6.852 Lecture 10

- Minimum spanning tree
  - Gallager-Humblet-Spira algorithm
- Reading: Chapter 15.5, Gallager-Humblet-Spira paper

Minimum spanning tree

- Assume
  - undirected graph (i.e., bidirectional communication)
  - distinct edge weights
  - size and diameter unknown
  - can identify in- and out-edges to same neighbor
- Problem:
  - find minimum spanning tree
    - guaranteed to be unique
  - each node knows which of its edges is in tree
  - asynchronous wakeup (model as input action)
    - also wake up when message is received

Minimum spanning tree

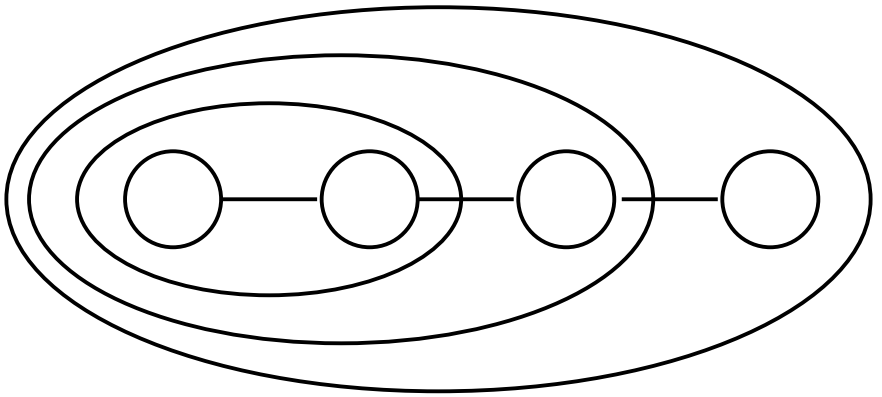
- Recall synchronous algorithm (SynchGHS)
  - proceeds in phases
    - maintain spanning forest (disconnected tree “fragments”)
      - each fragment has a leader
    - each fragment finds min weight outgoing edge (MWOE)
    - merge fragments using MWOEs to get fragments for next phase
      - determine new leader
  - in phase 0
    - each node initially in fragment by itself
    - use its min weight edge as MWOE
    - send “connect” across MWOE
    - leader of new fragment is adjacent to MWOE of two fragments
      - the one with the higher UID (don't really need UIDs though)

Minimum spanning tree

- Recall synchronous algorithm (SynchGHS)
  - in each phase after phase 0:
    - leader initiates search for MWOE (broadcast “initiate” via tree edges)
    - each node finds its MWOE
      - send “test” on potential edges, wait for “accept” or “reject”
      - test edges one at a time in order of weight to minimize messages
    - report results to leader (convergecast “report”)
      - remember direction of best edge
    - leader picks MWOE for fragment
      - send “change-root” to get there, then “connect” across MWOE
      - use remembered best edges
    - leader is adjacent to edge that is MWOE of two fragments
    - wait for phase to end

Minimum spanning tree

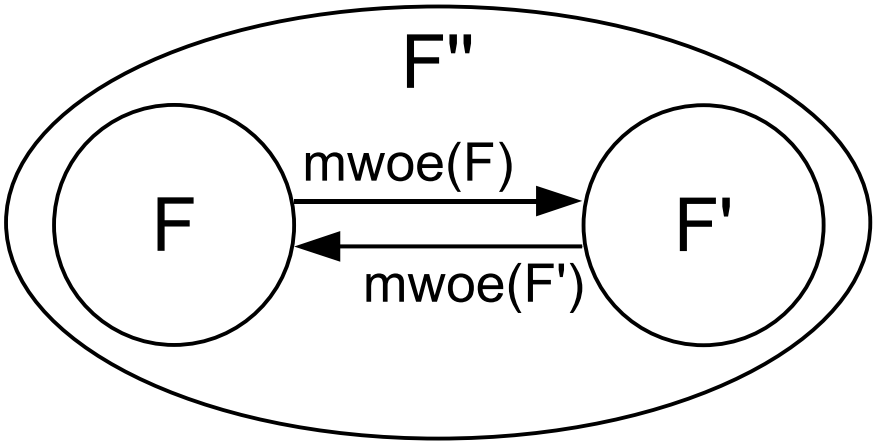
- Problems in translating to asynchronous setting
  - safety
    - determining outgoing edges (i.e., test-accept-reject protocol)
    - concurrent overlapping searches/convergecasts
    - merging fragments of different levels?
  - liveness
    - eventual termination?
  - complexity
    - O(log n) phases?



Minimum spanning tree

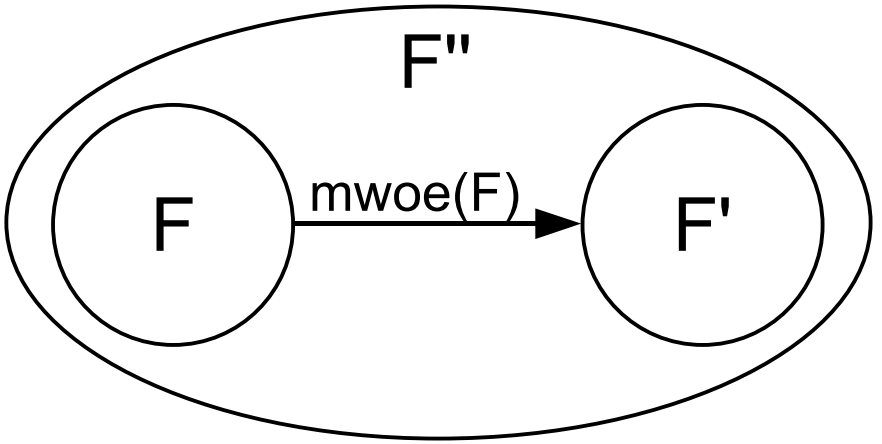
- Partially synchronize phases
  - maintain “level” (phase number) at each node
  - updated by initiate; sent with connect, test
  - don't respond to “test” with higher level
  - “absorb” lower level fragments trying to connect
    - fragments “merge” only if they share MWOE
  - skip reporting if too late (but propagate latest level)
- Rest is okay
  - termination
  - O(log n) phases
  - O(|E| + n log n) msg complexity

Minimum spanning tree



level(F) = level(F')
level(F") = level(F)+1

Merge



level(F) < level(F')
level(F") = level(F')

Absorb

F" not really a new fragment:
same core and level as F'

Minimum spanning tree

- Some tricky issues
  - might not search entire fragment
  - might skip levels
  - concurrent overlapping broadcasts of level
    - FIFO channels avoid need for check

Minimum spanning tree

- Determining minimum-weight outgoing edge
  - suppose fragment F has MWOE e to node n
  - recall: don't reply to test if level(n) < level(F)
  - if level(n) = level(F): just like synchronous GHS
  - if level(n) > level(F) then accept
    - fragment of n previously found outgoing edge e' lighter than e
    - all other outgoing edges of F of weight > weight(e) > weight(e')
    - so core of n's fragment at level(F) wasn't F's core

Minimum spanning tree

- Termination
  - we never delay progress of lowest level fragments (LLFs)
    - LLFs always eventually determine their MWOE
    - LLFs always eventually send “connect”
  - if MWOE of LLF is to higher level fragment
    - absorb when “connect” is processed
  - if MWOEs of all LLFs are to other LLFs
    - must be two LLFs that share MWOEs (why?)
    - so merge when “connect” is processed

Minimum spanning tree

- Complexity
  - msg: O(|E| + n log n)
    - 4|E| for test-reject msgs (one pair for each side of every edge)
    - n initiate msgs per level (broadcast: only sent on tree edges)
    - n report msgs per level (convergecast)
    - 2n test-accept msgs per level (one pair for each node)
    - n change-root/connect msgs per level (core to MWOE path)
    - log n levels
    - total: 4|E| + 5n log n
  - time: O(n log n (l + d)) if wakeup together

Minimum spanning tree

- GHS paper included informal arguments
  - convincing, but not formal
  - many successful attempts to formalize, but all complicated
    - lots of invariants because lots of variables, “subalgorithms”
    - some use simulation relations
    - recent proof by Moses and Shimony

Minimum spanning tree

- Optimizations
  - initial wakeup protocol (to get time bound)
  - don't test an edge that you rejected (in GHS)
  - don't require report if edge weight is greater than best so far
    - most likely for fragments being absorbed

Minimum spanning tree

- Applications
  - leader election
    - use variant of convergecast
- Optimal algorithms?
  - msg complexity:
    - $\Omega(|E|)$ if n unknown
    - $\Omega(n \log n)$ from leader election
  - time complexity:
    - trivial $\Omega(n)$; achieved by Awerbuch (1987)