# G205 Fundamentals of Computer Engineering

CLASS 24, Mon. Dec. 6 2004

Stefano Basagni

Fall 2004

M-W, 1:30pm-3:10pm

# Maximum Independent Set (MIS)

- A subset $V' \subseteq V$ of the vertices of a graph G=(V,E) is independent when for each $u,v \in V'$ the edge $\{u,v\} \notin E$
- MIS is an Optimization Problem
- Input: A Graph G=(V,E) with n vertices
- Output: A subset V′ of V that is independent and has maximum size

# MIS: Hardness

- No known algorithm computer a MIS in polynomial time
- Need for approximate solutions
- And approximation algorithm is an algorithm that produces a solution that is not optimal, but that approximates it
- We sacrifice optimality in favor of a "good" solution that can be computed efficiently

# MIS is HARD to Approximate

- Bad news
  - Not only MIS is computationally hard
  - It is also hard to approximate:
    - Approximate solutions are not so good
    - They are "unboundedly" far from the optimum
- We consider the simple greedy heuristic for the MIS

# Greedy Heuristic for MIS, 1

- Select the vertex with minimum degree and put it in the MIS
  - The degree of a vertex is the number of its neighbors
    - Cardinality of its adjacency list
  - Keep going till all the vertices are either in the MIS of COVERED by a vertices in the MIS

# Greedy Heuristic for MIS, 2

```
MIS(V,E,d) // d is the vector of degrees
  mis = Ø
  while V ≠ Ø do
    v = vertex with min degree
    mis = mis ∪{v}
    V = V – {{v} ∪ N(v)}
  return mis
```

# Greedy MIS: Maximal Solution

- The greedy solution provides a maximal independent set
  - An independent set is maximal when, if you add a vertex, the set is no longer independent
    - You cannot make an maximal independent set bigger
- This solution is also a minimal dominating set
  - A dominating set $D \subseteq V$ is a set such that a vertex $v \in V$ is either in D or it has a neighbor in D

# Using Greedy MIS on UDG to Computer a DS

- Bad news: Still computationally hard
- Better news: Minimum DS It is approximable "up to a constant"
  - It means that the ratio between the size of a DS computed by MIS greedy on UDGs and the size of a MDS is < c, c a constant
- This constant is 5

# Greedy MIS for MDS on UDG is 5-approximable, 1

- Key fact: In a UDG disk (radius 1) there are at most 5 independent nodes
- Consider an Optimal solution and a Greedy solution
- Since Opt is dominant, it dominates Greedy
- Assign every vertex of Greedy to one dominator in Opt (choose one if more)

# Greedy MIS for MDS on UDG is 5-approximable, 2

- For each u in Opt consider is assigned vertices $v_1(u)$, $v_2(u)$, …, $v_k(u)$ of Greedy
- How big is k?
- Well, all $v_i(u)$ must be distant 1 from u and they also have to be independent
- Greedy: at most 5 times bigger than Opt

# MIS and Dominating Sets and Wireless Networks

- UDGs model ad hoc networks
- IS and DS are useful for clustering ad hoc networks
  - Gives the network a hierarchical organization
  - Decreases the amount of information at each node
  - Enhances scalability
  - Helps in "resource assignment"

# Two Protocols

- Distributed Clustering Algorithm (DCA)
  - Quasi-mobile networks, periodical re-clustering. Allow complexity analysis, fast and simple
- Distributed and Mobility-Adaptive Clustering (DMAC) Algorithm
  - Same rules/procedures for clustering set up and maintenance, adaptive to nodes mobility and node/link failures

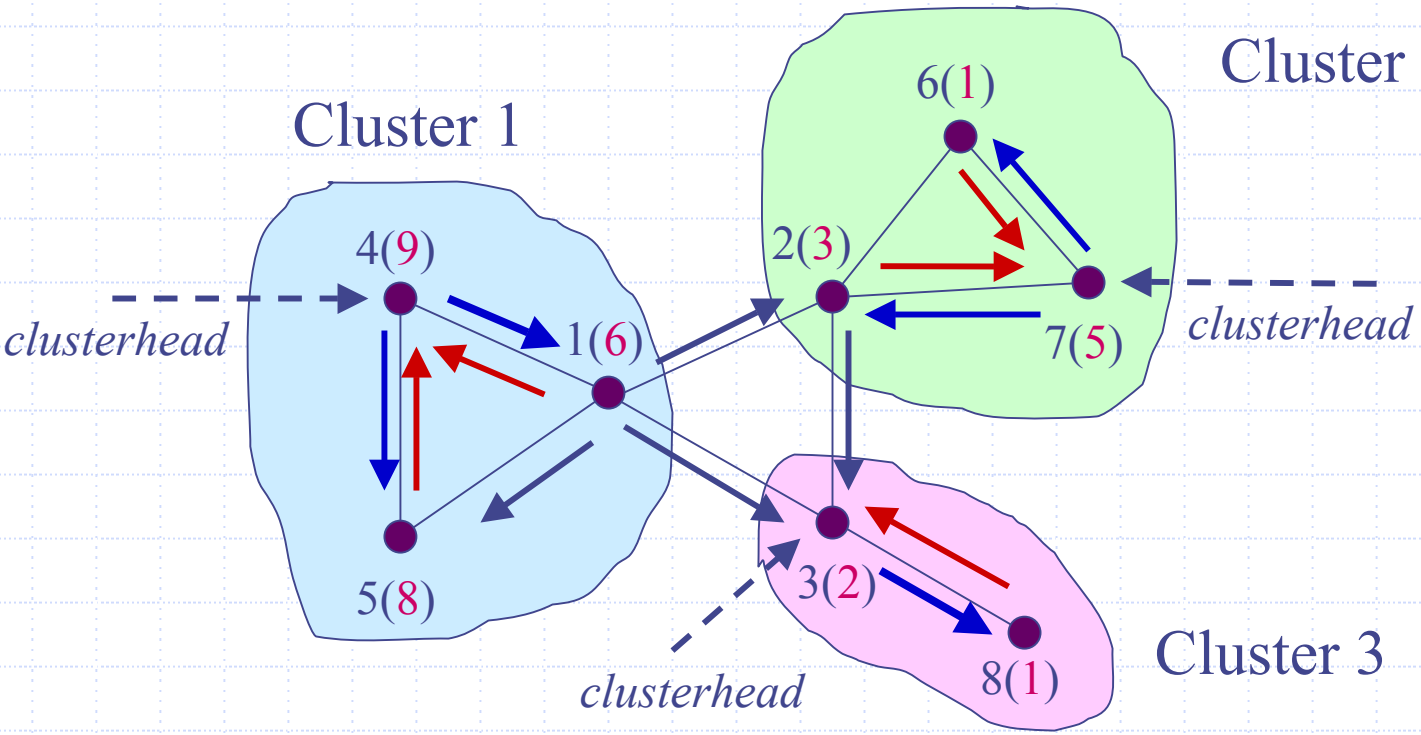
# DCA: Distributed Clustering Algorithm, 1

- Assumptions
  - Knowledge of IDs and weights of one-hop neighbors
  - Broadcast transmission of a packet in finite time (a "step")
  - Nodes do not move during clustering

# DCA, 2

- (Only) Two messages:
  - CH(v): Sent by a clusterhead v
  - JOIN(u,t): Sent by ordinary node u when it joins the cluster of clusterhead t
- Three (simple) procedures:
  - Init (start up)
  - OnReceivingCH(v), OnReceivingJOIN(u,v) (message triggered)

# Example




I Step II Step III Step IV Step V Step

# DCA: Provable Properties

- Consider

$$\tau: V \rightarrow \{1,2,3, \ldots , 2k\}$$

V = set of network nodes, k = number of clusters

- **Proposition**: Each node v in V sends exactly one message by $\tau(v)$ steps
- **Corollary 1**: DCA message complexity is n =|V|
- **Corollary 2**: DCA terminates correctly in at most 2k steps ( $\leq$ 2n)

# A Note on the Average Time Complexity

- We notice that

$$k \leq \alpha(G)$$

G = topology graph, $\alpha(G)$ = G's *stability number*

- We see the network as a *random graph*, for which

$$(2k \leq ) \; 2\,\alpha(G) = \text{circa } O(\log n)$$

Log's base is a function of n and the number of the network links

# Adapting to Mobility and Node/Link Failures: DMAC

- DMAC is for clustering set up AND maintenance
- Nodes can move during the clustering
- Each node reacts to
  - Reception of a message
  - Presence of a new link
  - Link failure
- Same assumptions of DCA, plus knowledge of neighbors' roles (no role = ordinary role)

# DMAC: The Procedures

- INIT
- Link-dependent procedures:
  - Link_Failure
  - New_Link
- Message-triggered procedures:
  - OnReceivingCH(v)
  - OnReceivingJOIN(u,t)

# Joining Clusterheads: Dynamic Backbone

- A theorem from Chlamtac and Farago:

  *If a network is connected, and DCA is used, then if and only if each clusterhead is linked to all the clusterheads at most three hops away, the resulting backbone network is connected*
- Inherently mobility adaptive and stateless
- Good if the random graph model could be used

# Dynamic Backbone: Some Simulation Results

- Networks with up to 2000 nodes (common parameters)
  - Number of Clusterheads < SQR(log n)
  - Number of Backbone Links < EXP(log n, 2.5) (when mapped on physical paths with at most three hops)
  - Number of Backbone links < EXP(log n, 2.2) (when links are obtained through directional antennas and/or power control)

# Some Applications

- Stateless multipoint communication: Routing, multicast and broadcast over the backbone
- Resource/user discovery
- Implementation of security in large networks of sensors
- Network management

# Clustering: Summary

- DCA+DMAC for clustering ad hoc networks:
  - Clusterhead selection based on mobility/node, dynamically changing parameters
  - Executed at each node with minimal topology knowledge
  - DCA time complexity is bound to a network parameter that depends on the network topology
  - DMAC is mobility/failure adaptive, fast and easy to implement

# Assignments

- Updated information on the class web page:

  www.ece.neu.edu/courses/eceg205/2004fa