamma_p^{e_p/k_p}\}$ ;
17:            11:  $y_i^+ := y_i^+ - \min\{y_i^+, r_i \lambda / \Gamma_p^{e_p/k_p}\}$ ;
18:            12:  $r_i := r_i / \Gamma_p^{1/k_p}$ 
19:           13: fi;
20:            14:  $w_i := w_i - \lfloor w_i \rfloor$ 
21:           15: fi
22:       end procedure
23:       receive  $x_j^+$ ,  $\tilde{\gamma}_j$  from
24:         11:  $y_i^+ := y_i^+ + \tilde{\gamma}_j \sum_j$ 
25:         12:  $w_i^+ := \sum_j a_{ij} x_j^+$ ;
26:         13:  $w_i := w_i + w_i^+$ ;  $f_i$ ;
27:         14: if  $w_i \geq 1$  then  $\tilde{r}_i$ ;
28:         15: send  $\tilde{r}_i$  to primal n
29:       od;
30:     od;
31:     increase_duals();
32:     send  $r_i$  to primal nei
33:   od;
34:  od;
35:  od;
36:  od;
37:   $y_i := y_i / \max_{j \in N_i^{(d)}} \frac{1}{c_j} \sum$ 

```

## Result after Phase A

- **Distributed Approximation** for Linear Program
- Instead of the optimal values  $x_i^*$  at nodes, nodes have  $x_i^{(\alpha)}$ , with

$$\sum_{i=1}^n x_i^{(\alpha)} \leq \alpha \cdot \sum_{i=1}^n x_i^*$$

- 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  $\delta_i^{(2)}$  (= maximum degree of neighbors in distance 2)
2. **Become a dominator** (i.e. go to the dominating set) with probability

$$p_i := \min\{1, x_i^{(\alpha)} \cdot \ln(\delta_i^{(2)} + 1)\}$$

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  $\alpha \log(\Delta+1) \cdot |\text{DS}_{\text{OPT}}|$ .
- Expected number of nodes joining the dominating set in step 4 is bounded by  $|\text{DS}_{\text{OPT}}|$ .

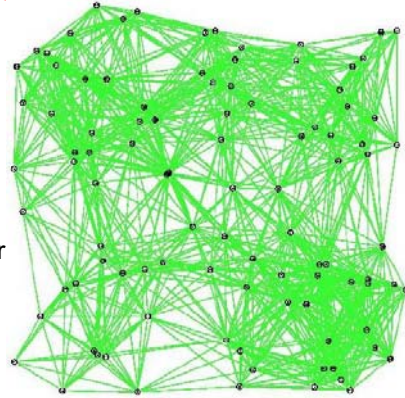
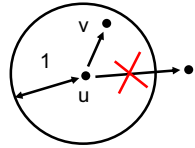
$$\text{Theorem: } E[|\text{DS}|] = O\left((\Delta + 1)^{c/\sqrt{k}} \log \Delta \cdot |\text{DS}_{\text{OPT}}|\right)$$

- Phase C  $\rightarrow$  essentially the same result for CDS



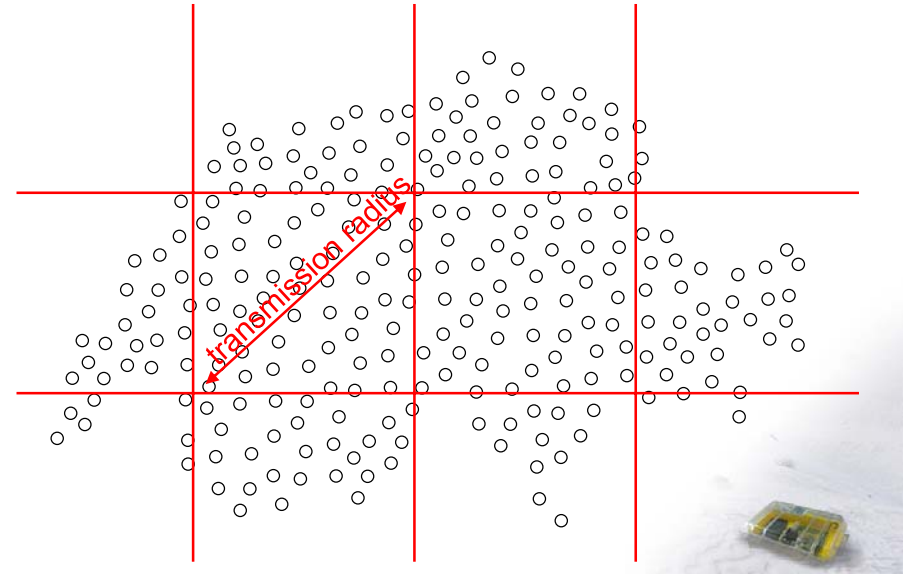
## Better and faster algorithm

- Assume that graph is a **unit disk graph (UDG)**



- Assume that nodes know their

## Then...



Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/34

## Grid Algorithm

- Beacon your position
  - If, in your virtual grid cell, you are the node closest to the center of the cell, then join the DS, else do not join.
  - 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.

Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/35

## Comparison

### k-local algorithm

- Algorithm computes DS
- $k^2 + O(1)$  transmissions/node
- $O(\Delta^{O(1)/k} \log \Delta)$  approximation
- General graph
- No position information

### Grid algorithm

- Algorithm computes DS
- 1** transmission/node
- $O(1)$**  approximation
- Unit disk graph (UDG)
- Position information (GPS)

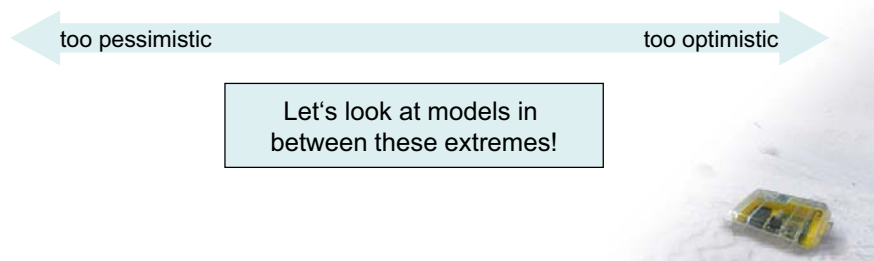


The **model** determines the distributed **complexity** of clustering

Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/36

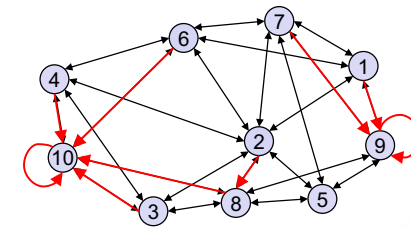
## Let's talk about models...

- 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**



## The “Largest-ID” Algorithm

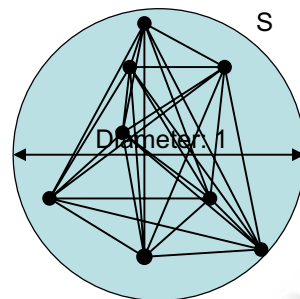
- All nodes have unique IDs, chosen at random.
- Algorithm for each node:
  1. Send ID to all neighbors
  2. Tell node with largest ID in neighborhood that it has to join the DS
- Algorithm computes a DS in 2 rounds (extremely local!!)



## “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:

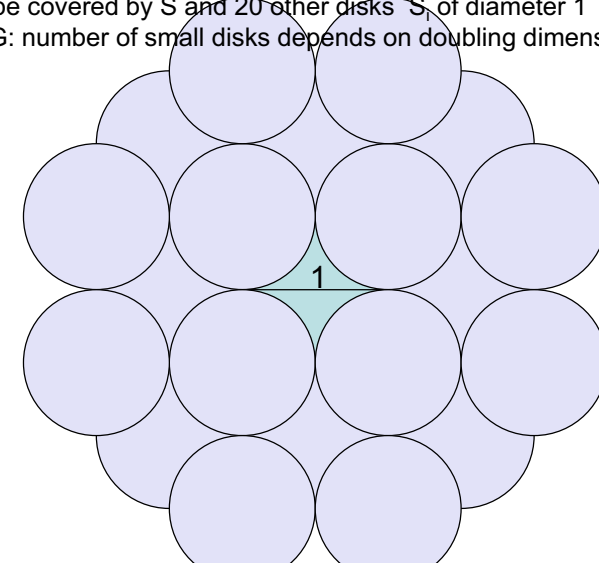
Nodes inside  $S$  have distance at most 1.  
→ they form a clique



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  $ID(u) > \max_{v \in S_i} \{ID(v)\}$  (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.

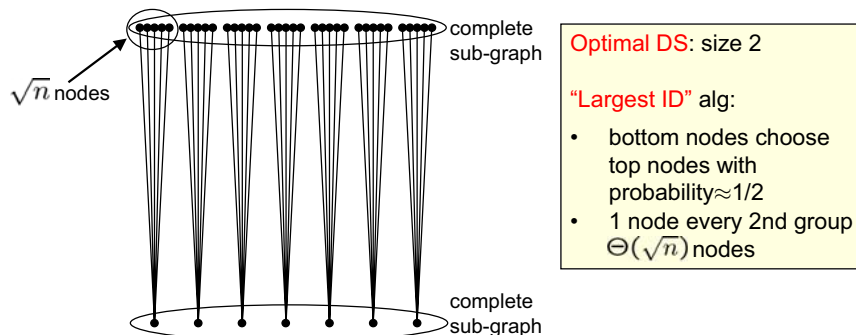
## “Largest ID” Algorithm, Analysis IV

- 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

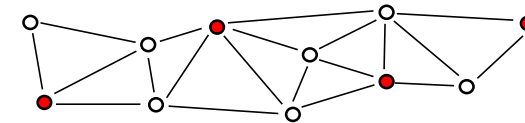
- Possible to do everything in  $O(1)$  rounds (messages get larger, local computations more complicated)
- If we slightly change the algorithm such that largest radius is  $1/4$ :
  - Sufficient to know IDs of all neighbors, distances to neighbors, and distances between adjacent neighbors
  - Every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

**UBG w/ distances:  $O(1)$  approximation in  $O(1)$  rounds**



## 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
  - $v \notin \text{MIS}, (u,v) \in E \rightarrow u \in \text{MIS}$
  - add  $v$  to MIS



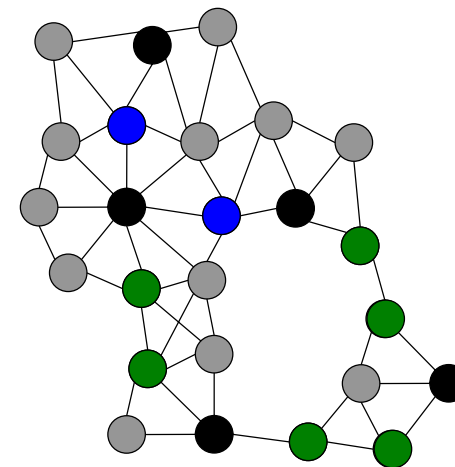
## Maximal Independent Set II

- Lemma:  
On bounded independence graphs:  $|\text{MIS}| \leq O(1) \cdot |\text{DS}_{\text{OPT}}|$
- Proof:
  1. Assign every MIS node to an adjacent node of  $\text{DS}_{\text{OPT}}$
  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  $\text{DS}_{\text{OPT}}$

$\rightarrow |\text{MIS}| \leq f(1) \cdot |\text{DS}_{\text{OPT}}|$
- Time to compute MIS on bounded-independence graphs:
  - Deterministically  $O(\log \Delta \cdot \log^* n)$
  - Randomized  $O(\log \log n \cdot \log^* n)$



## 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.**

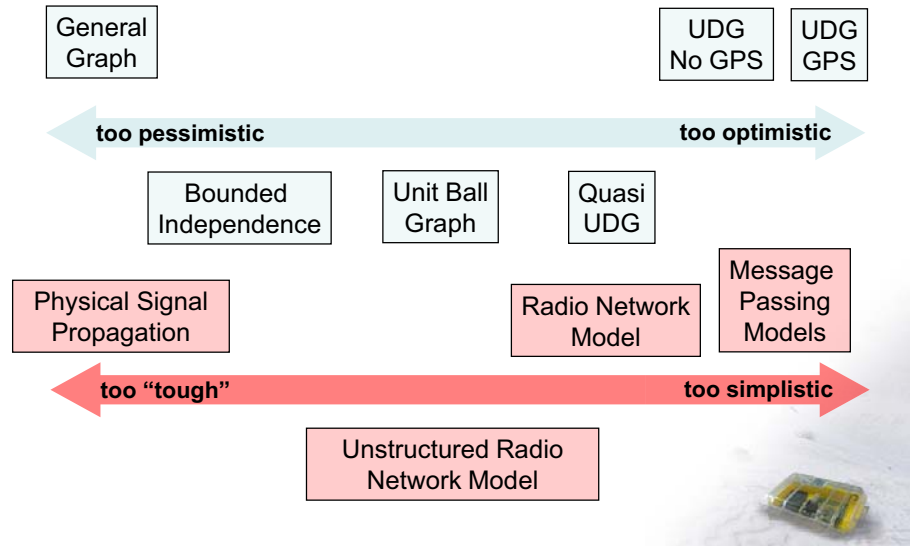
This gives a CDS!

**#2-hop connectors  $\leq f(2) \cdot |\text{MIS}|$**   
**#3-hop connectors  $\leq 2f(3) \cdot |\text{MIS}|$**

**$|\text{CDS}| = O(|\text{MIS}|)$**



## Models



Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/49

## 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 iff 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]

Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/50

## Unstructured Radio Network Model

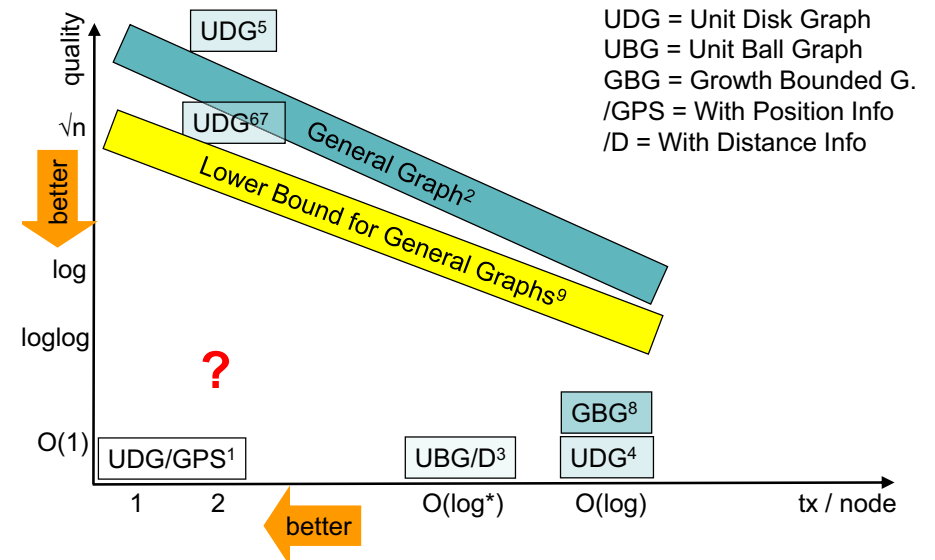
- Can MDS and MIS be solved efficiently in such a harsh model?

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

- And there is a matching lower bound.

Ad Hoc and Sensor Networks – Roger Wattenhofer – 7/51

## The model determines the complexity



## Open problem

- There is now a respectable body of research on clustering, and in particular dominating sets and connected dominating sets. However, in particular **locality** on special graphs is not yet fully understood:
- Let each node in a **unit disk graph** know its  $k$ -neighborhood for a **constant  $k$** , i.e., each node knows all nodes up to distance  $k$  including their interconnections. Given this information, each node must decide locally without any further communication whether it joins the dominating set or not. Is it possible to construct a valid dominating set that is only a **constant factor** larger than the optimal dominating set? (The best algorithms so far are the two MIS algorithms mentioned a few slides ago.)

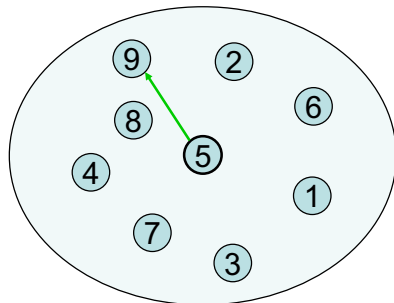


## BACKUP



## Election Algorithm

- Every node has a **random ID**
- Every node elects the neighbor with **highest ID** as its **leader**
- Every elected node joins the dominating set
  - First analyzed by Jie Gao et al. [Gao et al., SCG 2001]
  - Requires a single round of communication → lots of practical appeal

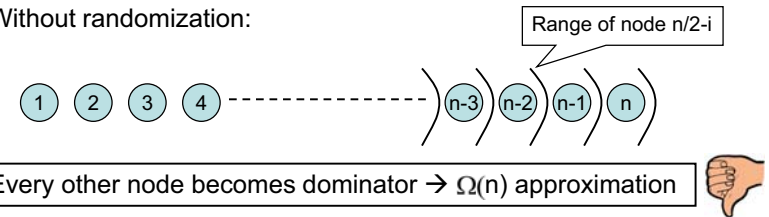


Is randomization required?  
Is this algorithms any good?

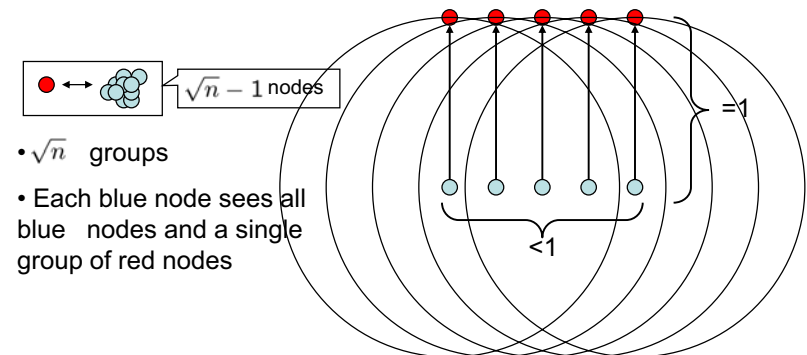


## Election Algorithm

- Without randomization:



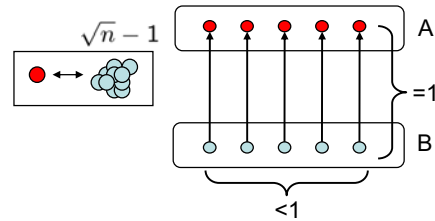
- With randomization:



- $\sqrt{n}$  groups
- Each blue node sees all blue nodes and a single group of red nodes

## Election Algorithm

• With randomization:



•  $\sqrt{n}$  groups

• Each blue node sees all blue nodes and a single group of red nodes

• Every blue node sees  $\sqrt{n}$  nodes in set B, and  $\sqrt{n} - 1$  nodes in set A

• The probability that the largest ID is in A is  $\frac{\sqrt{n} - 1}{2\sqrt{n} - 1} \approx \frac{1}{2}$

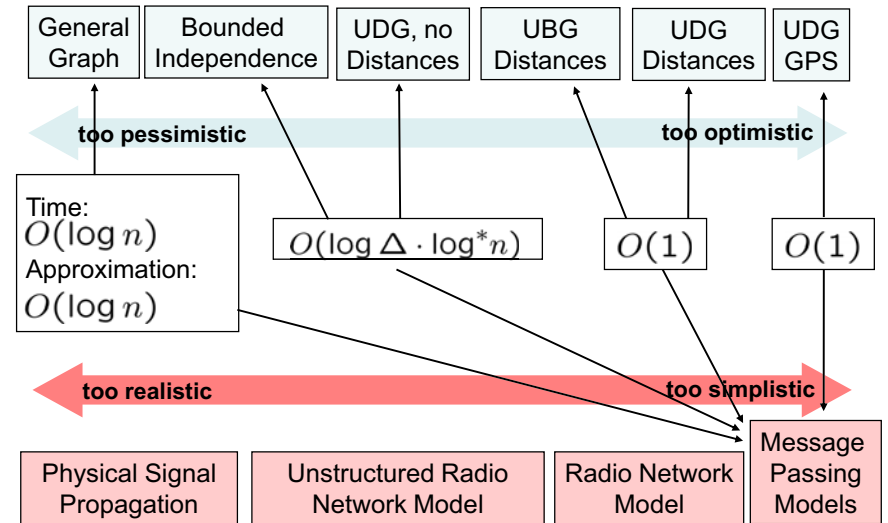
• By linearity of expectation, about  $\sqrt{n}/2$  blue nodes elect a node in A

Even with randomization  $\rightarrow \Omega(\sqrt{n})$  approximation



Many simple constant time DS algorithms have the same worst-case behavior, e.g. [Wu, Li, 99], [Dai, Wu, 04] etc...

## Summary Dominating Set I



## Summary Dominating Set II

