CPSC 465/565 Theory of Distributed Systems

James Aspnes

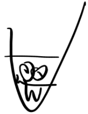
2023-09-01

Today's exciting topics

- Message passing complexity measures
- Broadcast and convergecast

Message passing model

- Buffer b_ij holds messages from p_i to p_j



- Execution: $C_0\alpha_1 C_1\alpha_2 C_2 \ldots$
  - $\alpha_t =$ delivery event at step t
- Nondeterminism: adversary chooses next event from enabled events
- Fairness: Every message sent is eventually received

What about complexity?

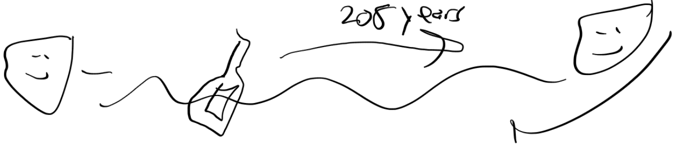
Complexity measures for message-passing

Last time:

- Local computation: free!
  - Justification: usually fast relative to message delivery times
- Message complexity = number of messages sent
- Bit complexity = total size of all messages

This time:

- Time complexity = time to finish
  - But: even in fair executions, no upper bound on delivery time



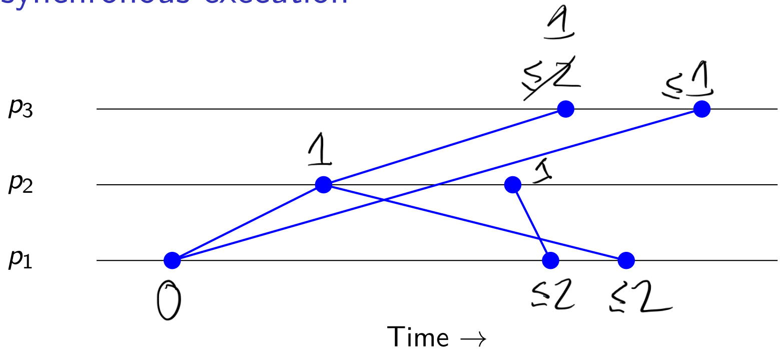
Time complexity without time bounds

- *Define* maximum message delay = 1 time unit.
- *Define* gap between two local computation events = 0 time units.
- Assign each event the max time consistent with these assumptions.

This gives a time measure that is agnostic about actual time delays.

(Similar to assumption behind $O()$ notation.)

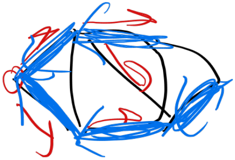
An asynchronous execution



Assign highest possible time to each event consistent with:

- $T(\text{first step}) = 0$.
- $T(\text{receive}) \leq T((\mathit{send})) + 1$.
- Consecutive local computation events have same time.

Example: flooding with parent pointers



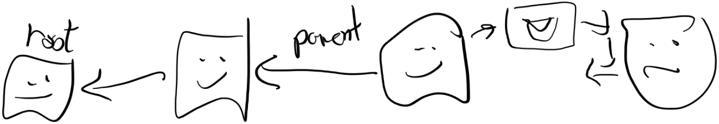
```
initially do
    if pid = root then
        parent ← root
        send m to all neighbors
    else
        parent ← ⊥

upon receiving m from j do
    if parent = ⊥ then
        parent ← p

        send m to all neighbors
```

Builds a spanning tree with broadcast as side-effect.

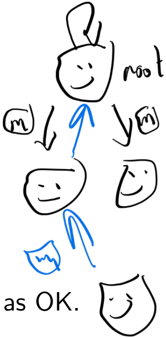
Correctness



Claim: eventually parent pointers form rooted spanning tree.

- Safety:
  1. $\text{parent}_i = \perp$ or parent_i is node on path to root
  2. If $m \in b_{ij}$, then $\text{parent}_i \neq \perp$.
- Liveness: Eventually $\text{parent}_i \neq \perp$.

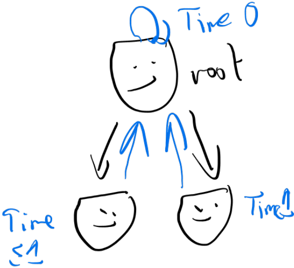
Safety proof



1. parent$_i = \perp$ or parent$_i$ is node on path to root
2. If $m \in b_{ij}$, then parent$_i \neq \perp$.

- Check initial configuration:
  1. parent$_i = \perp$ unless $i =$ root; count parent$_{\text{root}} =$ root as OK.
  2. If $m \in b_{ij}$, then $i =$ root and parent$_{\text{root}} \neq \perp$.
- Check each claim against steps that could make it false:
  1. If i sets parent$_i$ to j:
     - i receives m from j
     - $m \in b_{ij} \Rightarrow$ parent$_j \neq \perp \Rightarrow$ parent$_j$ is on path to root
  2. If i sends m:
     - Same code sets parent$_i \neq \perp$

Liveness proof



Eventually $\text{parent}_i \neq \perp$.

Same induction on distance as simple flooding.

But now we will include time.

- Claim: For all i, $\text{parent}_i \neq \perp$ by time $d(i, \text{root})$.
- Proof: By induction on $d(i, \text{root})$:
  - Base: $d(i, \text{root}) = 0 \Rightarrow i = \text{root}$, so $\text{parent}_i = i$ at time 0.
  - Induction step: Suppose $d(i, \text{root}) = d + 1$
    - Then there is some neighbor j that sets $\text{parent}_j \neq \perp$ by time d.
    - When j does this, it sends m to i.
    - At most one time unit later ($\leq d + 1$), i receives m.
    - When i receives m, sets $\text{parent}_i \neq \perp$ if not already set.

Time complexity

D = diameter
= max dist
between 2 nodes

Count until last message delivered.

- Most distant node is at $\max_i d(i, \text{root}) \leq \text{diam}(G)$.
- Every node sets $\text{parent}_i \neq \perp$ by time D.
- Last message is sent by time D.
- Last message is received by time $D + 1$.

So time complexity is $D + 1$.

Detecting termination

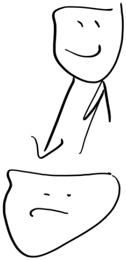
Add acknowledgments:

upon receiving m **from** j **do**
    if parent $= \perp$ **then**
        parent $\leftarrow j$
        send m to all neighbors
    else
        send nack to j

upon receiving ack **from** j **do**
    children $\leftarrow$ children $\cup \{j\}$

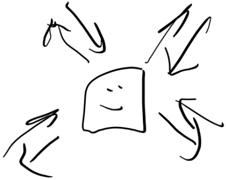
upon receiving nack **from** j **do**
    nonChildren $\leftarrow$ nonChildren $\cup \{j\}$

as soon as children $\cup$ nonChildren $=$ neighbors **do**
    send ack to parent



Done when root gets ack or nack from all neighbors.

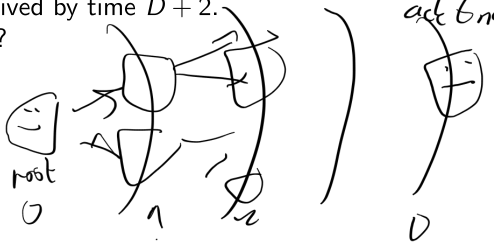
Complexity of the terminating version



- Message complexity
  - Every edge gets m in each direction
  - Plus ack or nack in response
  - Total is $4|E|$ messages.
- Time complexity
  - Outgoing messages reach everybody in time D.
  - All nacks are sent by time $D+1$.
  - All nacks are received by time $D+2$.
  - What about acks?

$2|E|$ receiving msgs

$+2|E|$ ack & nacks



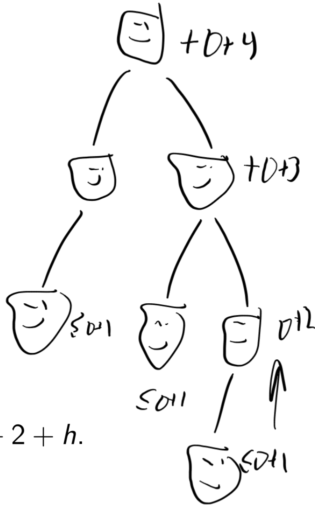
Timing of acks

Look at position of each node in the tree:

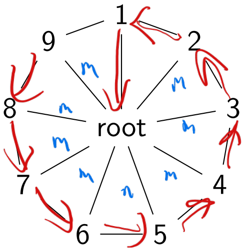
- If I am a leaf:
  - I send ack once I collect all nacks
  - This happens by time $D + 2$
- If I am a parent:
  - I have to wait for all my kids.

Induction hypothesis:

- Node at height h sends ack at time $\leq D + 2 + h$.
- $\Rightarrow$ finish by time $D + 2 +$ (height of tree).



How tall can the tree get?



root
1
1
1
2
1
3
1

depth $= n-1$

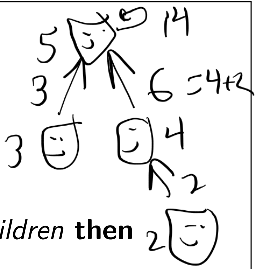
- Max distance from root is 1, so all nodes recruited by time 1.
- But maybe some nodes are recruited faster.
- Worst case: height of tree $= n - 1$.
- Gives $D + 2 + (n - 1)$ worst-case time for broadcast with acks.

Convergecast

Sometimes it's convenient to collect data from an existing tree:

```
initially do
    if I am a leaf then
        send input to parent

upon receiving v_c from c do
    append (c, v_c) to buffer
    if buffer contains messages from all my children then
        v ← f(buffer, input)
        if pid = root then
            return v
        else
            send v to parent
```

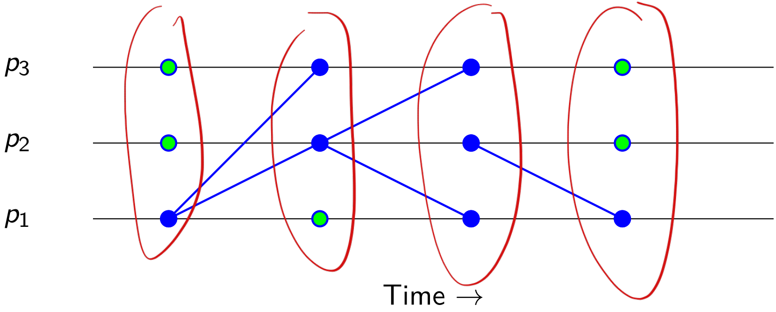


Broadcast to build the tree and **convergecast** to collect data.

Performance of convergecast

- Message complexity $= n - 1$
- Time complexity:
  - Leaves send at time 0
  - Parent sends by max child time $+1$
  - $\Rightarrow$ each i sends by height(i)
  - $O(n)$ time in worst case.

Synchronous model

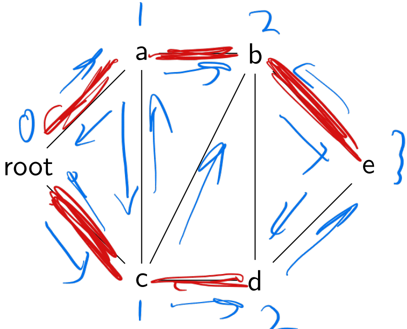


Assumption: every message delay $= 1$ exactly.

Formal version:

- Sequence of rounds $r = 0, 1, 2, \ldots$
- Every node gets one delivery event per round.
- Every message sent in r is received in $r + 1$.

Synchronous broadcast gives shortest-path tree



Proof: By induction, i receives m exactly at time $d(\text{root}, i)$.

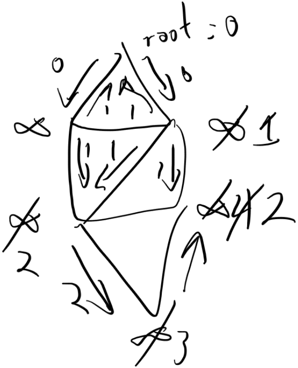
~~\end{center}~~

What if we don't have synchrony?

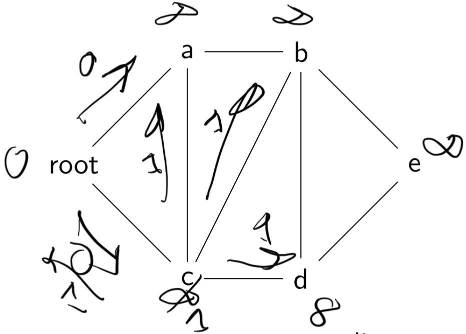
Attempt 1: Use Dijkstra's algorithm.

```
initially do
    if pid = initiator then
        dist ← 0
        send dist to all neighbors
    else
        dist ← ∞

upon receiving d from p do
    if d + 1 < dist then
        dist ← d + 1
        parent ← p
        send dist to all neighbors
```



Dijsktra in action



Claim: Every process converges to correct distance.

- Safety: $\text{dist}_i \geq d(i, \text{root})$ and for any $d \in b_{ij}$, $d \geq d(i, \text{root})$.
- Liveness: $\text{dist}_i = d(i, \text{root})$ no later than time $d(i, \text{root})$.

Ind on d: Base: true for d=0

Ind step: closer nbr at dist d sets by time d

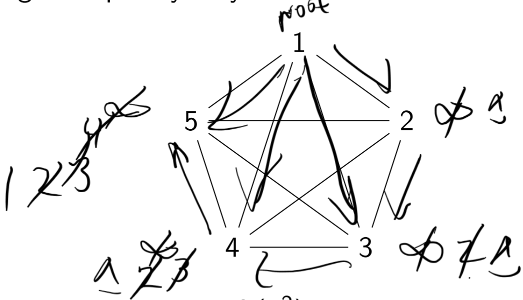
⇒ msg ⇒ I set to d+1 by time d+1.

Worst-case complexity

diameter

Liveness proof gives $O(D)$ time.

But message complexity may be bad:



- Careful schedule yields $\Theta(n^2)$ updates.
- $\Rightarrow \Theta(n^3)$ messages.
- With synchrony, only $2|E| = O(n^2)$ messages.

Fix by simulating synchrony

(α synchronizer)

initially do

  if pid $=$ root **then**

    dist $\leftarrow 0$

    send $[= 0]$ to all neighbors

  else

    send $[\geq 0]$ to all neighbors

as soon as *I received* $[\geq d]$ *from all neighbors* **do**

  send $[\geq d + 1]$ to all neighbors

as soon as *I received* $[= d]$ *from at least one neighbor and* $[\geq d]$ *from the others* **do**

  dist $\leftarrow d + 1$

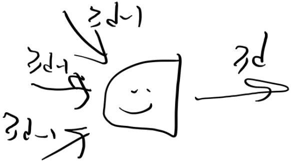
  send $[= d + 1]$ to all neighbors

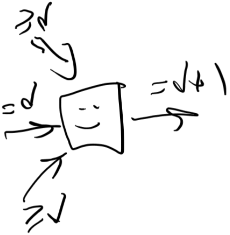
  halt

$[= d] \to$ I am exactly d

$[\geq d] \to$ I am at least d

Correctness





- Safety:
  - ✓ If i sends $[\geq d]$, $d(i, \text{root}) \geq d$.
  - If i sends $[= d]$, $d(i, \text{root}) = d$.

+ similar constraints on msgs.

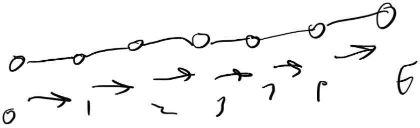
- Liveness:
  - If $d(i, \text{root}) = d$, then
    - For each $d' < d$, i sends $[\geq d']$ by time d'.
    - i sends $[= d]$ by time d.

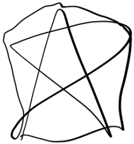
Easy induction on d.
(rhymes w/ lazy)

Cost

- From safety property:
  - Distance-d node sends only $[\geq 0], [\geq 1], \ldots, [\geq d-1], [= d]$.
  - This gives at most $D+1$ messages per direction per edge.
  - $\Rightarrow$ Message complexity $= O(D|E|)$.
- Liveness property:
  - Most distant node sends $[= d]$ by time d.
  - $\Rightarrow$ Time complexity $= O(D)$.

For example, get $O(n^2)$ messages and $O(1)$ time on K_n.





$D = 1$

$O(Dn) = O(n^2)$

Next time

Leader election!