6.852: Distributed Algorithms

- Leader election in a synchronous ring
  - lower bound for comparison-based algorithms
  - non-comparison-based algorithms
- Algorithms in general synchronous networks
  - leader election
  - breadth-first search
  - broadcast and convergecast
  - shortest paths
- Reading: chap 3.6, 4.1-2
- Next: 4.3-4

Last lecture

- Leader election in a synchronous ring
  - LeLann-Chang-Roberts algorithm
    - pass UIDs in one direction, elect max
    - proof: invariants
    - time complexity: n (or 2n for halting, unknown size)
    - msg complexity: $O(n^2)$
  - Hirschberg-Sinclair algorithm
    - successive doubling (uses bidirectional channels)
    - msg complexity: O(n log n)
    - time complexity: O(n) (dominated by last phase)
  - Non-comparison-based algorithms
    - wait quietly until your “turn”, determined by UID
    - msg complexity: O(n)
    - time complexity: $O(u_{min}\ n)$, or $O(n\ 2^{u_{min}})$ if n unknown

Lower bounds for leader election

- Can we get lower time complexity?
  - easy n/2 lower bound (informal)
- Can we get lower message complexity?
  - $\Omega(n \log n)$ message complexity
- Assumptions
  - comparison-based algorithm
  - unique start state (except for UID), deterministic

Comparison-based algorithms

- Depend only on relative order of UIDs
  - identical start state, except for UID
  - manipulate ids only by copying, sending, receiving, and comparing (<, =, and >)
  - can use results of comparisons to decide what to do
    - what (if anything) to send to neighbors
    - whether to elect self leader
    - local state transition

Lower bound proof (overview)

- For any n, there is a ring of size n such that in that ring, any leader election algorithm has:
  - $\Omega(n)$ "active" rounds
  - $\Omega(n/i)$ msgs sent in active round i (for $i > \sqrt{n}$)
  - Thus, $\Omega(n \log n)$ msgs total.
- For $n = 2^b$, use "bit-reversal ring"
- Generalize for other n: c-symmetric rings
- Key lemma: Processes whose neighborhoods "look the same" act the same (until information from outside their neighborhoods reaches them).
  - need lots of active rounds to break symmetry

Lower bound proof

- a round is *active* if some (non-null) msg is sent
- *k-neighborhood* of a process:
  the 2k+1 processes within distance k
- $(u_1, u_2, \ldots, u_k)$ & $(v_1, v_2, \ldots, v_k)$ *order-equivalent* if
  - $u_i < u_j$ iff $v_i \leq v_j$ for all i,j
- two process states s and t *correspond* with respect to $(u_1, u_2, \ldots, u_k)$ & $(v_1, v_2, \ldots, v_k)$ if they are identical except that occurences of u_i in s are replaced by v_i in t for all i (& no other UIDs)
  - analagous defn for corresponding messages

Lower bound proof

- Key lemma: Suppose A is a comparison-based algorithm on a synchronous ring network with processes i and j. If the sequences of UIDs in their k-neighborhoods are order-equivalent then at any point after at most k active rounds, i and j are in corresponding states (with respect to their k-neighborhoods' UID sequences).
- Proof: Induction on r = #completed rounds.
- Base: r = 0.
  - Start states of i and j are identical except for UIDs.
  - They correspond wrt k-nbhd for any $k \geq 0$.

Lower bound proof

- Inductive case:
  - Assume true after round r-1, for all i,j,k.
  - Prove true after round r, for all i,j,k.
  - Fix i,j,k, where i and j have order-equiv k-nbhds.
  - Assume i ≠ j and at most k of first r rounds are active.
    - Trivial otherwise
  - By IH: i and j in corresponding states wrt k-nbhds.
  - Case analysis:
    - If neither i nor j receives non-null msg, make corresponding transition, so end up in corresponding states (wrt k-nbhds).

Lower bound proof

- Either i or j receives non-null msg in round r.
  - round r is active: at most k-1 active of first r-1 rounds
  - (k-1)-nbhds of i-1 and j-1 are order-equivalent
  - By IH: after round r-1, processes i-1 and j-1 in corresponding states wrt their (k-1)-nbhds (and thus wrt k-nbhds of i and j).
  - Thus, msg from i-1 to i and from j-1 to j correspond.
  - Similarly for msgs from i+1 to i and from j+1 to j.
  - So i and j are in corresponding states and receive corresponding messages, so make corresponding transition and end up in corresponding state.

Lower bound proof

- Corollary 1: Suppose A is a comparison-based leader-election algorithm on a synchronous ring network and k is an integer such that for any process i, there is a distinct process j such that i and j have order-equivalent k-neighborhoods. Then A has more than k active rounds.
- Proof: By contradiction.
  - Suppose A elects i in at most k active rounds.
  - By assumption, there is a distinct process j with an order-equivalent k-neighborhood.
  - By previous lemma, i and j are in corresponding states, so j is also elected—a contradiction.

Lower bound proof

- Corollary 2: Suppose A is a comparison-based algorithm on a synchronous ring network, and k and m are integers such that the k-neighborhood of any process is order-equivalent to that of at least m-1 other processes. Then at least m messages are sent in A's kth active round.
- Proof: By defn, some process sends a message in A's kth active round. By assumption, at least m-1 other processes have order-equivalent k-neighborhoods. By the lemma, immediately before this round, all these processes are in corresponding states. Thus, they all send messages in this round, so at least m messages are sent.

Lower bound proof

- We want a ring with many order-equivalent neighborhoods.
- For powers of 2: bit-reversal rings
  - UID is bit-reversed process number
  - for every segment of length $n/2^b$, there are (at least) 2^b order-equivalent segments (including original)
    - for every process i, at least n/4k processes (including i) with order-equivalent k-neighborhoods for $k < n/4$.
  - more than n/8 active rounds
  - #msgs $\geq n/4 + n/8 + n/12 + ... + 2 = \Omega(n \log n)$

Lower bound proof

- c-symmetric ring: For every l such that $\sqrt{n} < l < n$, and every sequence S of length l in the ring, there are at least $\lfloor cn/l \rfloor$ order-equivalent occurrences.
- [Frederickson-Lynch] There exists c such that for every positive integer n,
  there is a c-symmetric ring of size n.
- Given c-symmetric ring, argue similarly to before.

General synchronous networks

- Digraph G = (V,E) and set of messages M
  - V = set of processes
  - E = set of communication channels
  - distance(i,j) = shortest distance from i to j
  - diam = max distance(i,j) for all i,j
  - assume: strongly connected (diam < ∞), UIDs
- For each process:
  - states
  - start: nonempty subset of states
  - msgs: maps (state,out-nbr) to $M_\perp$
  - trans: maps (state,in-nbrs→$M_\perp$) to states

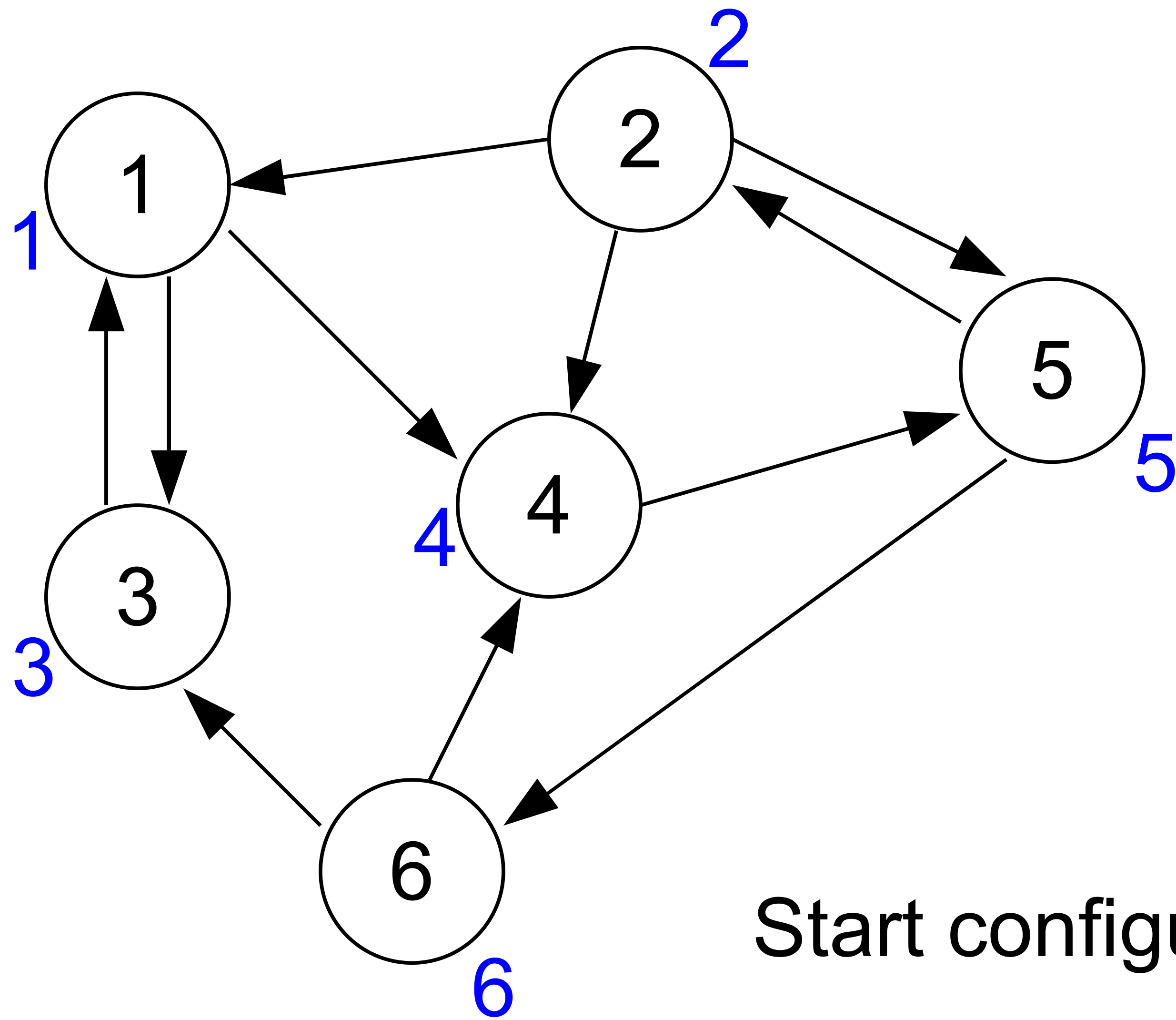
Leader election in general network

- Simple “flooding” algorithm:
  - Assume diameter is known (diam).
  - Every round: Send max UID seen to all neighbors.
  - Stop after diam rounds.
  - Elect self iff own UID is max seen.

Leader election in general network

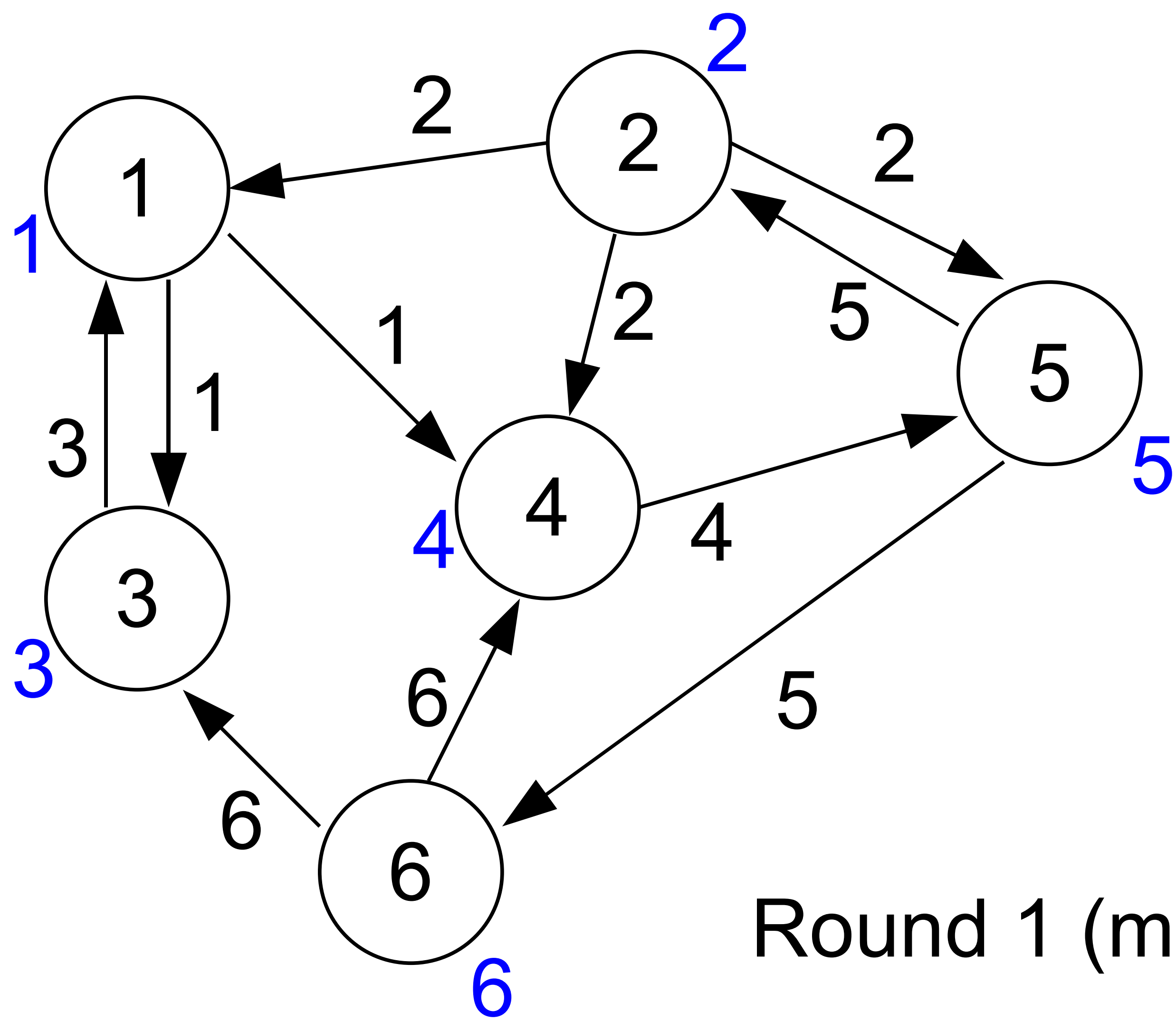
- states
  - UID
  - max-uid (initially UID)
  - status (one of: unknown, leader, not-leader)
  - round
- msgs
  - if round < diam send send max-uid to all neighbors
- trans
  - increment round
  - max-uid := max (max-uid, UIDs received)
  - if round = diam then
    - status := leader if max-uid = UID, not-leader otherwise

Leader election in general network



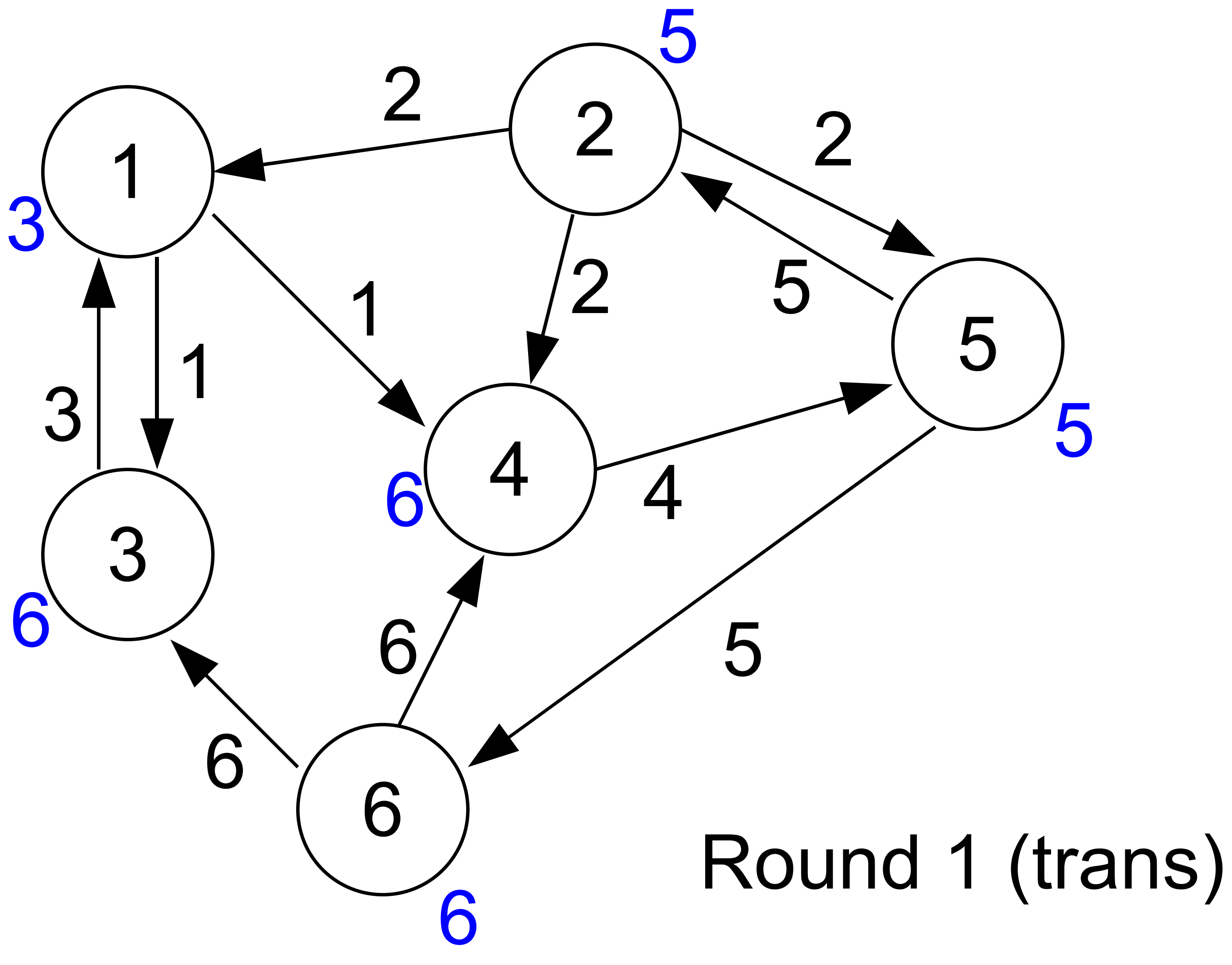
Start configuration

Leader election in general network

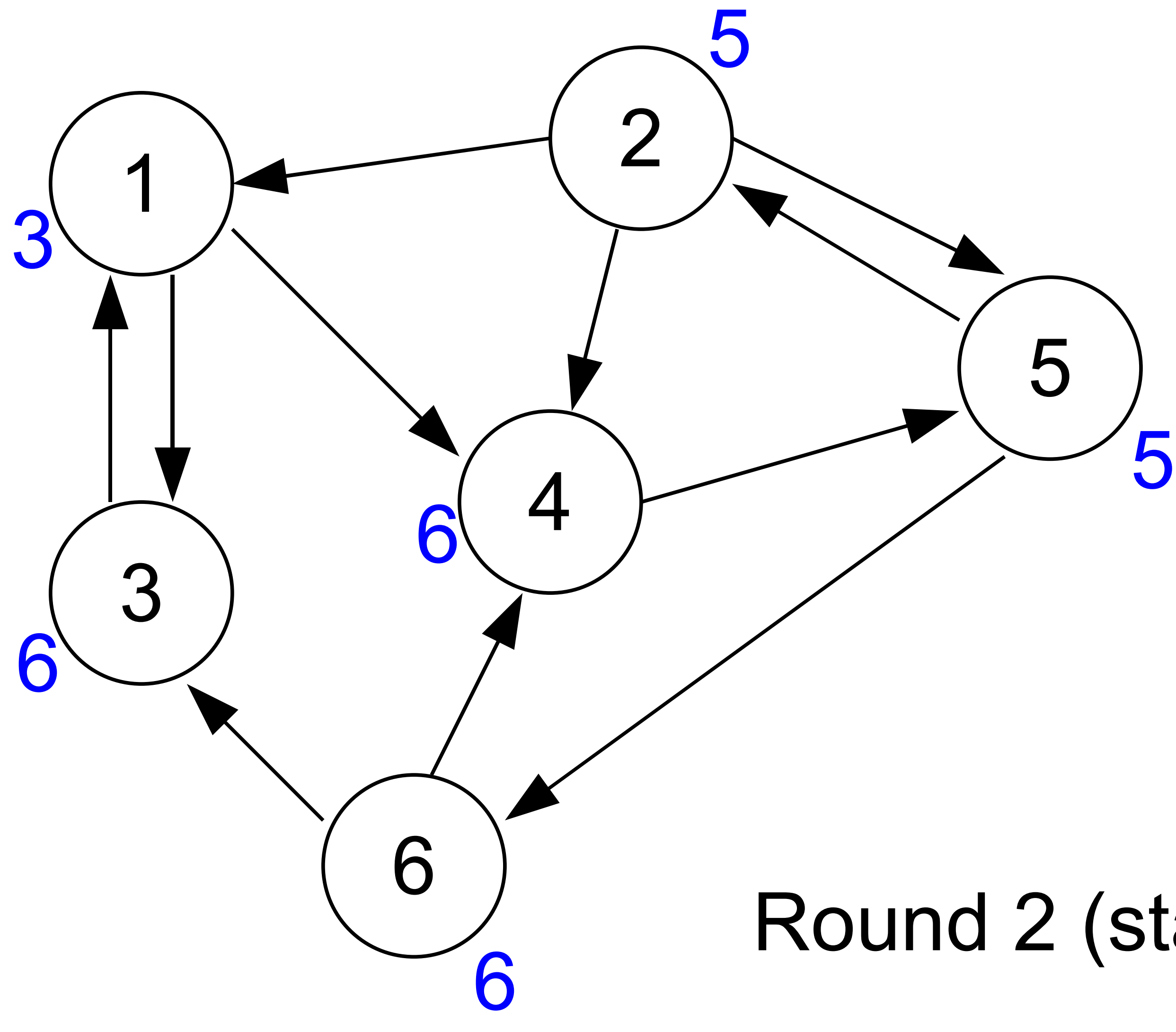


Round 1 (msgs)

Leader election in general network

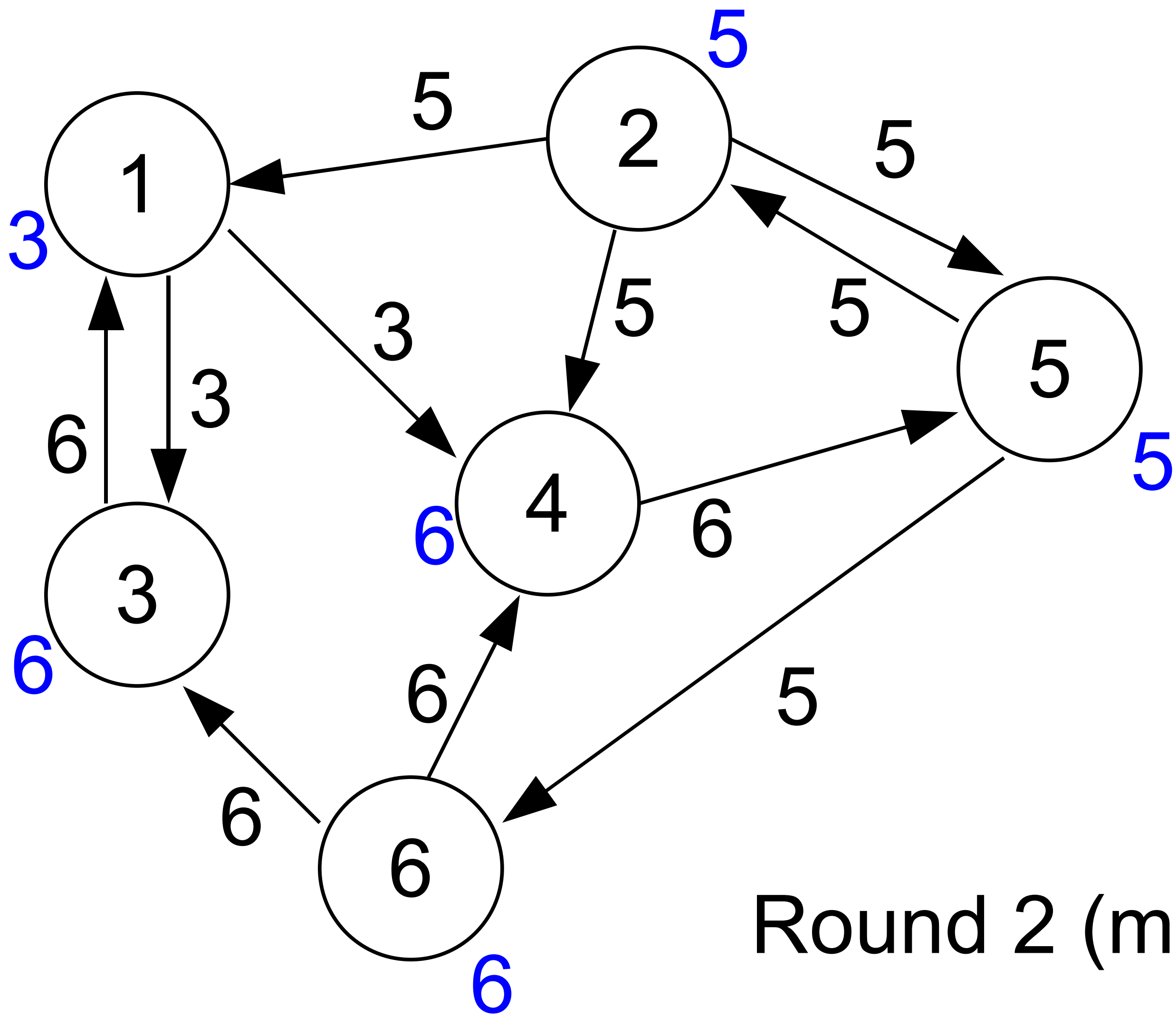


Leader election in general network



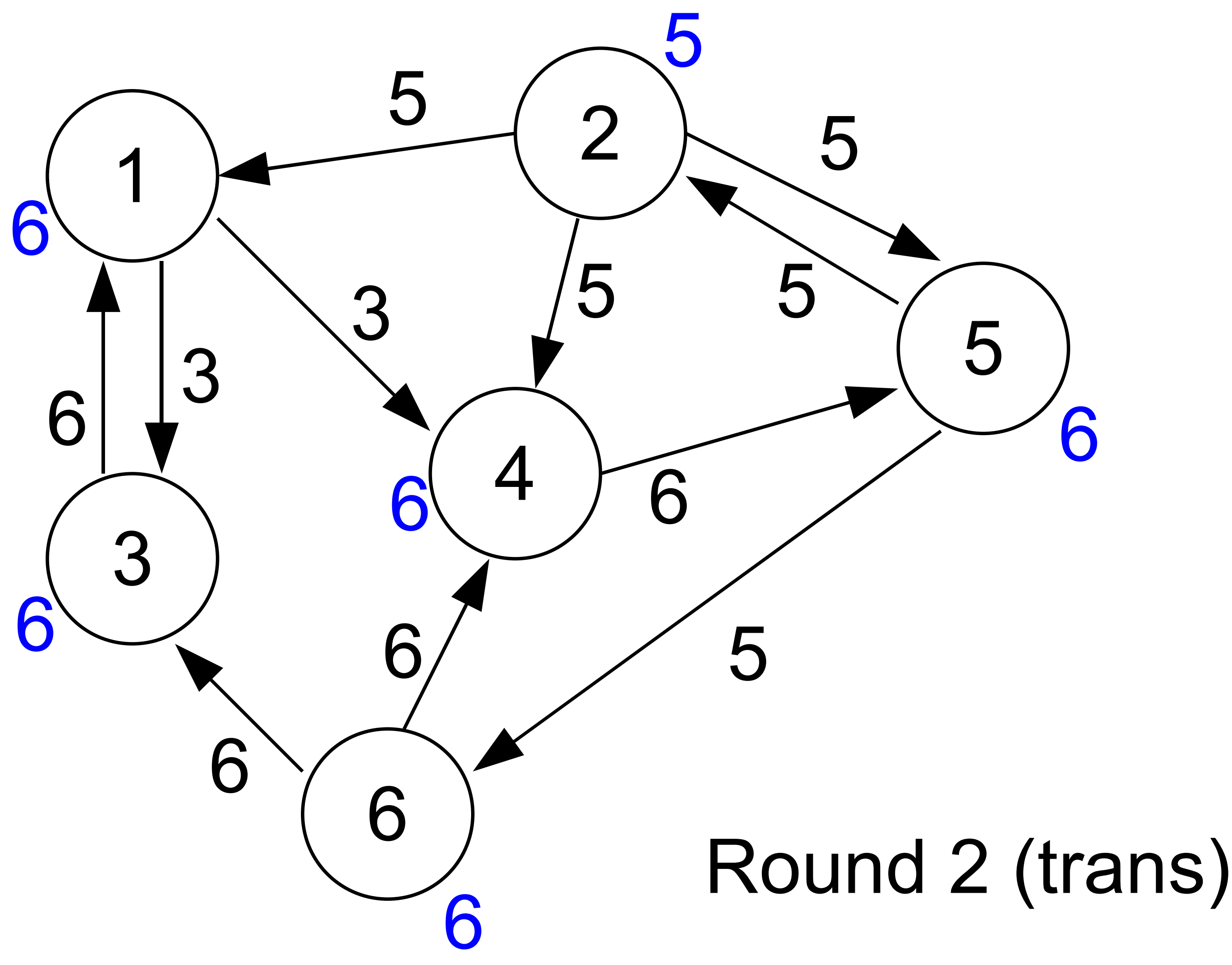
Round 2 (start)

Leader election in general network

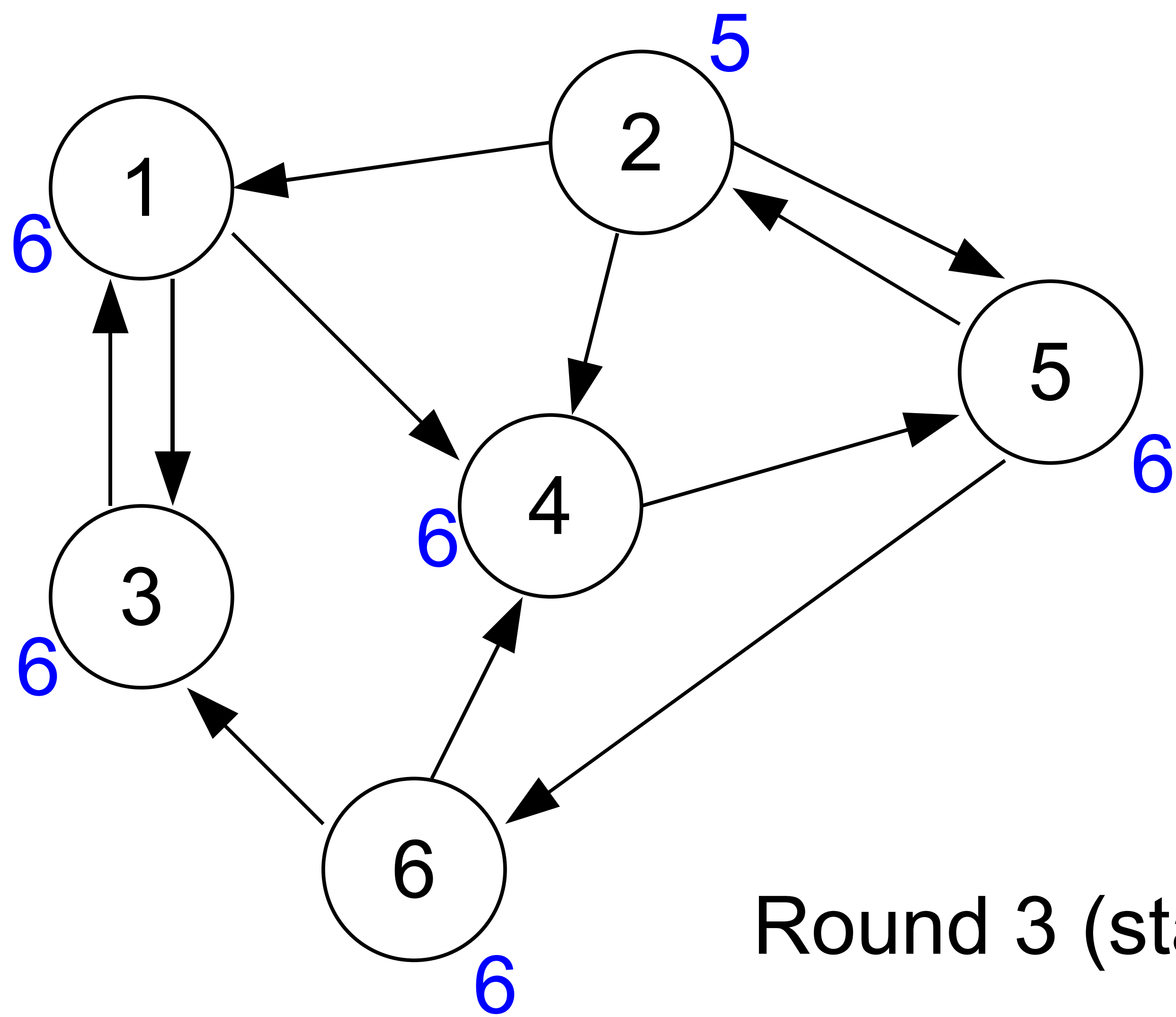


Round 2 (msgs)

Leader election in general network

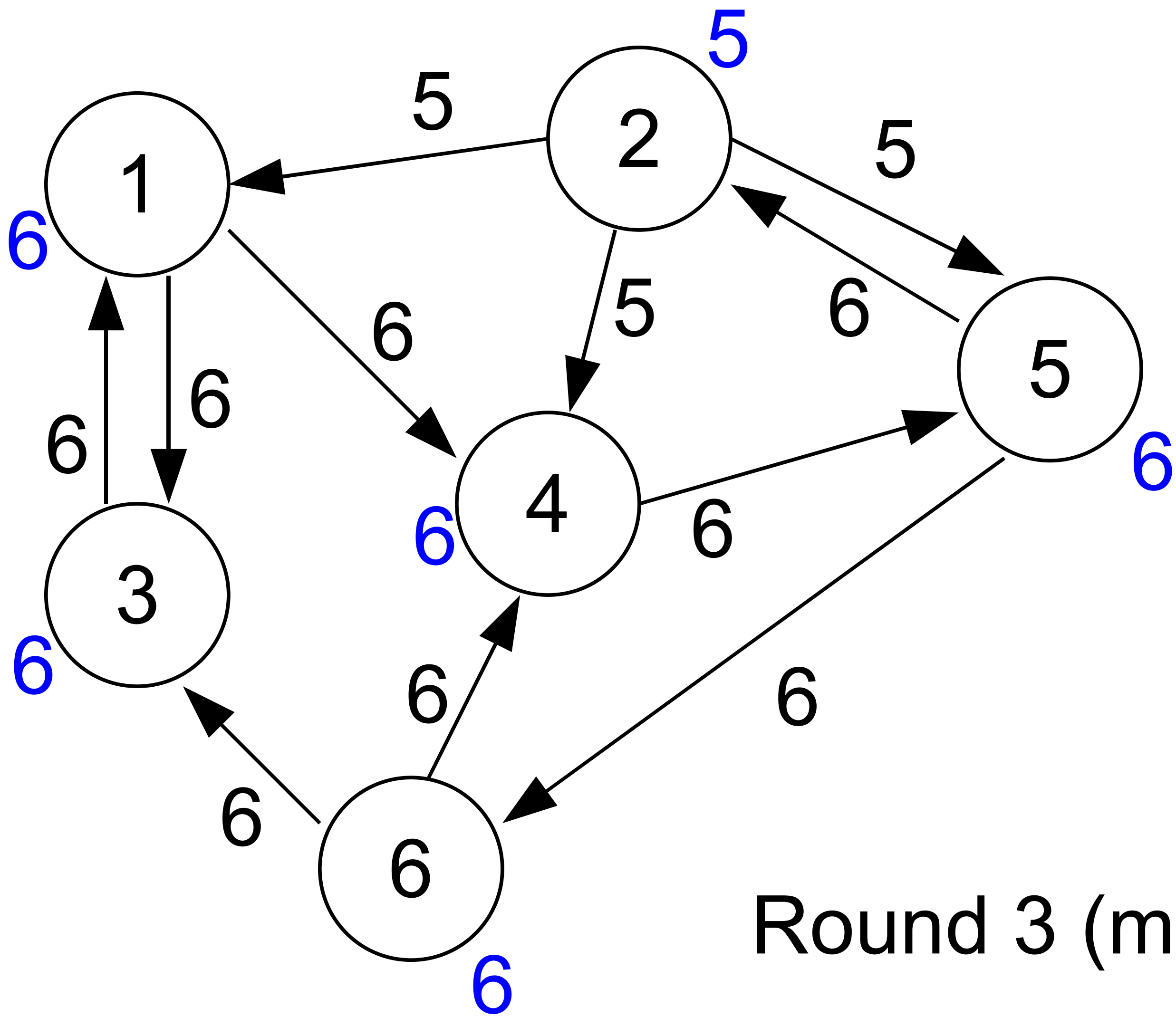


Leader election in general network



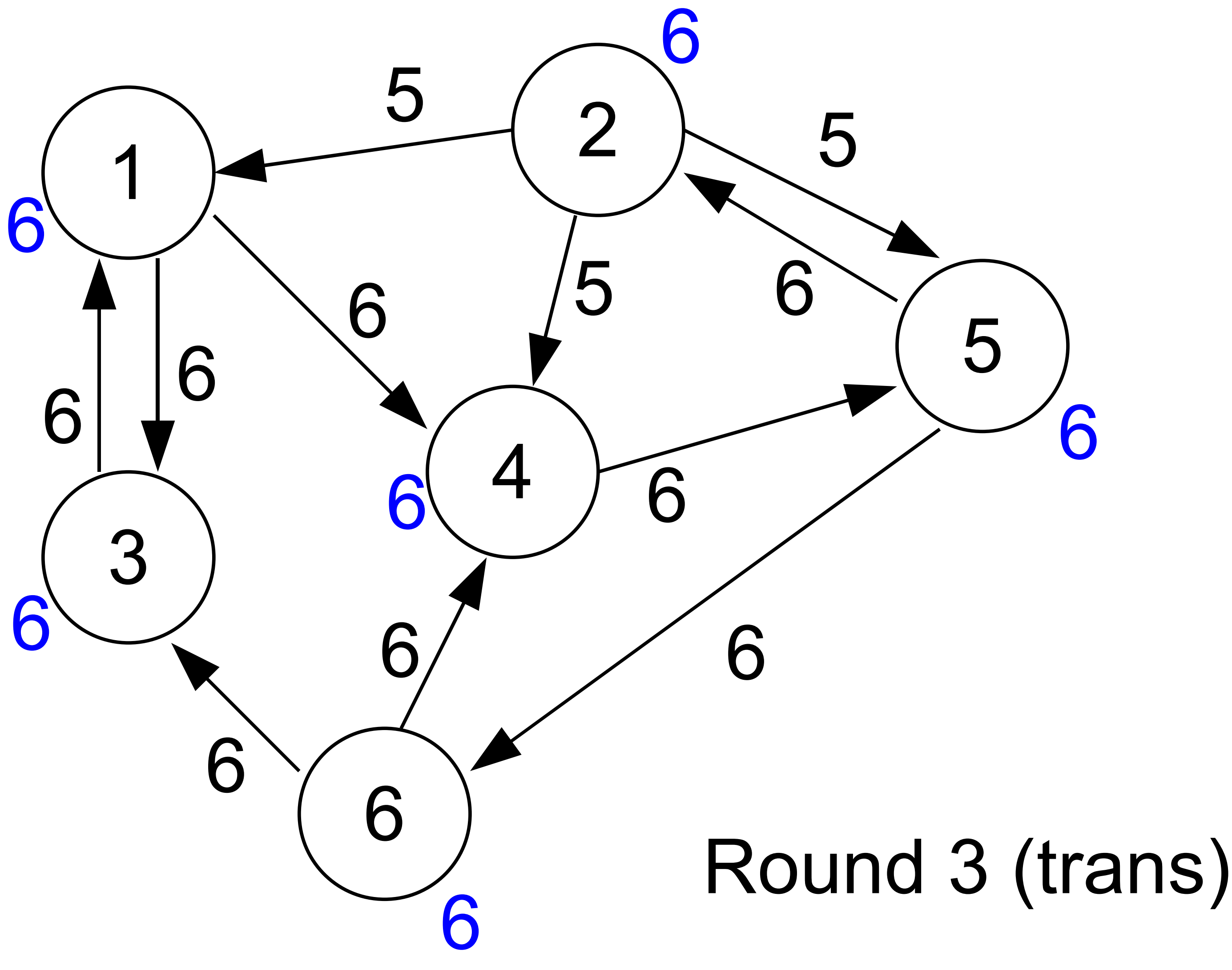
Round 3 (start)

Leader election in general network

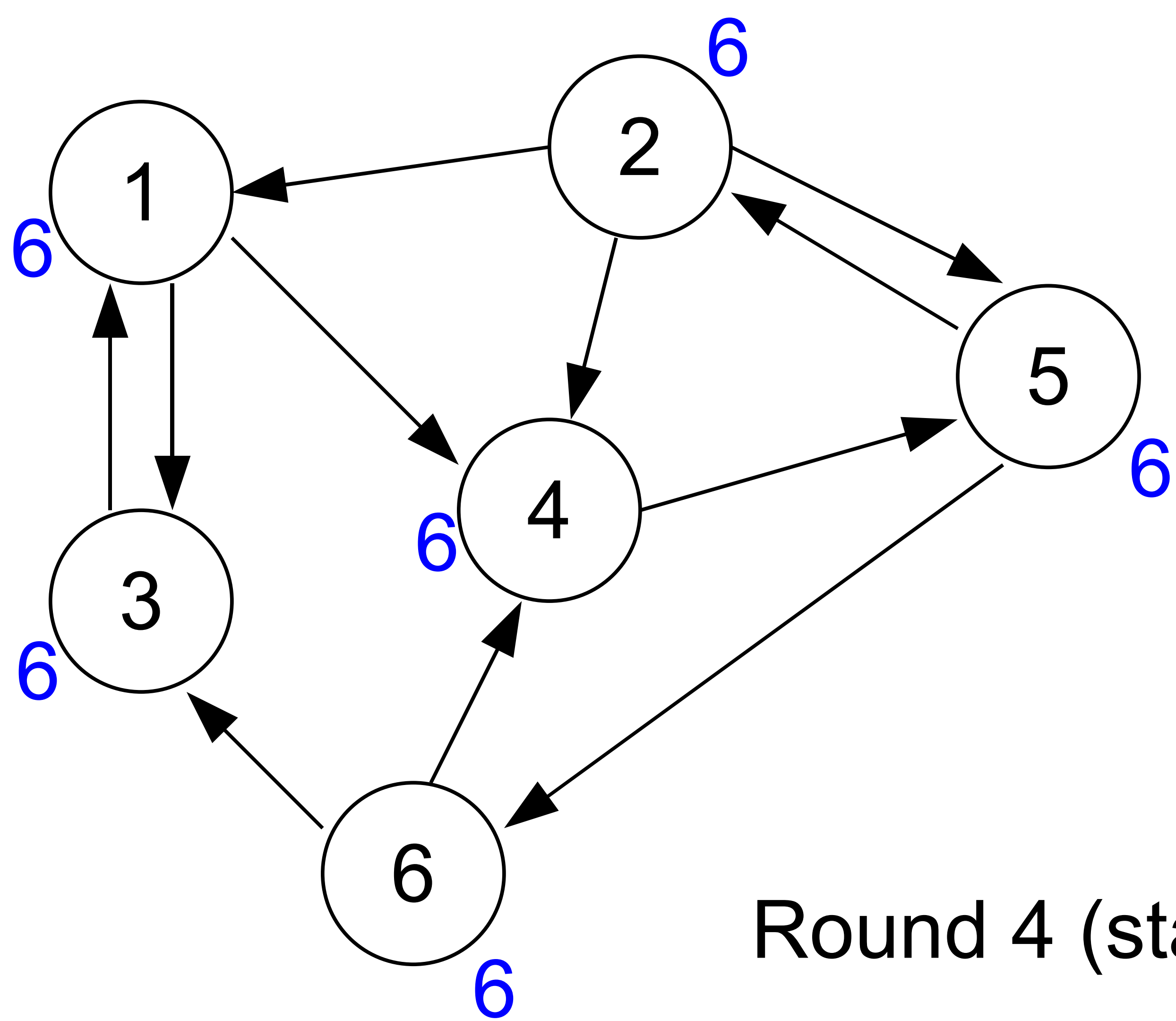


Round 3 (msgs)

Leader election in general network

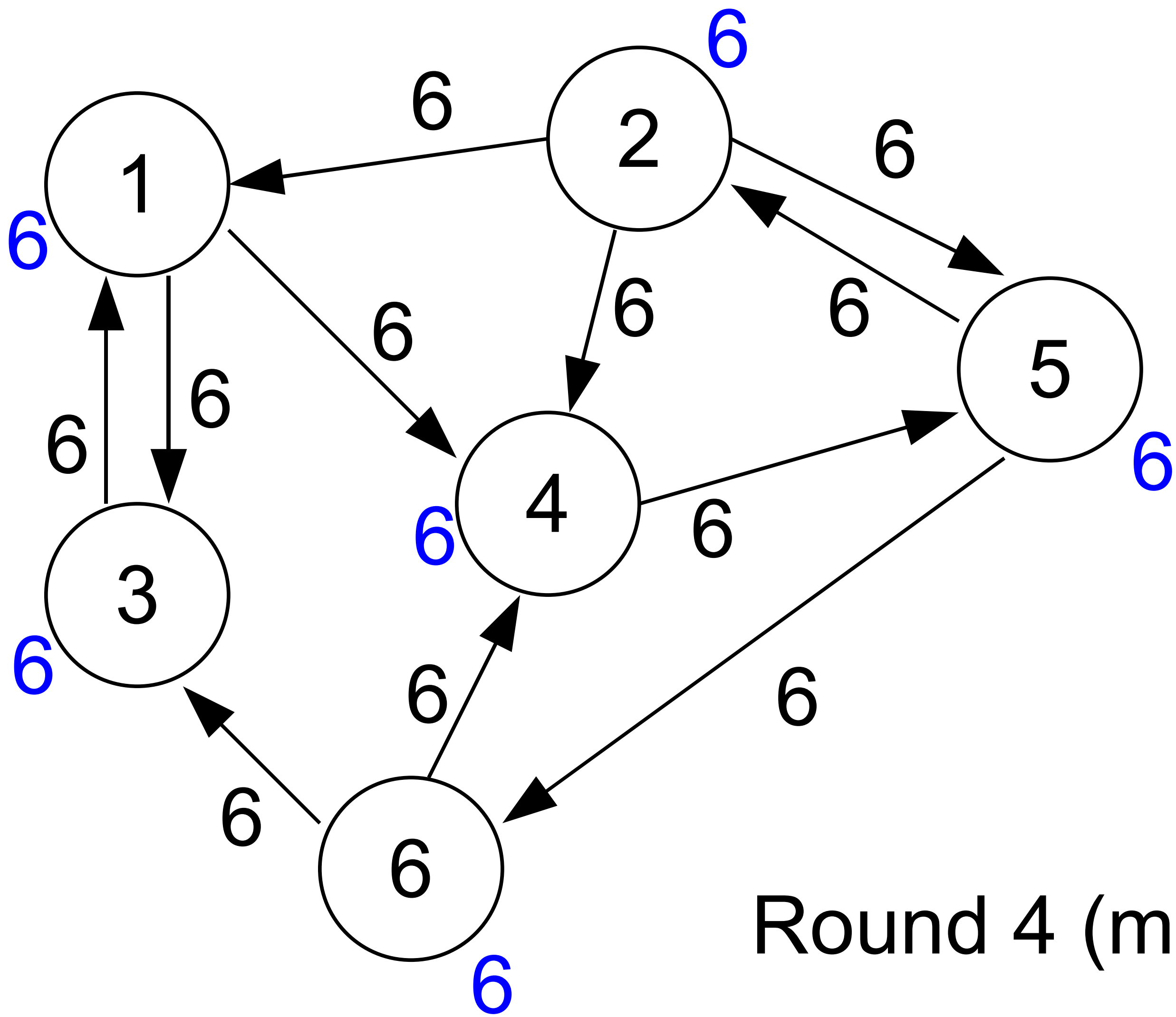


Leader election in general network



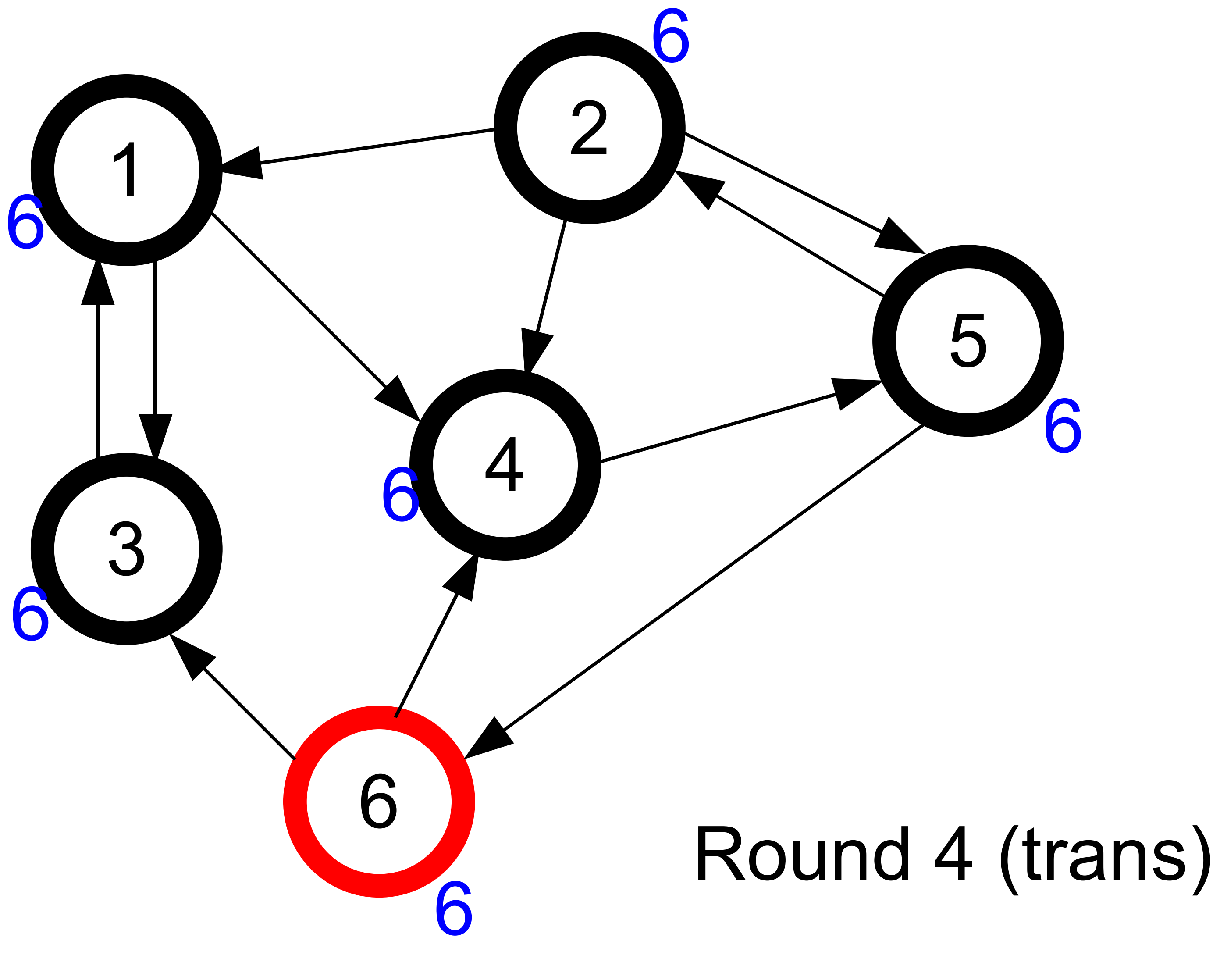
Round 4 (start)

Leader election in general network



Round 4 (msgs)

Leader election in general network



Leader election in general network

- Simple “flooding” algorithm:
  - Assume diameter is known (diam).
  - Every round: Send max UID seen to all neighbors.
  - Stop after diam rounds.
  - Elect self iff own UID is max seen.
  - Time complexity: diam
  - Msg complexity: diam |E|
- Proof?

Leader election in general network

- After round r:
  - if distance(j,i) ≤ r then $\text{max-uid}_i \geq UID_j$
- Proof (by induction on r):
  - Base: r = 0
    - distance(j,i) = 0 implies j = i, and $\text{max-uid}_i = UID_i$
  - Inductive step: assume for r-1, prove for r

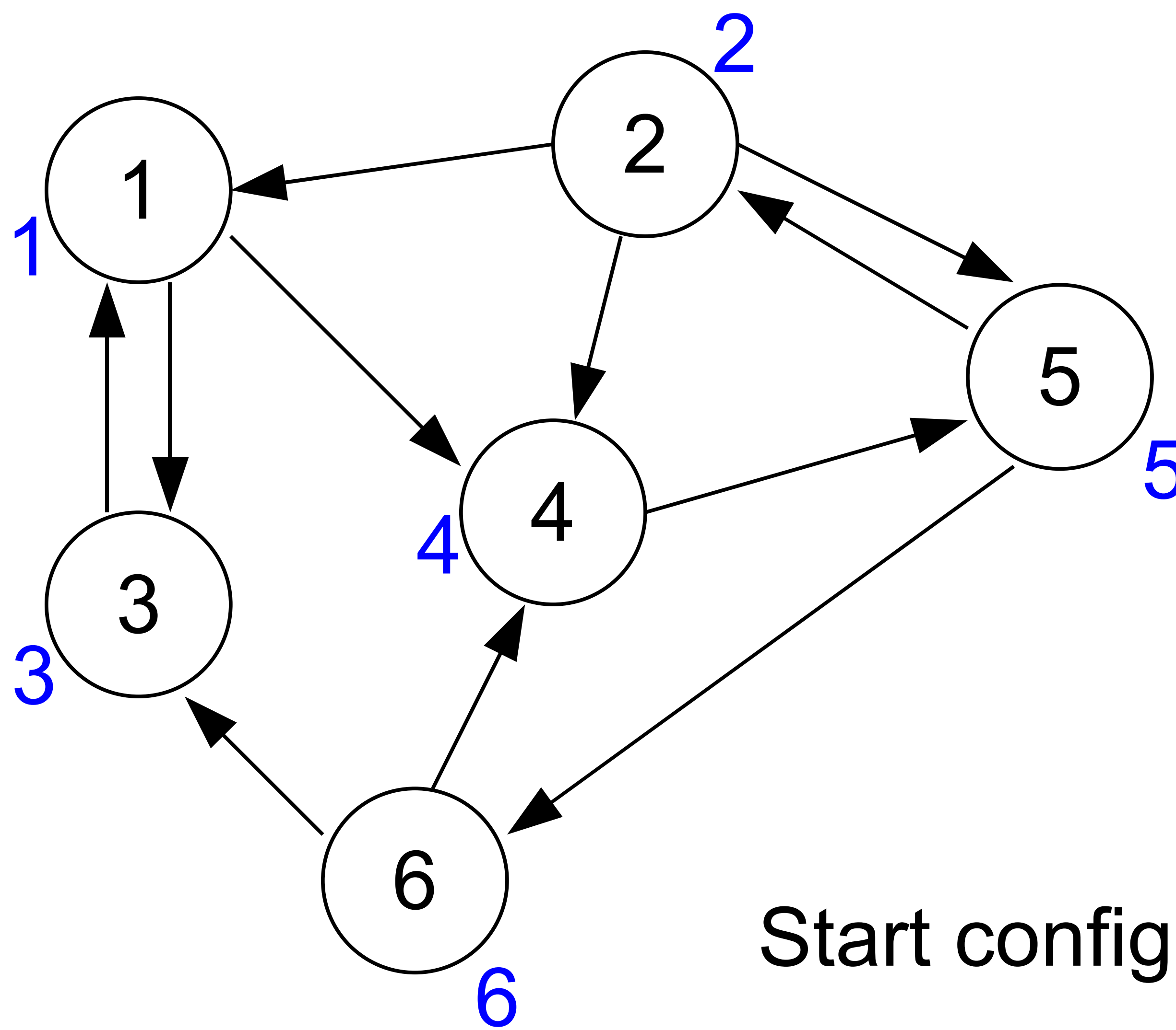
Leader election in general network

- Do we need to know diameter?
- Can we reduce time complexity?
- Can we reduce message complexity?

Leader election in general network

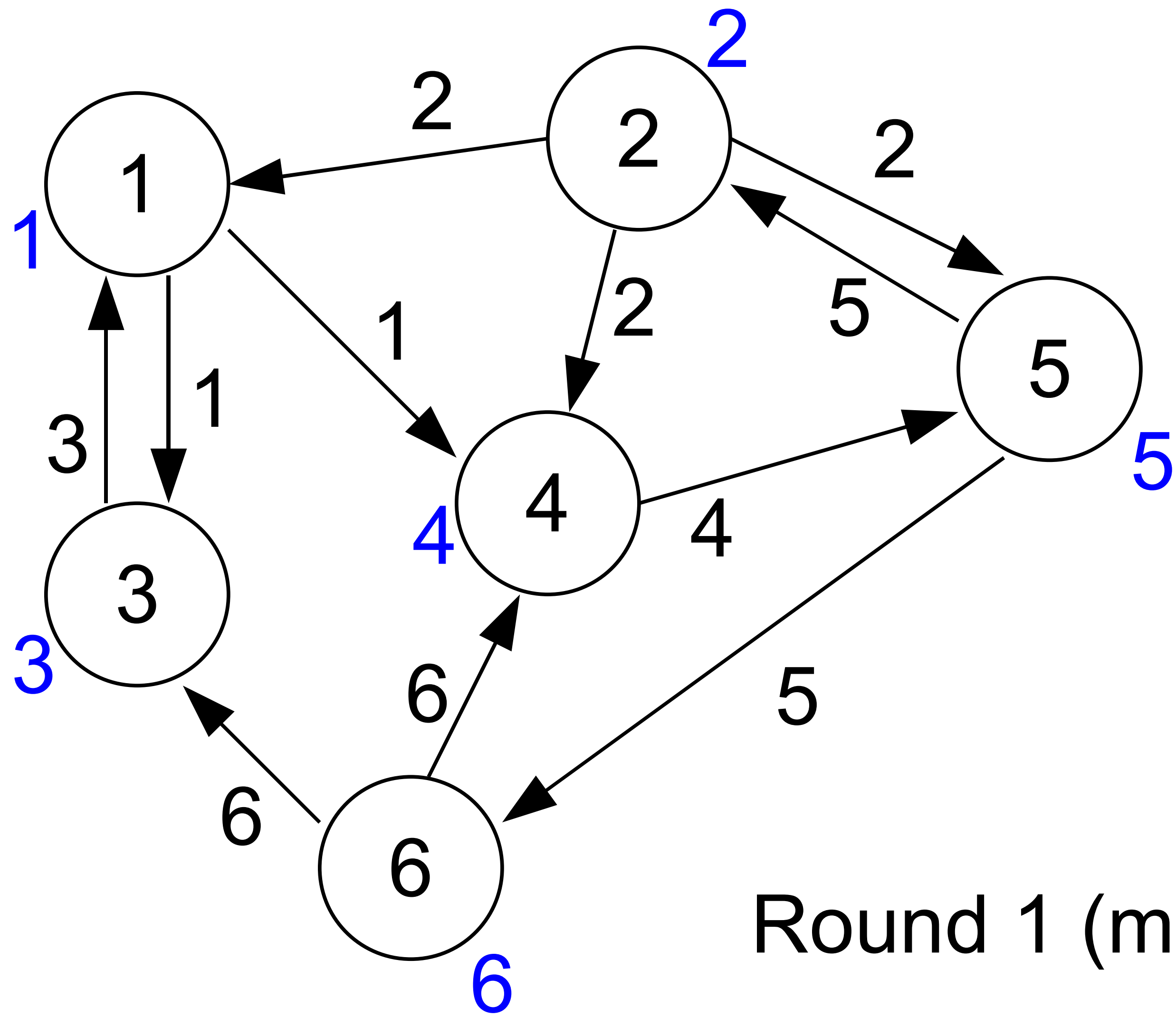
- Reducing message complexity
  - don't send same UID twice
  - new state var: new-info: Boolean, initially true
  - only send max-uid if new-info = true
  - new-info := (max UID received > max-uid)

Leader election in general network



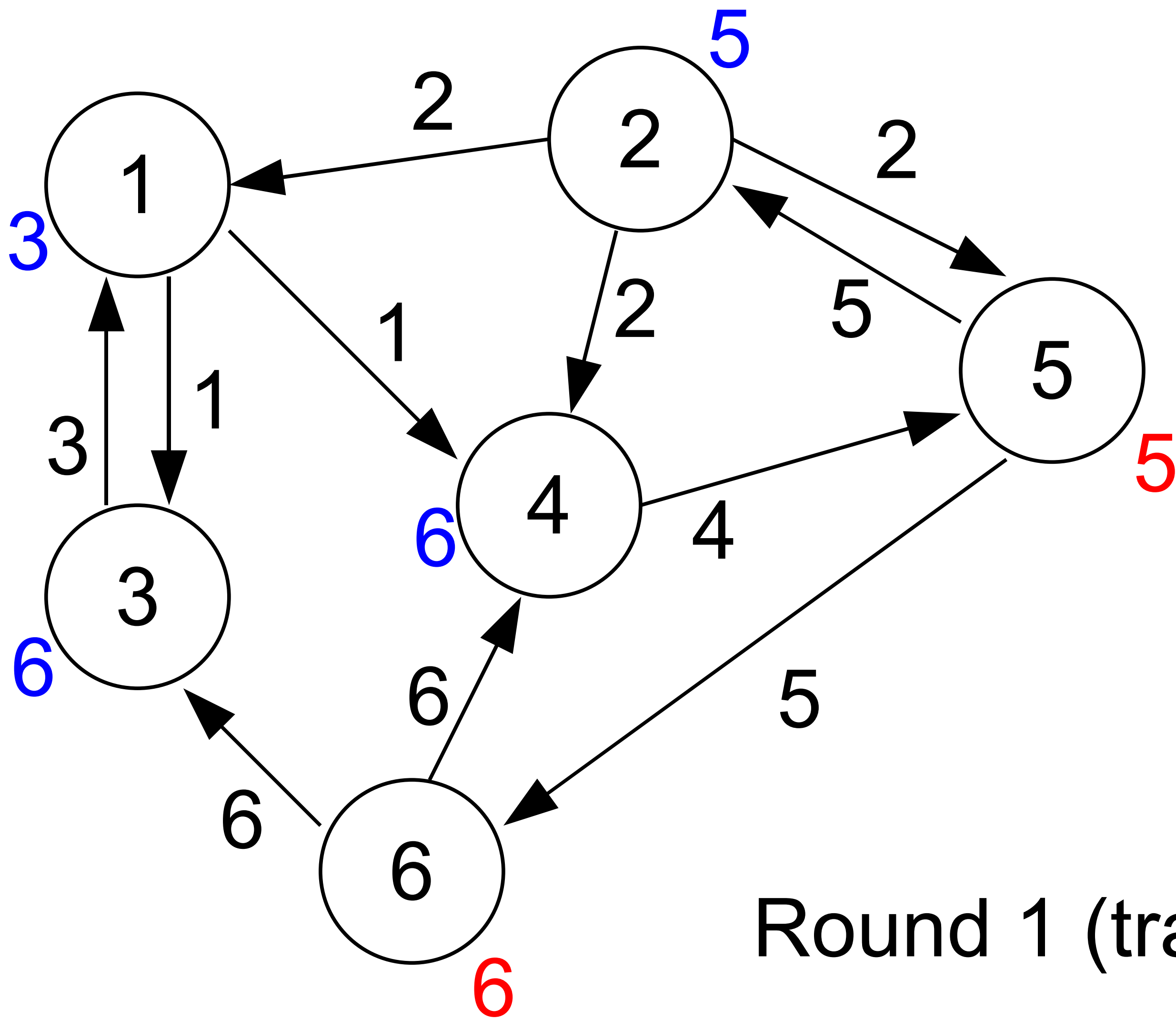
Start configuration

Leader election in general network



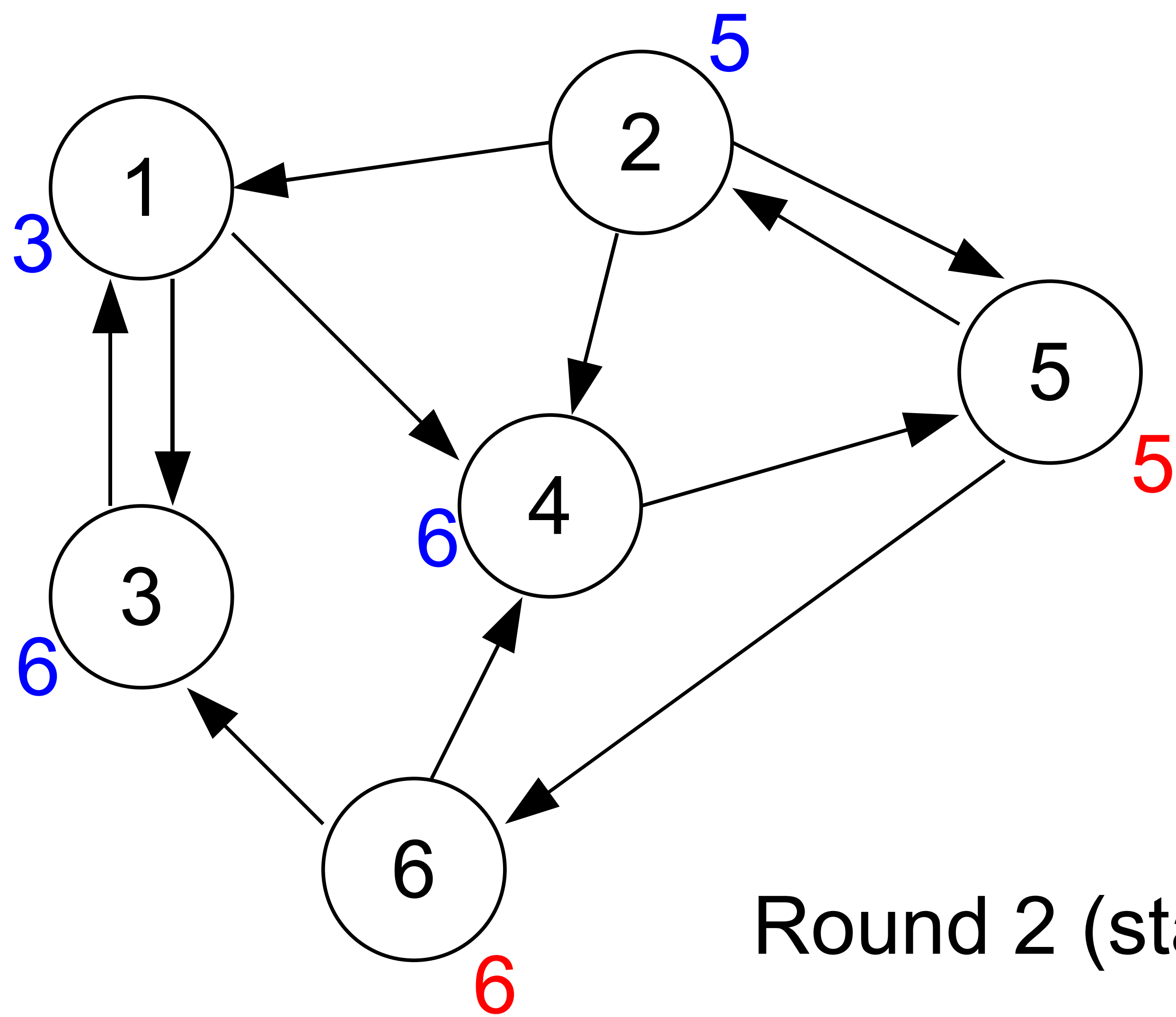
Round 1 (msgs)

Leader election in general network



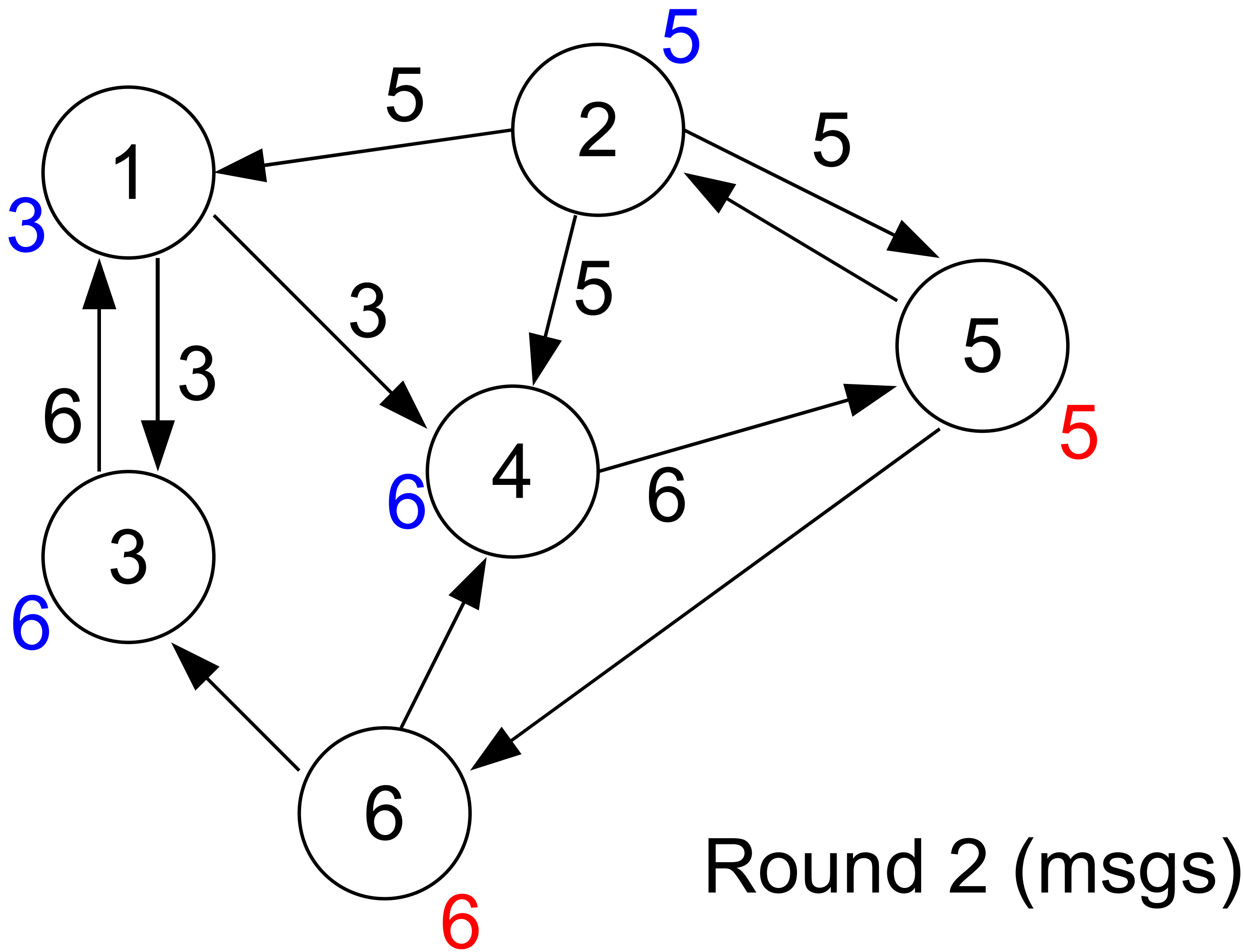
Round 1 (trans)

Leader election in general network

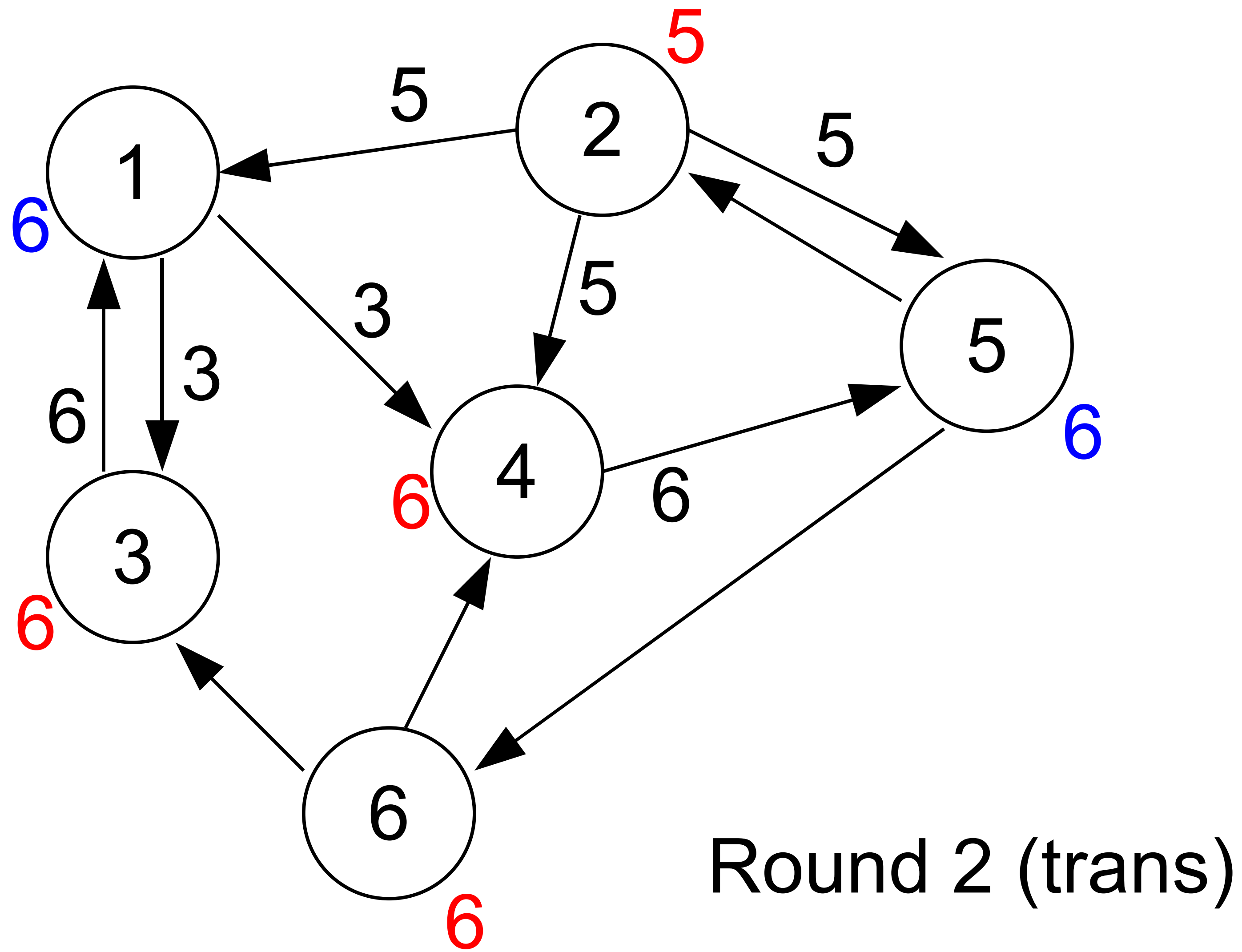


Round 2 (start)

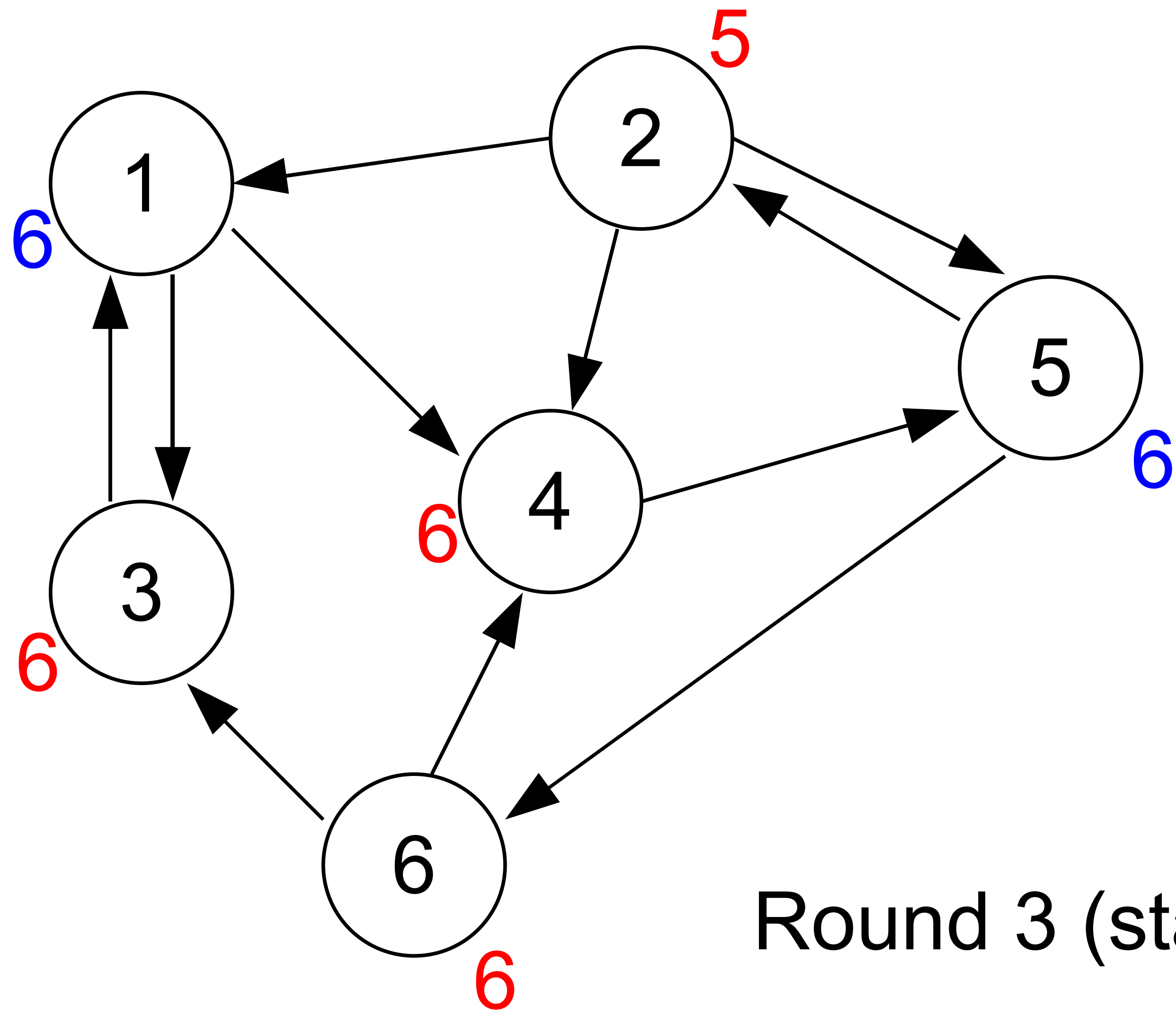
Leader election in general network



Leader election in general network

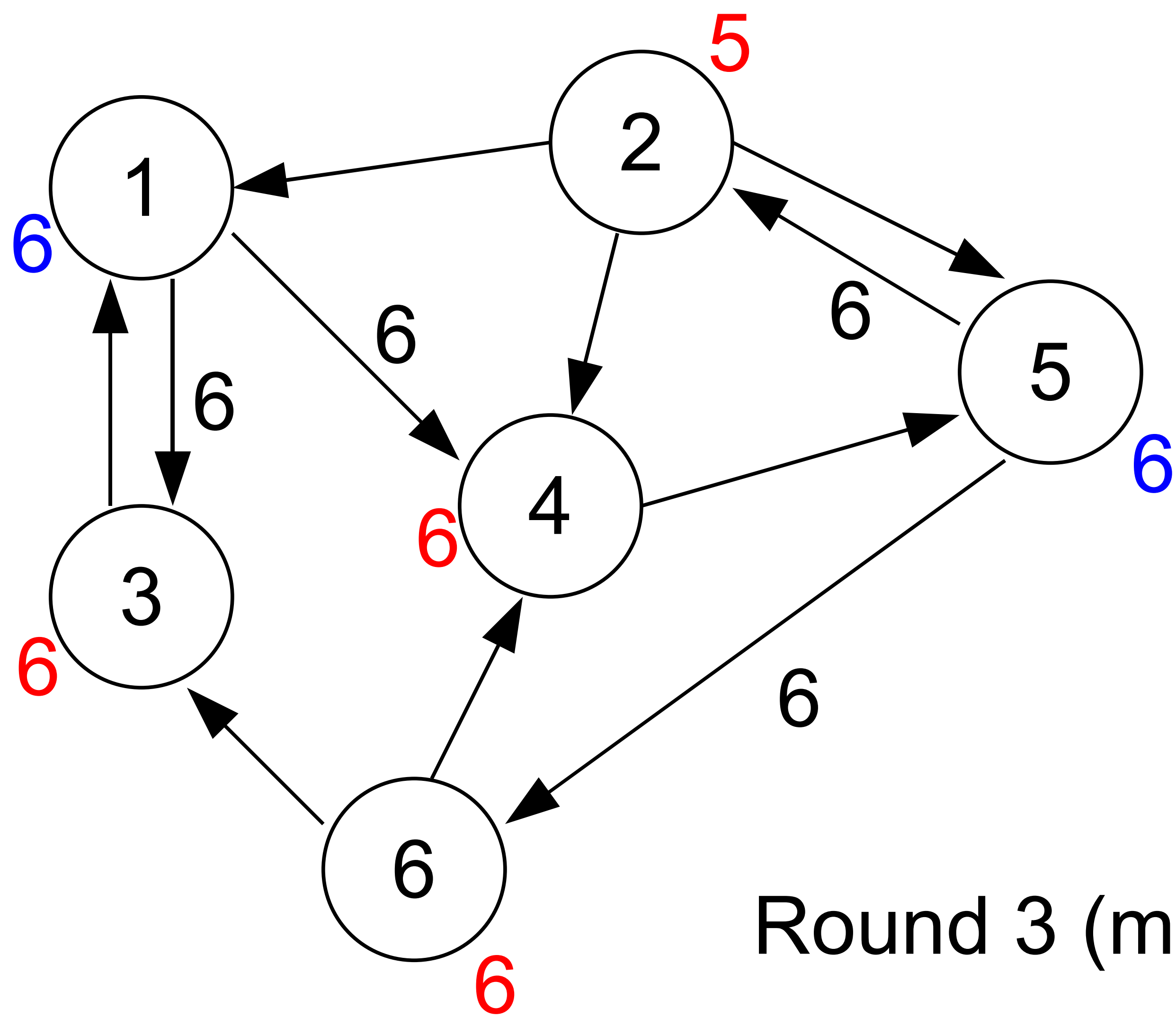


Leader election in general network



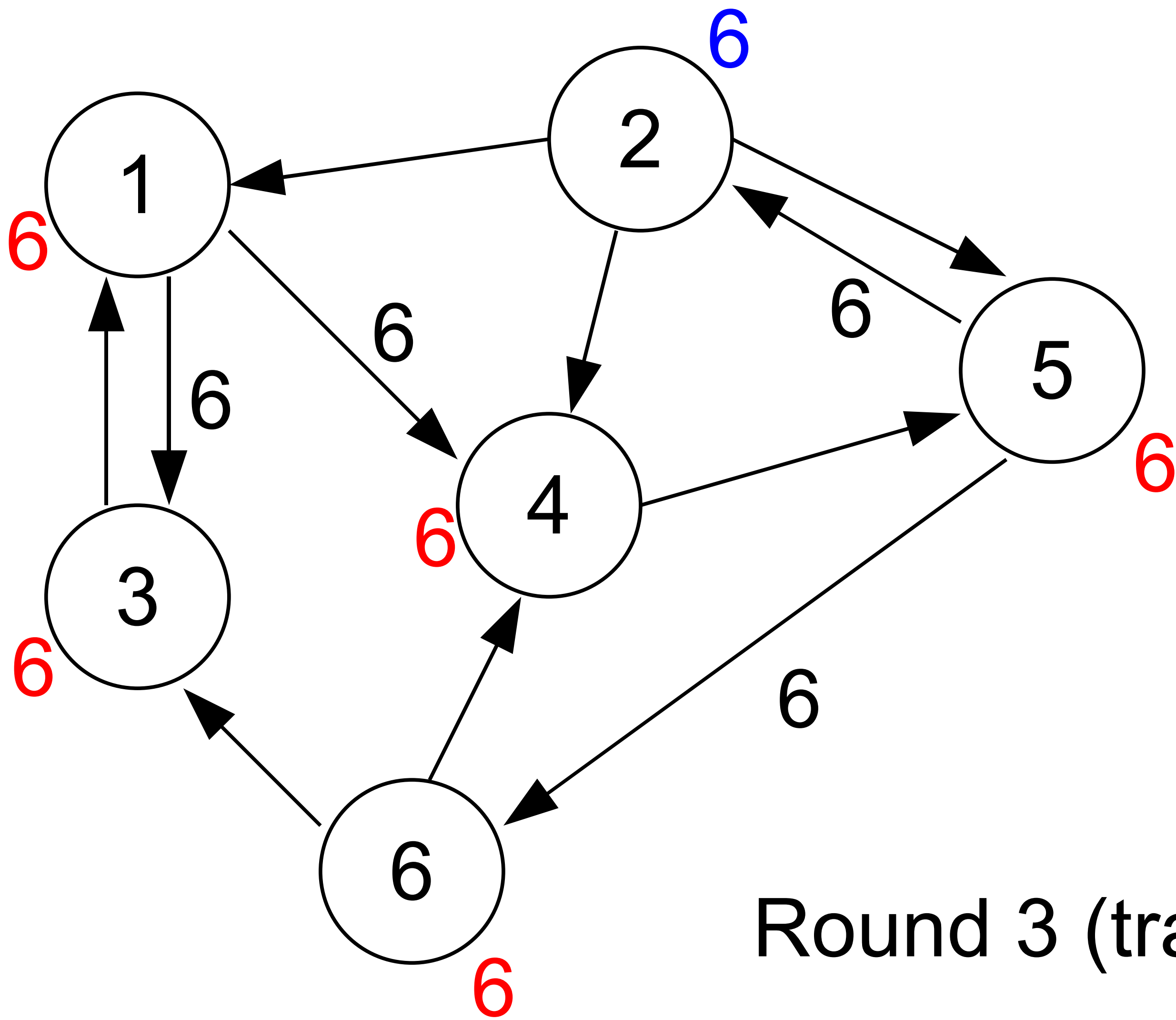
Round 3 (start)

Leader election in general network

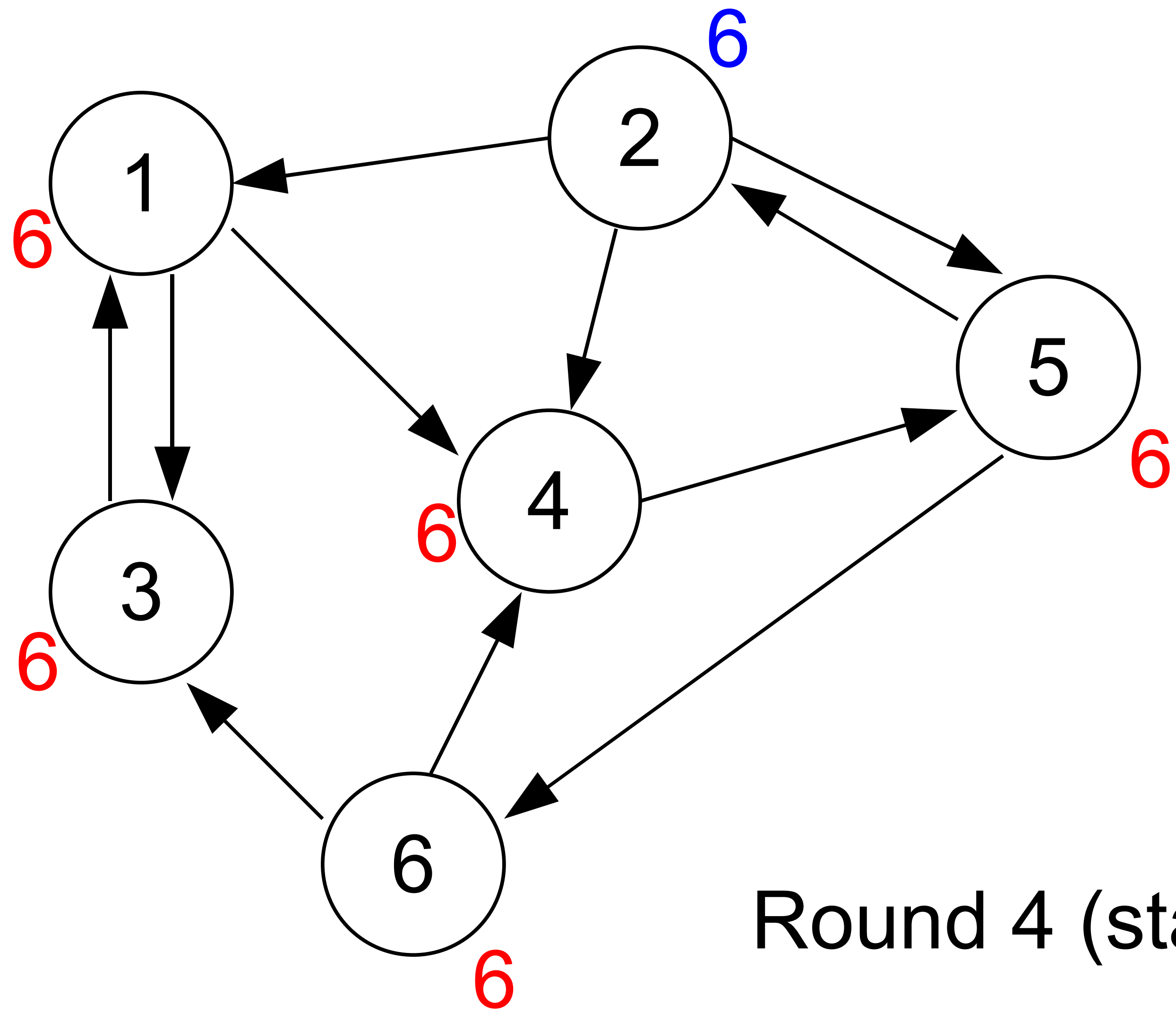


Round 3 (msgs)

Leader election in general network

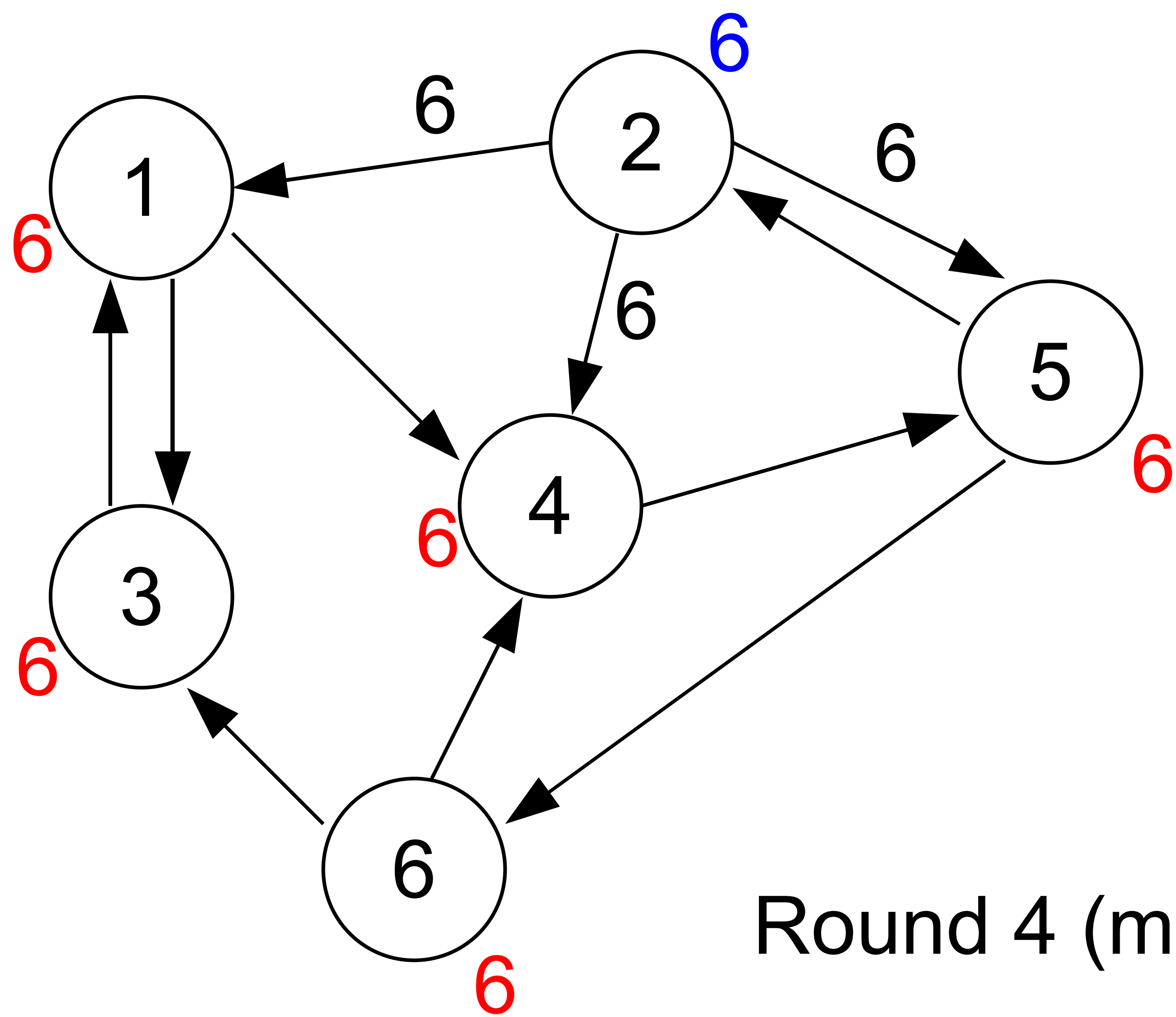


Leader election in general network



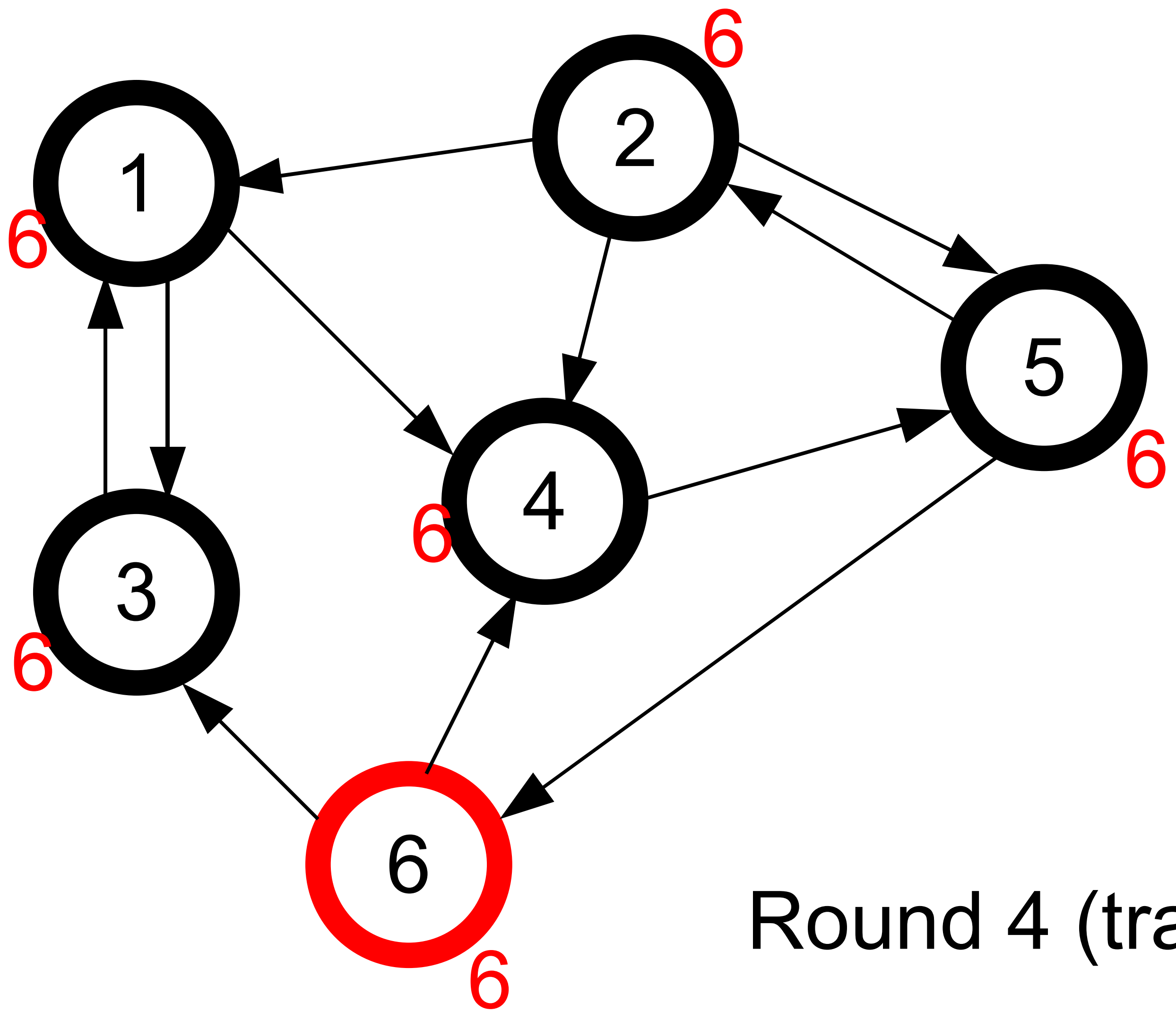
Round 4 (start)

Leader election in general network



Round 4 (msgs)

Leader election in general network



Round 4 (trans)

Leader election in general network

- Reducing message complexity
  - don't send same UID twice
  - new state var: new-info: Boolean, initially true
  - only send max-uid if new-info = true
  - new-info := (max UID received > max-uid)
- Proof
  - repeat previous proof
  - simulation

Simulation relation

- "Run two algorithms side by side"
- Define *simulation relation* between states
  - satisfied by start states
  - preserved by every transition
  - outputs should be the same in related states

Simulation relation

- All state variables in original are the same in both algorithms
- Base case: by definition
- Inductive step
- Invariant:
  - If i is in-nbr of j and $maxuid_i > maxuid_j$ then new_i = true
  - prove by induction

What's with the proofs?

Next week

- Breadth-first search
- Shortest paths
- Spanning trees

Non-comparison-based algorithms

- Can we reduce msg complexity if we aren't constrained to comparison-based algorithms?
- Consider the case where:
  - n is known
  - UIDs are positive integers
- Algorithm:
  - Phase 1, 2, 3,....; n rounds each
  - Phase k exclusively dedicated to UID k
    - Process with UID k sends it on first round of phase k then become leader and halt (elects min)
    - Other processes pass it on, then halt (not leader).
  - Msg complexity: n
  - Time complexity: u_{min} n

Non-comparison-based algorithms

- What if we don't know n?
- VariableSpeeds algorithm in book
  - UID k moves one hop every 2^k rounds
  - propagate only smallest seen so far
  - msg complexity: O(n)
  - time complexity: $O(n\ 2^{u_{min}})$

- What if we know more?

Leader election in general network