Lecture 19

Broadcast routing

Eytan Modiano
Broadcast Routing

- Route a packet from a source to all nodes in the network

- Possible solutions:
  - Flooding: Each node sends packet on all outgoing links
    Discard packets received a second time
  - Spanning Tree Routing: Send packet along a tree that includes all of the nodes in the network
A graph $G = (N,A)$ is a finite nonempty set of nodes and a set of node pairs $A$ called arcs (or links or edges).

$N = \{1,2,3,4\}$
$A = \{(1,2),(2,3),(1,4),(2,4)\}$

$N = \{1,2,3\}$
$A = \{(1,2)\}$
Walks and paths

• A walk is a sequence of nodes \((n_1, n_2, ..., n_k)\) in which each adjacent node pair is an arc.

• A path is a walk with no repeated nodes.

Walk \((1,2,3,4,2)\)  Path \((1,2,3,4)\)
A cycle is a walk \((n_1, n_2, \ldots, n_k)\) with \(n_1 = n_k\), \(k > 3\), and with no repeated nodes except \(n_1 = n_k\).
A graph is connected if a path exists between each pair of nodes.

- An unconnected graph can be separated into two or more connected components.
Acyclic graphs and trees

- An acyclic graph is a graph with no cycles.
- A tree is an acyclic connected graph.

Acyclic, connected  unconnected  Cyclic, not tree

The number of arcs in a tree is always one less than the number of nodes.

- Proof: start with arbitrary node and each time you add an arc you add a node => N nodes and N-1 links. If you add an arc without adding a node, the arc must go to a node already in the tree and hence form a cycle.
Subgraphs

- $G' = (N', A')$ is a subgraph of $G = (N, A)$ if
  1) $G'$ is a graph
  2) $N'$ is a subset of $N$
  3) $A'$ is a subset of $A$

- One obtains a subgraph by deleting nodes and arcs from a graph
  - Note: arcs adjacent to a deleted node must also be deleted

Graph $G$  
Subgraph $G'$ of $G$
Spanning trees

- $T = (N', A')$ is a spanning tree of $G = (N, A)$ if
  - $T$ is a subgraph of $G$ with $N' = N$ and $T$ is a tree
Spanning trees

- Spanning trees are useful for disseminating and collecting control information in networks; they are sometimes useful for routing.

- To disseminate data from Node n:
  - Node n broadcasts data on all adjacent tree arcs
  - Other nodes relay data on other adjacent tree arcs

- To collect data at node n:
  - All leaves of tree (other than n) send data
  - Other nodes (other than n) wait to receive data on all but one adjacent arc, and then send received plus local data on remaining arc
General construction of a spanning tree

• Algorithm to construct a spanning tree for a connected graph $G = (N,A)$:

1) Select any node $n$ in $N$; $N' = \{n\}$; $A' = \{\}$

2) If $N' = N$, then stop ($T=(N',A')$ is a spanning tree)

3) Choose $(i,j) \in A$, $i \in N'$, $j \notin N'$

   $N' := N' \cup \{j\}$; $A' := A' \cup \{(i,j)\}$; go to step 2

• Connectedness of $G$ assures that an arc can be chosen in step 3 as long as $N' \neq N$

• Is spanning tree unique?
Spanning tree algorithm

- The algorithm never forms a cycle, since each new arc goes to a new node.

- $T = (N', A')$ is a tree at each step of the algorithm since $T$ is always connected, and each time we add an arc we also add a node.

- Theorem: If $G$ is a connected graph of $n$ nodes, then

  1) $G$ contains at least $n-1$ arcs
  2) $G$ contains a spanning tree
  3) if $G$ contains exactly $n-1$ arcs, $G$ is a spanning tree
Distributed algorithms to find spanning trees

1) A fixed node sends a "start" message on each adjacent arc of the graph

2) Each other node marks the first arc on which a start message was received as a spanning tree arc and then sends a "start" message on each other arc
   - This is a distributed implementation of the general spanning tree algorithm
   - It has several problems shared by many such algorithms:
     a) who chooses the starting node?
     b) When does the algorithm terminate?
     c) The resulting tree is somewhat random
Min weight spanning tree

- Given a graph with weights assigned to each arc, find a spanning tree of minimum total weight (MST)

- Define a "fragment" to be a subtree of a MST

- Theorem:
  - Given a fragment $F$ of an MST, let $a(i,j)$ be a minimum weight outgoing arc from $F$, where $j$ is not in $F$.
  - Then, $F$ extended by arc $a(i,j)$ & node $j$ is a fragment.

- Proof:
  - Let $M$ be the MST that does not include $a(i,j)$.
  - Since $a(i,j)$ is not part of $M$, then adding $a(i,j)$ to $M$ must cause a cycle. There must be some link in the cycle $b \neq a$ which is outgoing from $F$.
  - Deleting $b$ and adding $a$ creates a new spanning tree. Since weight of $b$ cannot be less then weight of $a$, $M'$ must be a MST.
    - If weight of $a = weight$ of $b$, then both are MST's otherwise $M$ could not have been an MST.
MST algorithms

• Generic MST algorithm steps:
  – Given a collection of subtrees of an MST (called fragments) add a minimum weight outgoing edge to some fragment

• Prim-Dijkstra: Start with an arbitrary single node as a fragment
  – Add minimum weight outgoing edge

• Kruskal: Start with each node as a fragment;
  – Add the minimum weight outgoing edge, minimized over all fragments
Prim-Dijkstra Algorithm

Step 1

Step 2

Step 3

Step 4

Step 5
Suppose the arcs of weight 1 and 3 are a fragment

- Consider any spanning tree using those arcs and the arc of weight 4, say, which is an outgoing arc from the fragment.
- Suppose that spanning tree does not use the arc of weight 2.
- Removing the arc of weight 4 and adding the arc of weight 2 yields another tree of smaller weight.
- Thus an outgoing arc of min weight from fragment must be in MST.