Information Spreading in Dynamic Networks

On the Power of Forwarding Algorithms

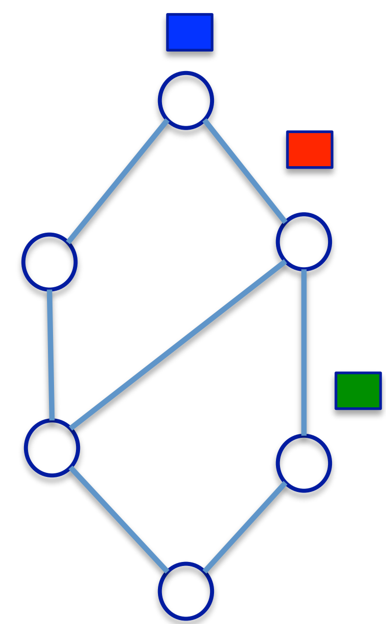
Rajmohan Rajaraman

July 29, 2013 @ Fields Institute

Chinmoy Dutta, Zhifeng Sun, Emanuele Viola (Northeastern)
Gopal Pandurangan (NTU, Singapore; Brown)

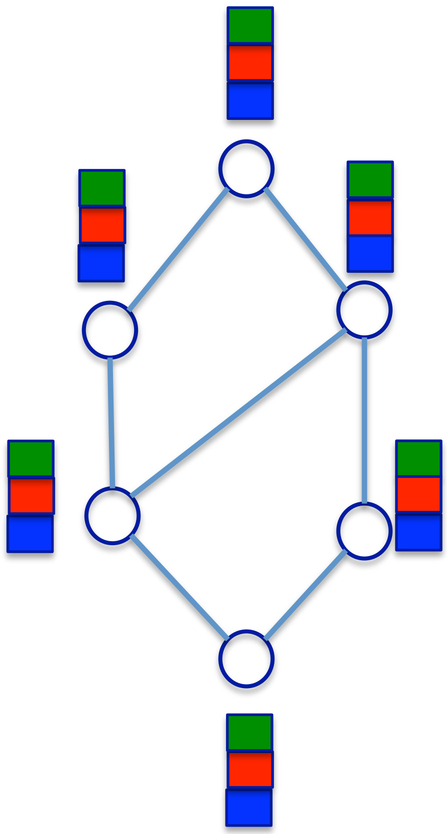
Information Spreading

- There are k distinct data items (tokens) in a network
- Goal: A copy of each token is communicated to each node
- Communication model:
  - Tokens can be stored, copied, and forwarded
  - At most one token or a small message (O(log n) size) can be sent across an edge in one round
- Question: How many rounds do we need?



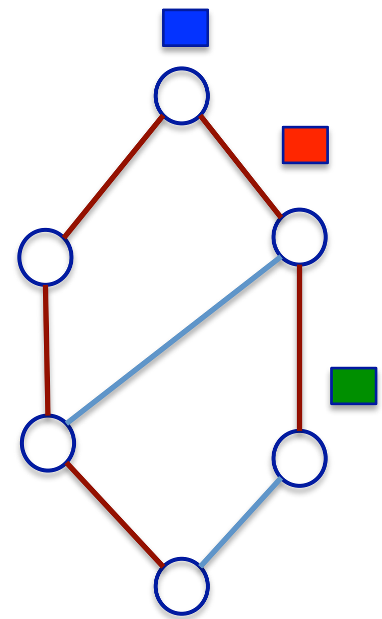
Information Spreading

- There are k distinct data items (tokens) in a network
- Goal: A copy of each token is communicated to each node
- Communication model:
  - Tokens can be stored, copied, and forwarded
  - At most one token or a small message ($O(\log n)$ size) can be sent across an edge in one round
- Question: How many rounds do we need?



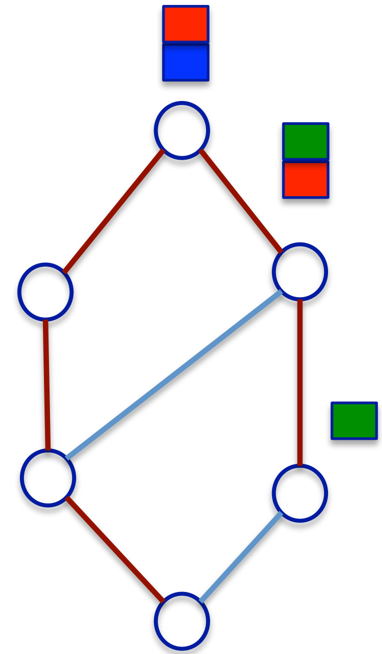
Spreading in Static Networks

- Idea: Have each token reach a new node in each round
  - Not achievable owing to bandwidth constraint
- Carefully schedule the token transmissions
- Spanning tree algorithm:
  - Pipeline tokens up the tree
  - Pipeline tokens down the tree
- Completed in O(n+k) rounds



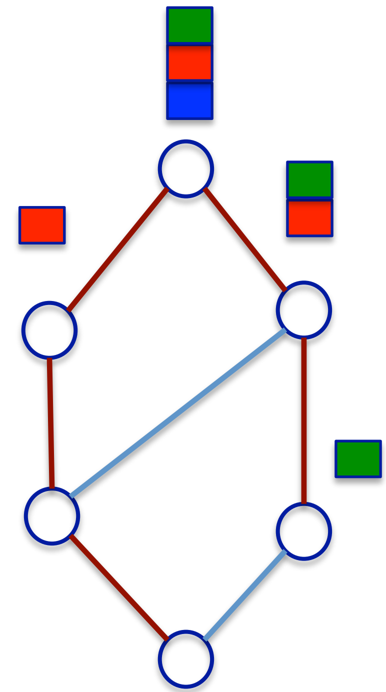
Spreading in Static Networks

- Idea: Have each token reach a new node in each round
  - Not achievable owing to bandwidth constraint
- Carefully schedule the token transmissions
- Spanning tree algorithm:
  - Pipeline tokens up the tree
  - Pipeline tokens down the tree
- Completed in O(n+k) rounds



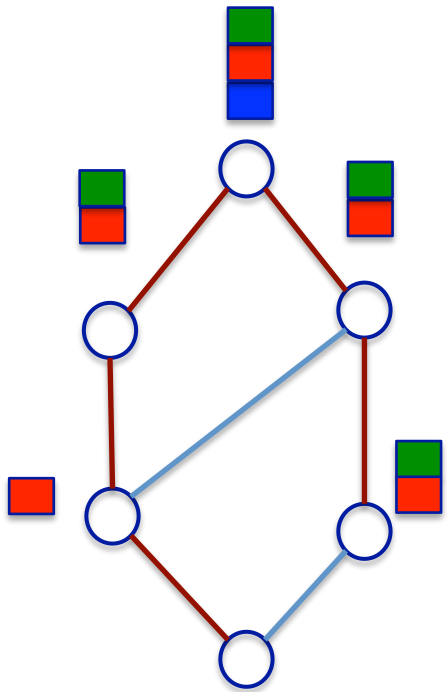
Spreading in Static Networks

- Idea: Have each token reach a new node in each round
  - Not achievable owing to bandwidth constraint
- Carefully schedule the token transmissions
- Spanning tree algorithm:
  - Pipeline tokens up the tree
  - Pipeline tokens down the tree
- Completed in O(n+k) rounds



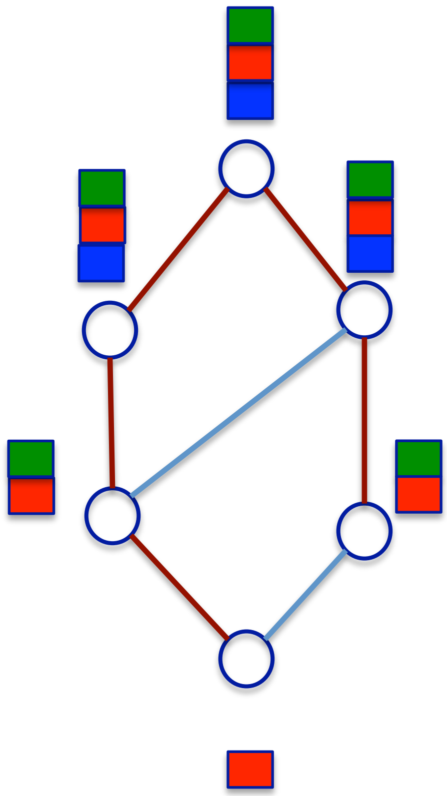
Spreading in Static Networks

- Idea: Have each token reach a new node in each round
  - Not achievable owing to bandwidth constraint
- Carefully schedule the token transmissions
- Spanning tree algorithm:
  - Pipeline tokens up the tree
  - Pipeline tokens down the tree
- Completed in O(n+k) rounds



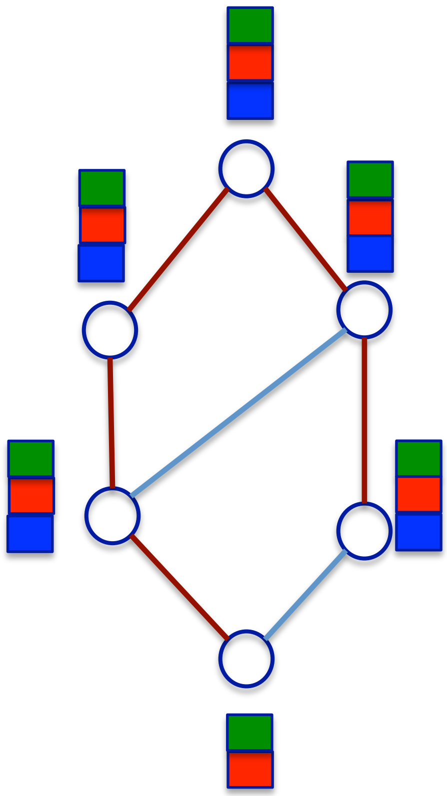
Spreading in Static Networks

- Idea: Have each token reach a new node in each round
  - Not achievable owing to bandwidth constraint
- Carefully schedule the token transmissions
- Spanning tree algorithm:
  - Pipeline tokens up the tree
  - Pipeline tokens down the tree
- Completed in O(n+k) rounds



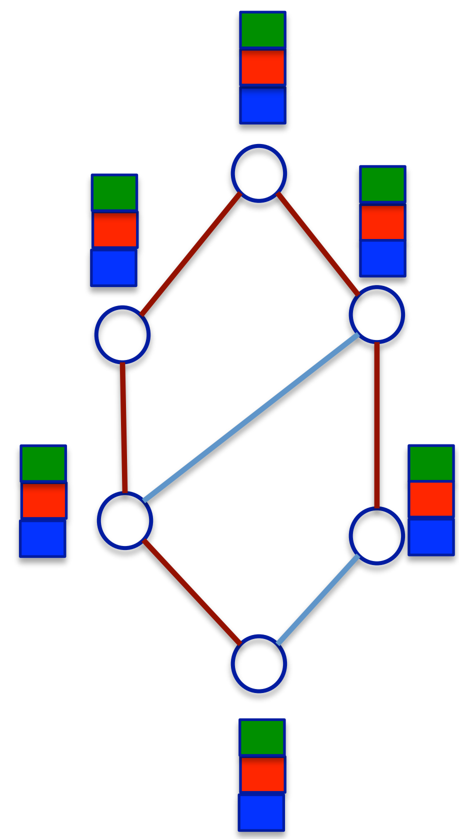
Spreading in Static Networks

- Idea: Have each token reach a new node in each round
  - Not achievable owing to bandwidth constraint
- Carefully schedule the token transmissions
- Spanning tree algorithm:
  - Pipeline tokens up the tree
  - Pipeline tokens down the tree
- Completed in O(n+k) rounds



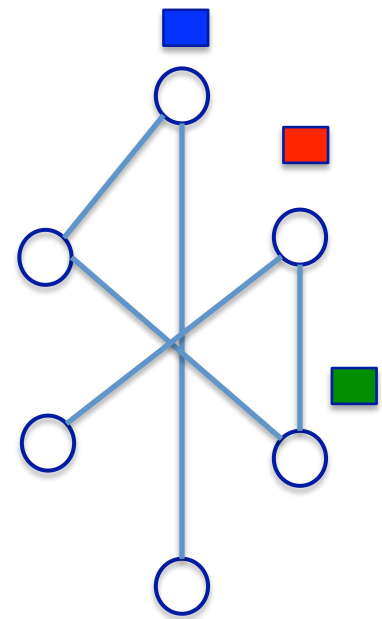
Spreading in Static Networks

- Idea: Have each token reach a new node in each round
  - Not achievable owing to bandwidth constraint
- Carefully schedule the token transmissions
- Spanning tree algorithm:
  - Pipeline tokens up the tree
  - Pipeline tokens down the tree
- Completed in O(n+k) rounds



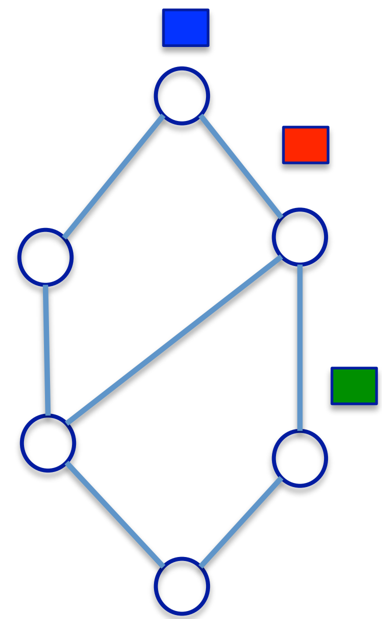
Spreading in Dynamic Networks

- What can be done when the network is dynamic?
  - Nodes are fixed, but edges change dynamically
- Central question:
  - Can the k-gossip problem be completed in O(n+k) rounds?
- Answer will depend on model:
  - Power of adversary in control of network dynamics
  - Communication model



Spreading in Dynamic Networks

- What can be done when the network is dynamic?
  - Nodes are fixed, but edges change dynamically
- Central question:
  - Can the k-gossip problem be completed in O(n+k) rounds?
- Answer will depend on model:
  - Power of adversary in control of network dynamics
  - Communication model



Motivation for Study

- Many networks are inherently dynamic
  - Links and link quality can change with node mobility and communication environment
- Also important for static networks:
  - Large-scale parallel interconnects and distributed networks
  - Applications do not run in isolation
  - Information spreading task needs to complete in a network carrying other traffic
  - Impact of competing unpredictable traffic can be modeled by a dynamic network
- Why an adversarial model?
  - Lower bounds explore the limits of what can be achieved
  - Upper bounds yield very strong guarantees

Computing over Dynamic Networks

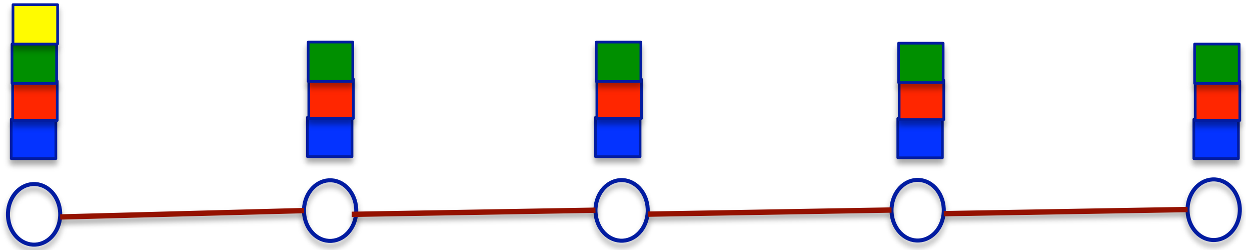
- Self-stabilization [Dijkstra 74, …, Dolev 00, …]
  - Convergence to steady state from arbitrary initial state in the absence of dynamics
- Load balancing [Aiello-Awerbuch-Maggs-Rao 93, …]
  - Balance an initial arbitrary distribution of tokens
- Packet routing and multi-commodity flow
  - The Slide protocol [Awerbuch-Mansour-Shavit 89, Afek-Gafni-Rosen 92, …]
  - Local balancing [Awerbuch-Leighton 93,94, …]
- Random walks [Avin-Koucky-Lotker 08]
- Information dissemination [Kuhn-Lynch-Oshman 10]

Information Spreading: k-Gossip

- Initially, k tokens distributed among a subset of nodes
- Goal: Disseminate the tokens to every node in the network as quickly as possible
- Applications:
  - Counting the number of nodes
  - All-to-all communication
  - Primitive for more complex distributed computing

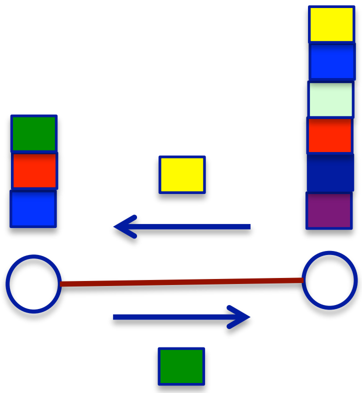
Quest for a Simple Local Protocol

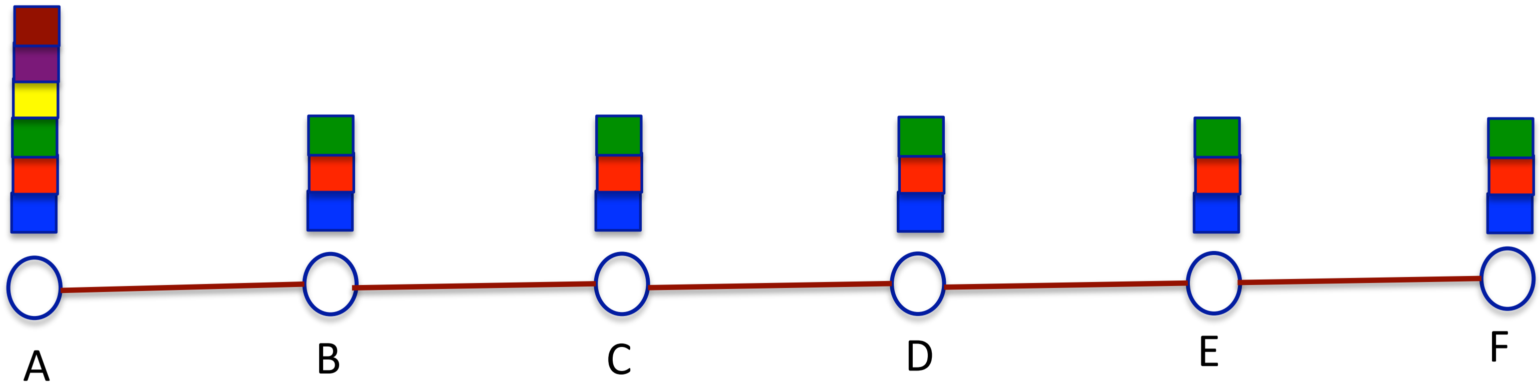
- Let S_u be the set of tokens at u
- Take 1: RANDOM
  - Node u broadcasts a token chosen uniformly at random from S_u
- The Good: Can be efficiently implemented in any adversarial network model
  - Token transmitted is independent of neighborhood
- The Bad: Requires $\Omega(n^2)$ rounds in the worst-case even for static networks
  - Progress along an edge requires expected $\Omega(n)$ rounds



Quest for a Good Local Protocol

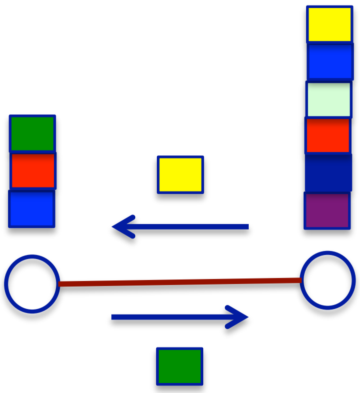
- Let S_u be the set of tokens at u
- Take 2: DIFF
  - Along edge (u,v), node u sends an arbitrary token chosen from $S_u - S_v$
- The Good: Completes in O(n) rounds in any static network
  - Analysis by a delay sequence argument
- The Bad: May need $\Omega(n^2)$ rounds under adversarial dynamics

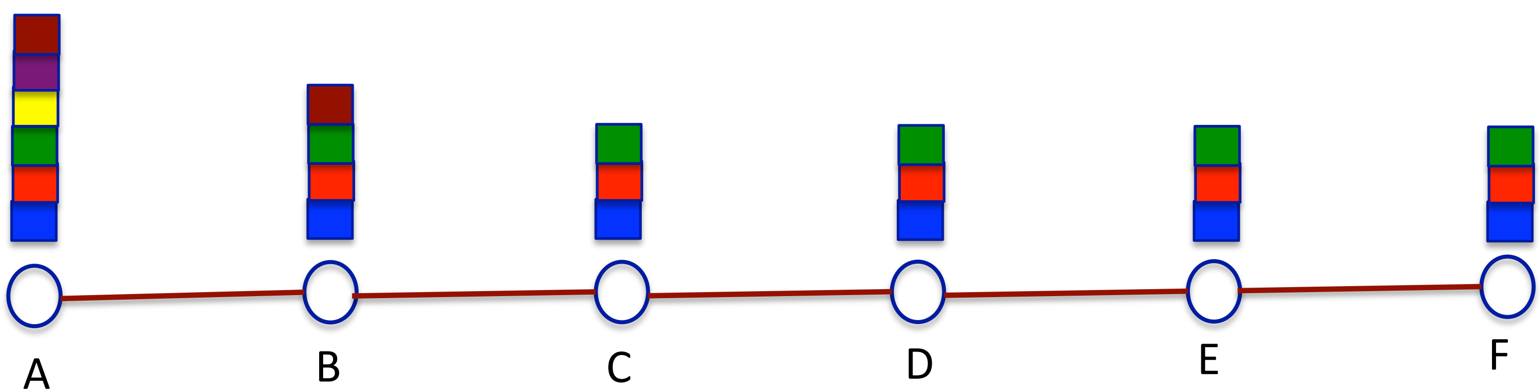




Quest for a Good Local Protocol

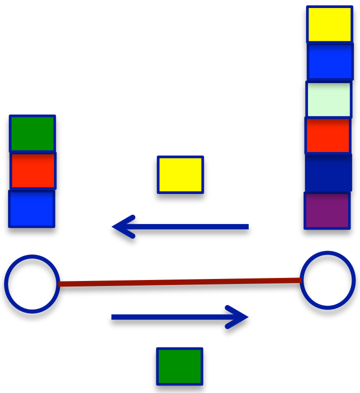
- Let S_u be the set of tokens at u
- Take 2: DIFF
  - Along edge (u,v), node u sends an arbitrary token chosen from $S_u - S_v$
- The Good: Completes in O(n) rounds in any static network
  - Analysis by a delay sequence argument
- The Bad: May need $\Omega(n^2)$ rounds under adversarial dynamics

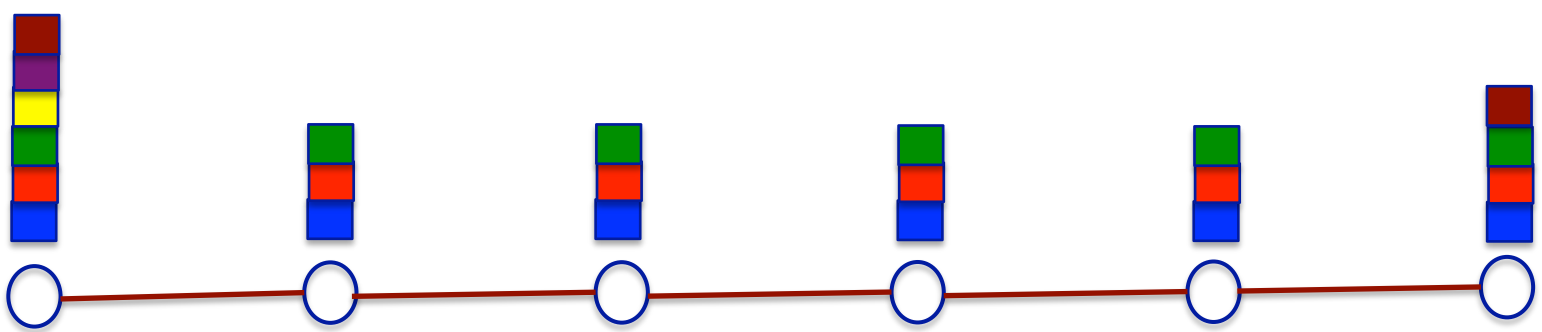




Quest for a Good Local Protocol

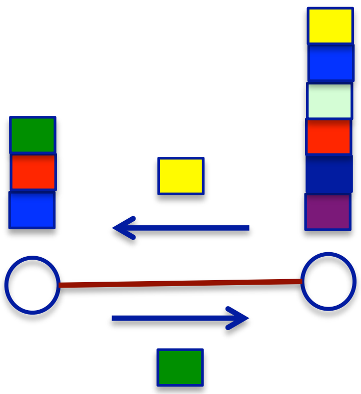
- Let S_u be the set of tokens at u
- Take 2: DIFF
  - Along edge (u,v), node u sends an arbitrary token chosen from $S_u - S_v$
- The Good: Completes in O(n) rounds in any static network
  - Analysis by a delay sequence argument
- The Bad: May need $\Omega(n^2)$ rounds under adversarial dynamics





Quest for a Good Local Protocol

- Take 3: RAND-DIFF
  - Along edge (u,v), node u sends a token chosen uniformly at random from $S_u - S_v$
  - A weak adversary cannot set up worst-case edges
- The Good: Certainly beats the preceding lower bound example
- The Bad: Cannot be implemented efficiently
  - Determining whether the set difference is nonempty requires Ω(n) bits on communication complexity [Kalyanasundaram-Schnitger 92, Razbarov 92]
  - Even if network is relatively static, too many rounds of communication
  - Also applies to DIFF



Network and Protocol Models

- Synchronous network model
  - Computing progresses in synchronous rounds
- Adversarial model:
  - Strong adaptive adversary: In round r, in parallel
    - An adversary presents connected network G_r
    - Each node decides what message to broadcast
  - Weak adaptive adversary:
    - Network may remain static for O(log(n)) rounds
    - Each node knows its neighbors
  - Oblivious adversary:
    - Adversary does not know the algorithm's moves
- Types of algorithms:
  - Forwarding: Do not manipulate tokens in any way other than store, copy, and forward them
  - General: The messages may be arbitrary, and nodes can construct tokens by processing multiple messages

Previous Results

- Token-forwarding against a strong adaptive adversary [Kuhn-Lynch-Oshman 10]:
  - Every deterministic forwarding algorithm requires $\Omega(n\log(n))$ rounds
  - $\Omega(n^2)$ lower bound for a restricted class of forwarding algorithm
- Network coding against a strong adaptive adversary [Haeupler 11, Haeupler-Karger 11]:
  - $O(n)$ rounds whp for token/message size >= $n\log(n)$
  - $O(n^2/\log(n))$ rounds whp for token/message size >= $\log(n)$
  - Can be made deterministic for larger message sizes
  - Centralized will allow smaller message sizes

This Talk

- Every online token-forwarding algorithm for k-gossip needs $\Omega(nk/\log(n))$ rounds under a strong adaptive adversary
  - For large token/message sizes, establishes an $\Omega(n/\log(n))$ gap between token-forwarding and network coding
  - Applies even to well-mixed distributions
- Can we break this "quadratic" barrier under weaker adversary models?
  - A variant of RAND-DIFF that completes in $O(n \text{ polylog}(n))$ rounds, starting from a well-mixed distribution, against a weak adaptive adversary
- Open problems, offline model, …

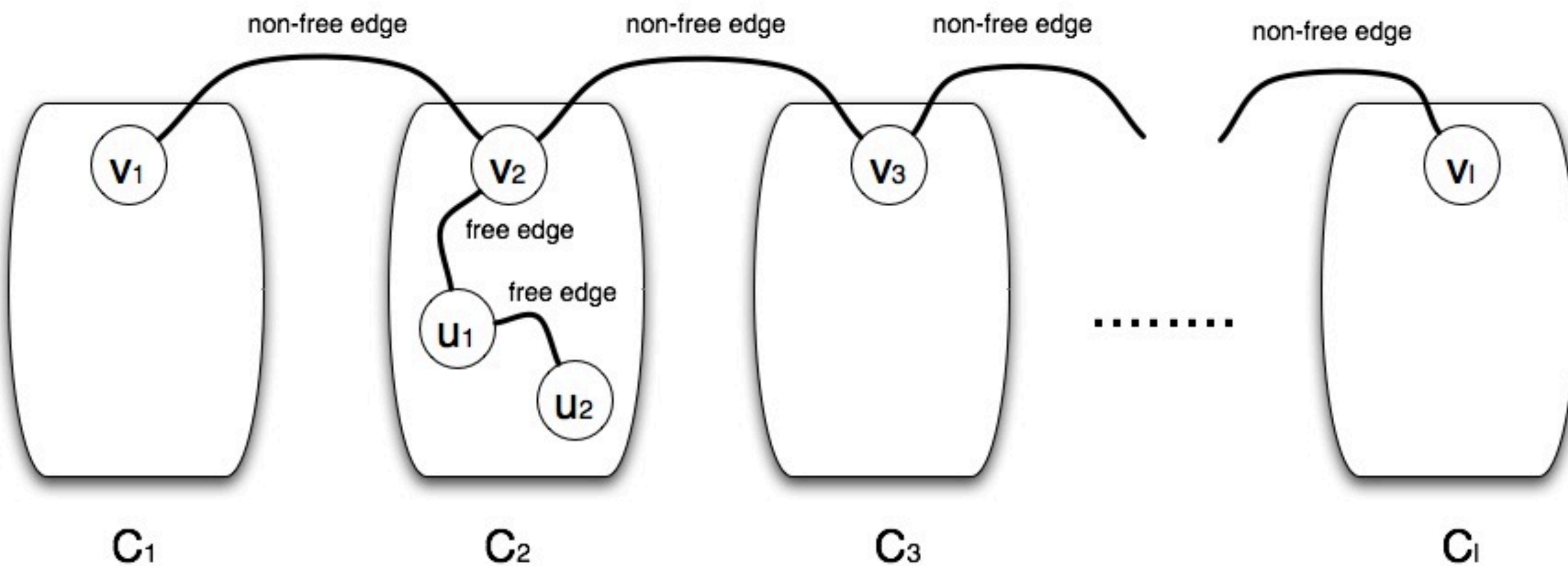
Lower Bound: Basic Setup

- Initial distribution:
  - Every node has a distinct token (k = n)
- Focus on centralized deterministic algorithms
  - In each round, the algorithm selects for each node a token to broadcast
  - The algorithm can choose to broadcast other information as well
  - Then, adversary selects the network
  - Strong adversary model

Lower Bound: Free Edges

- In a round r, call an edge (u,v) free if at the start of round r, u has the token that v broadcasts, and vice versa
- In each round, adversary can add free edges without any cost
  - No useful token exchange along a free edge
- The free edges may not form a connected network

Lower Bound: Free and Non-Free Edges



To ensure connectivity, the adversary needs to connect the connected components with non-free edges

Lower Bound: Useful Token Exchanges

- Consider the graph induced by the free edges
- The adversary selects one node from each component and connects them in a line
- The number of useful token exchanges is at most twice the number of connected components
- How do we bound the number of connected components?
  - Depends on the tokens being broadcast
  - As the computation proceeds, difficult to keep track

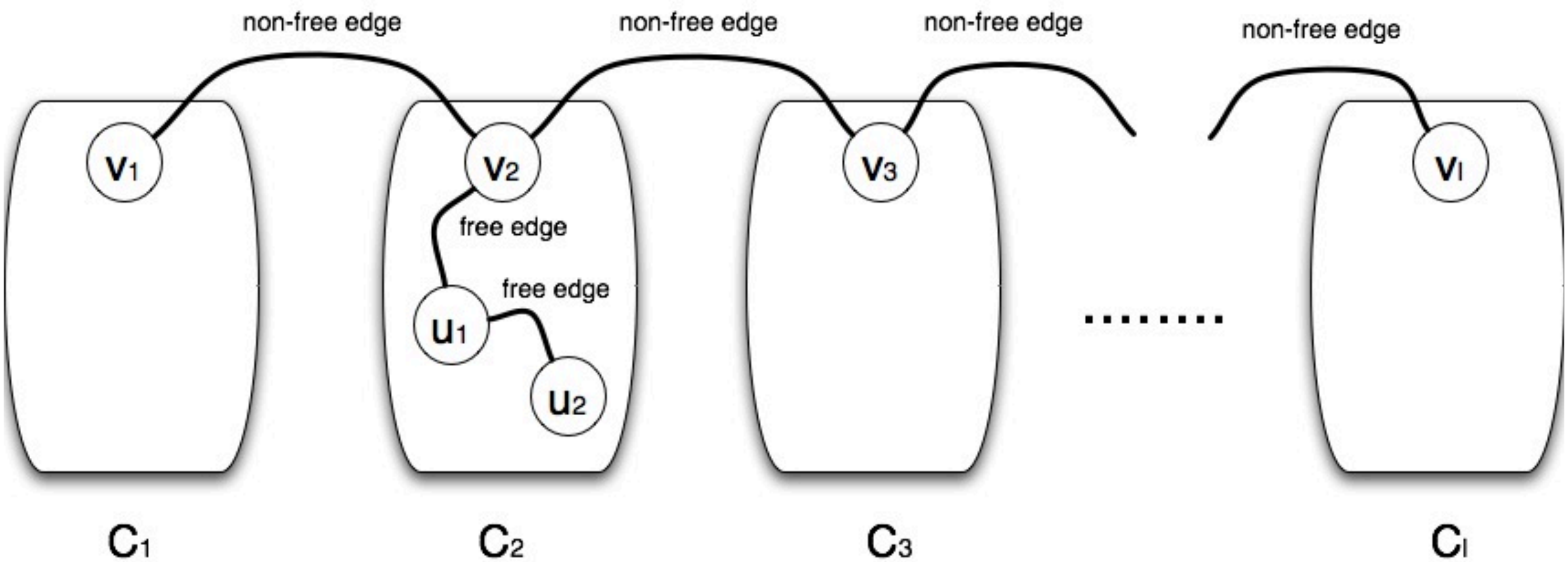
Lower Bound: Half-Empty Configuration

- A sequence of nodes $v_1,\ldots,v_m$ is half-empty with respect to a sequence of tokens $t_1,\ldots,t_m$
  - For every $i \neq j$, either v_i is missing t_j or v_j is missing t_i
  - We call the pair $(\langle v_i \rangle, \langle t_i \rangle)$ a half-empty configuration of size m
- Key point:
  - The definition is entirely based on the absence of tokens, not on the tokens being broadcast

Lower Bound: Proof Steps

- Necessity:
  - 2m useful token exchanges ➔ half-empty configuration of size m
- Monotonicity:
  - Half-empty configuration of size m in round r ➔ half-empty configuration of size m in round 1
- Size:
  - Well-mixed distribution initially: each node has each token independently with probability ¾ ➔ largest half-empty configuration is of size O(log n) whp
- Useful token exchanges in each round is O(log(n)) whp
- Number of token exchanges needed is $\Omega(n^2)$ whp
- Number of rounds needed equals $\Omega(n^2/\log(n))$ whp

Lower Bound: Necessity



- m useful token exchanges imply at least m/2 components
- Take any node v_i from component C_i and let t_i be the token broadcast by v_i
- Then $\langle v_i \rangle$ is half-empty with respect to $\langle t_i \rangle$

Lower Bound: Monotonicity

- Recall that the definition requires
  - For every $i \neq j$, either v_i is missing t_j or v_j is missing t_i
- Any half-empty configuration in round r also exists at the start of round 1

Lower Bound: O(log n) bound on Size

- Let E_m denote the event that there exists a half-empty configuration of size m
- Need m nodes $v_1,\ldots,v_m$ and m tokens $t_1,\ldots,t_m$ such that for each $i \neq j$, either v_i is missing t_j or v_j is missing t_i
- By calculation below, whp $m = O(\log(n))$

$$\Pr[E_m] \leq \binom{n}{m} \cdot \frac{n!}{(n-m)!} \cdot \left(\frac{1}{2}\right)^{\binom{m}{2}} \leq n^{2m} \cdot \frac{1}{2^{m(m-1)/2}} \leq \frac{2^{2m \log n}}{2^{m(m-1)/2}}$$

Lower Bound: Generalizations

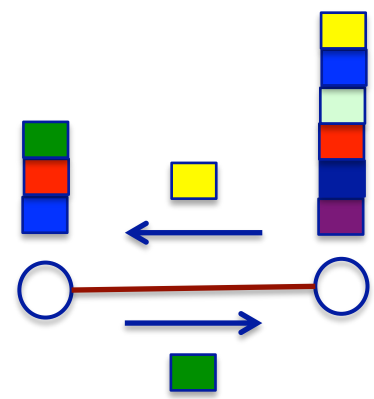
- Takeaway: Even if each node has a large fraction of tokens, "closing the deal" requires nearly quadratic rounds
- Extends to other initial distributions
  - When each node has one token (singleton):
    - Using Hall's Theorem, set up a perfect node-token matching in the well-mixed distribution
    - Can reduce well-mixed distribution to singleton distribution
  - For k tokens, with each token at exactly one node, $\Omega(nk/\log(n))$ lower bound
- Extends to randomized algorithms where the adversary knows the coin outcomes in the current round, but is unaware of future ones
- Extends to multiple tokens per round and other dynamic network models [Kuhn-Haeupler 12]

Revisiting the Adversarial Model

- Online: A sequence of evolving networks
  - Strong Adaptive: Algorithm decides token transmissions without knowledge of neighborhood
  - Weak Adaptive: Algorithm is aware of neighborhood at each round of token transmission
  - Oblivious: Adversary fixes network sequence in advance, which is revealed incrementally
- Offline: Sequence of graphs known to algorithm in advance

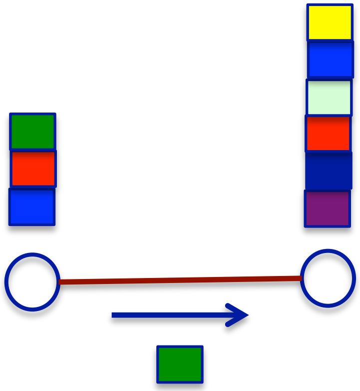
Towards a New Protocol

- Recall the RAND-DIFF protocol
- Along edge (u,v), node u sends a token chosen uniformly at random from $S_u - S_v$
  - Weak adversary cannot set up worst-case edges
- Main hurdle:
  - Lower bound of Ω(n) bits on the communication complexity of set difference



The RAND-SYM-DIFF Protocol

- Along edge (u,v), one token chosen uniformly at random from $(S_u - S_v)$ union $(S_v - S_u)$
- RAND-SYM-DIFF:
  - Repeatedly execute the above step
- Implementation in O(log n) rounds:
  - Each node sends a "fingerprint" of its set
  - If the fingerprints differ, repeat this over a random binary search
- The above uses shared randomness
- For private randomness, use pseudorandom generator for combinatorial rectangles [Lu 02, Gopalan-Meka-Reingold-Trevisan-Vadhan 12]



Upper bound for RAND-SYM-DIFF

- Unable to analyze RAND-SYM-DIFF for arbitrary initial token distributions
- Completes in O(nlog(n)log(k)) rounds with high prob.
  - If network remains stable for O(log n) rounds at a time
  - And the initial token distribution is well-mixed
- Analysis sketch: In each round, one of these is true
  - A node is missing only O(log(n)) tokens
  - A node receives a constant fraction of its missing tokens
  - If number of distinct token sets being held by the nodes is r, then the total number of missing tokens is O(nr) while the number of tokens exchanged is Ω(r)

Summary

- Information spreading over dynamic networks
  - Applicable to computing environments with either dynamic topologies or dynamic network traffic
- Near-tight lower bound for strong adversary model
  - A first separation result of this kind between forwarding and network coding algorithms
- SYM-DIFF, a simple practical protocol under weak adversaries that achieves near optimal bounds for well-mixed distributions
  - Communication complexity of sampling problems
- Three centralized offline approximation algorithms

Open Problems

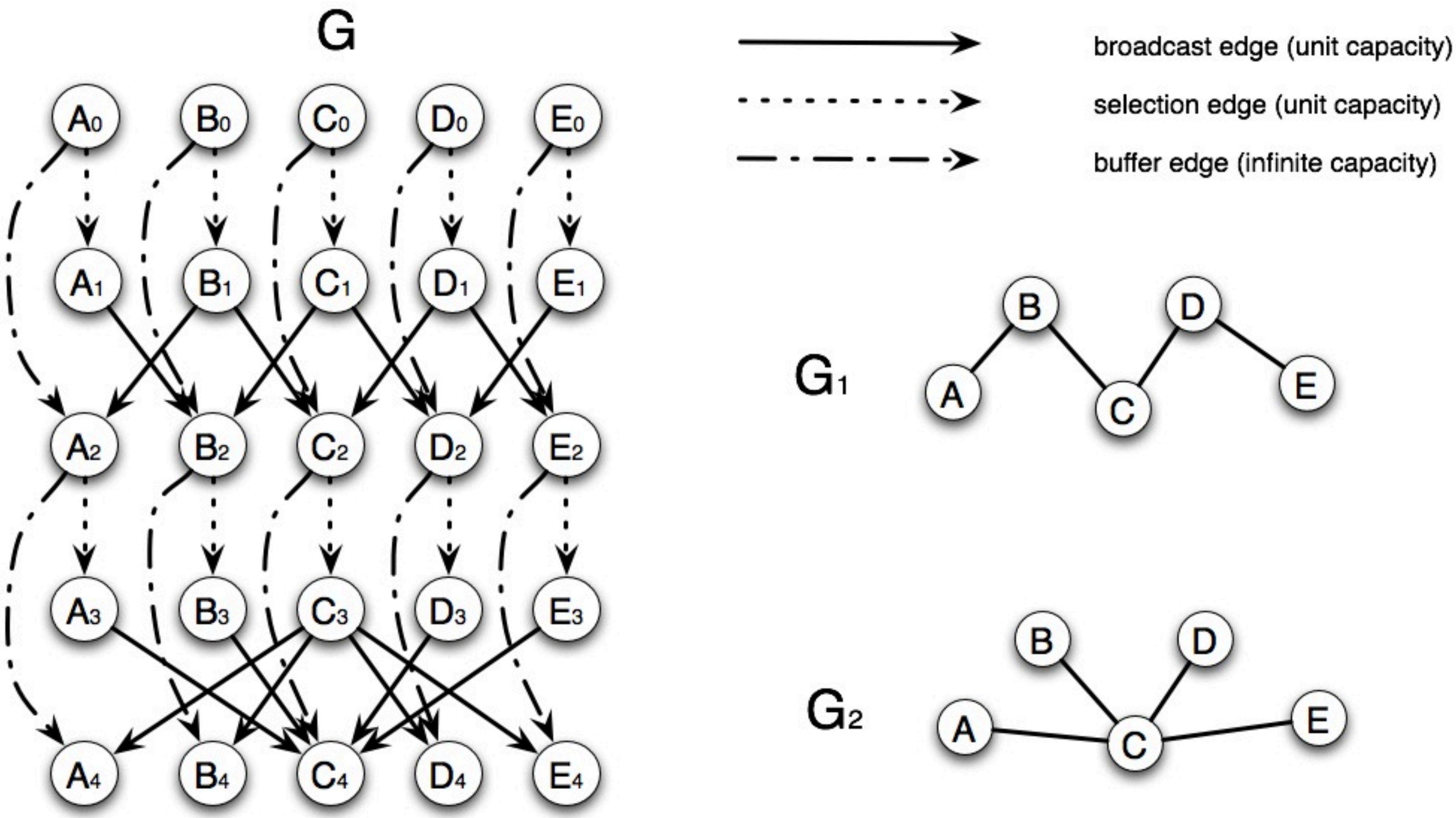
- Strong adaptive adversary:
  - Best general bounds for small message sizes
- Weak adaptive adversary:
  - What is the best upper bound achievable?
  - Analysis of SYM-DIFF protocol in the general case
- Bounds for oblivious adversary
- Alternative dynamic network models that may restrict the range of dynamics
- Offline problem:
  - Can near-linear rounds be achieved for the offline problem in the broadcast model?
  - What is the best approximation factor achievable?
  - Explore further connections to Directed Steiner tree problem and the network coding advantage

Questions?

The Offline Problem

- Input:
  - A sequence <G_r> of graphs
  - An initial distribution of tokens
- Output:
  - A schedule of token dissemination
  - Goal to minimize the number of rounds
- Constraint:
  - Each message contains at most one token
  - We consider both broadcast and multiport models

A Flow Network



Centralized Offline Algorithms

- O(n sqrt{k log(n)}) round broadcast algorithm
  - A series of flows using random source-sink pairs
- O((n + k) $\log^2$(n)) round algorithm in which each edge can carry a token in each round
  - Change the flow network to accommodate the communication model
  - A series of flows using random source-sink pairs
- O(n^ε) bicriteria approximation assuming O(log(n)) tokens can be sent per round
  - Packing of directed Steiner trees in flow network

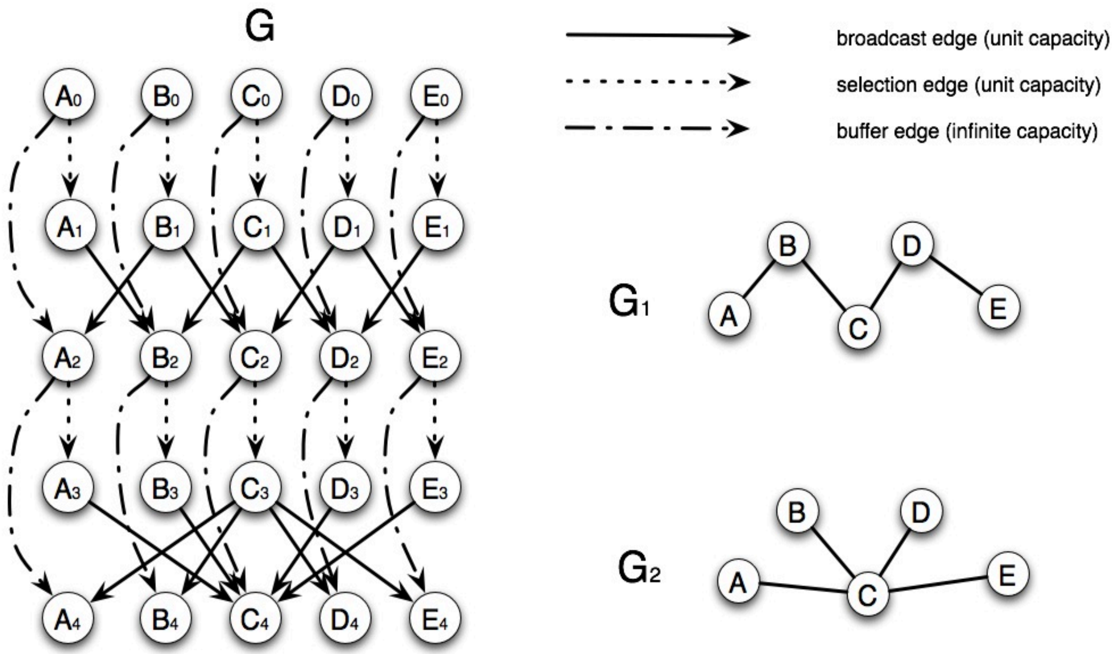
Modeling Network Dynamics

- Adversarial: Dynamics controlled by an adversary
  - Edges arrive/depart over time
  - There may be node churn
- Stochastic: Dynamics through a probabilistic process
  - Neighbors of new nodes randomly selected
  - Edge failure/recovery events drawn from probability distribution
- Strategic: Dynamics through strategic interactions
  - Each node is a potentially independent agent, with its own utility function, and rationally behaved
  - Focus on equilibria or transient behavior

An $O(n\sqrt{k \log n})$ round algorithm

- Phase I (Gather): Repeat m times
  - A destination node is selected at random and each token is sent to the destination
- Phase II (Disperse): For each token, in sequence
  - Each token is broadcast by every node holding that token for Θ(nlog(n)/m) rounds
- Claim I: Each iteration of Phase I can be completed in O(n) rounds
  - Number of rounds = O(nm + knlog(n)/m), minimized when m equal $\sqrt{k \log n}$ to give $O(n\sqrt{k \log n})$ rounds
- Claim II: Gossip is successfully completed at the end of Phase II with high probability

Analysis of Phase I



- Connect nodes at top level to a source with unit-capacity edges
- Consider destination node at level $2n$ as sink
- The source-sink min-cut has capacity at least n, so all n tokens can be routed to destination in at most $2n$ steps

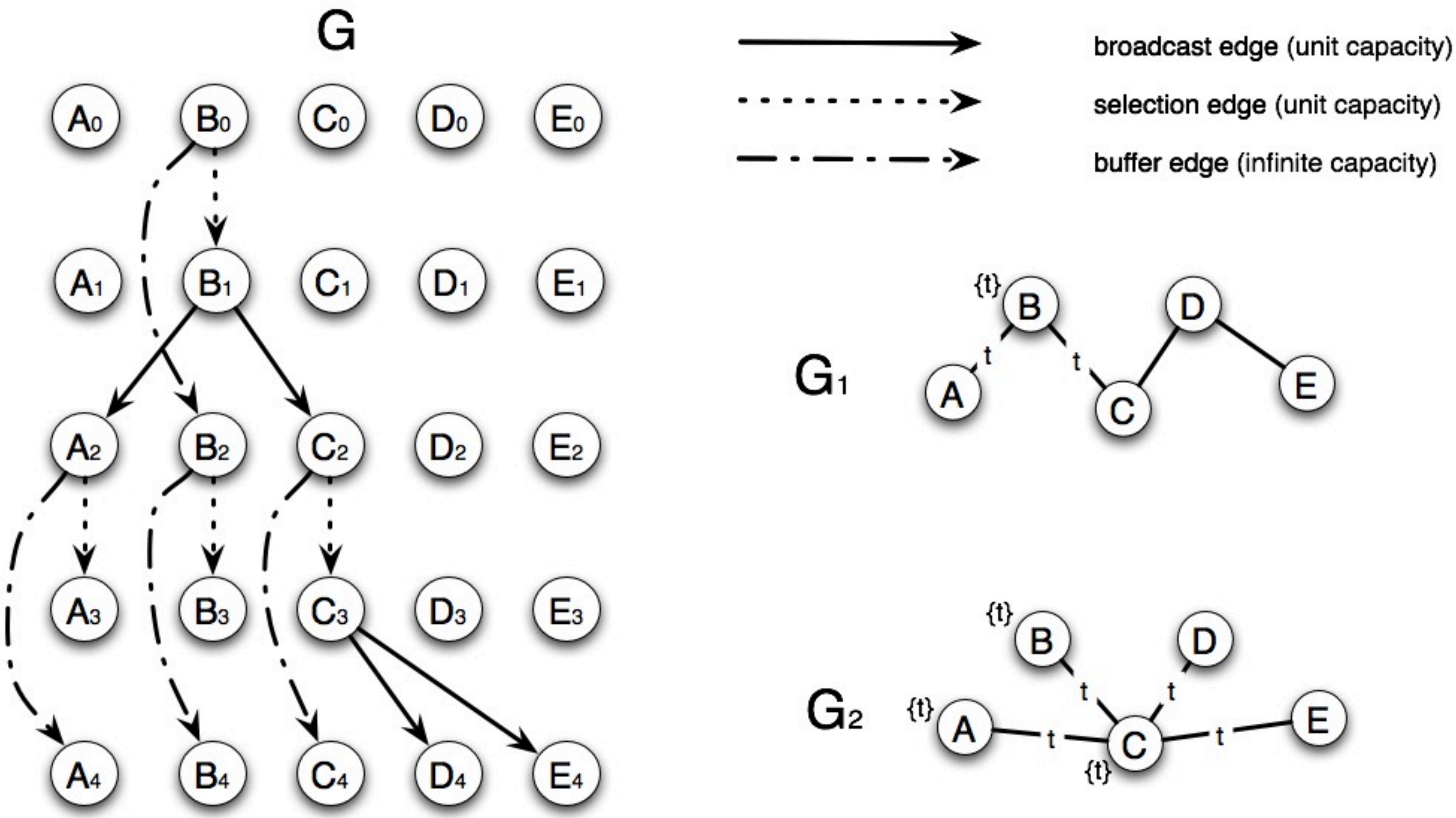
Analysis of Phase II

- Let S be set of m destination nodes
- With high probability, each node is within $O(n\log(n)/m)$ "hops" of S in the layered graph
  - Straightforward Chernoff bound argument
- Every node holding each token broadcasts for $\Theta(n\log(n)/m)$ rounds
  - Every node receives every token whp
- Can be derandomized using the method of conditional expectations and pessimistic estimators

Offline Via an Optimization Lens

- Input: A sequence of graphs <G_r> and an initial distribution of tokens
- Output: A schedule for gossip that minimizes completion time
- NP-hard
  - Reduction from the problem of maximizing the number of disjoint set covers
- Suppose L* is the minimum number of rounds needed for gossip, among all graph sequences
  - We know that L* is $O(n\sqrt{k \log n})$
  - Question: How well can we approximate L*?

Packing Directed Steiner Trees



Linear Programming Rounding

$$\max \sum_{T \in \Gamma} x_T$$

$$\sum_{T: e \in T} x_T \le c_e \qquad e \in E$$

$$x_T \ge 0 \qquad T \in \Gamma$$

x_T = indicator variable for tree T

Γ = set of all candidate Steiner trees

c_e = capacity of edge e

- Construct L*-layered directed flow graph
- Compute $O(n^{\varepsilon})$ approximation for Fractional Steiner tree packing [Cheriyan-Salvatipour 06]
- Randomized rounding
  - Packs same number of trees as optimum
  - Incurs O(log(n)) blowup in capacity constraint
- Repeat $O(n^{\varepsilon})$ times to pack all trees
- Yields $(n^{\varepsilon}, O(\log(n)))$ approximation