Deterministic Protocols in the SINR Model without Knowledge of Coordinates

William K. Moses Jr.
Work done in collaboration with Shailesh Vaya

06-12-16

Talk Plan

- Motivate SINR model and set it up.
- Multi-Broadcast - problem statement + related work + motivation.
- Achieving multi-broadcast.
- Wakeup - problem statement + overview of solution.
- Achieving wakeup.
- Briefly mention other results obtained.
- Possible future directions.

Outline

Outline

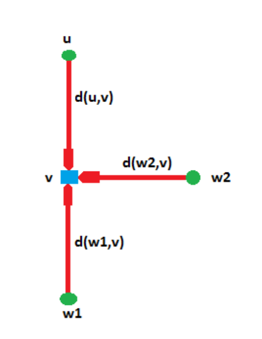
Outline

Motivation

Outline

SINR Model - In a Nutshell

Example Network



- u trying to transmit to v. Needs to overcome signals from $w1$ and $w2$.
- Strength of signal of u at v $= \frac{P_u}{d(u,v)^\alpha}$. Similar for $w1$ and $w2$.
- In SINR model, u's message is received by v iff

$$\frac{\frac{P_u}{d(u,v)^\alpha}}{\mathcal{N} + (\frac{P_{w1}}{d(w1,v)^\alpha} + \frac{P_{w2}}{d(w2,v)^\alpha})} \geq \beta.$$

SINR Model - General Case

Model Parameters

1. Path loss constant - $\alpha \geq 2$
2. Threshold constant - $\beta \geq 1$
3. Ambient noise - $\mathcal{N} \geq 0$

Following inequality determines if a message from u will be received by a station v. Let $\mathcal{T}$ be set of stations transmitting in given round.

- $\dfrac{\frac{P_u}{d(u,v)^\alpha}}{\mathcal{N} + \sum_{i=\mathcal{T}\setminus u} \frac{P_i}{d(i,v)^\alpha}} \geq \beta$

Addition of Weak Devices

Model Parameter Added

- Sensitivity parameter - $\epsilon > 0$

An additional inequality now also helps determine when a message from u will be received by a station v.

- $\frac{P_u}{d(u,v)^\alpha} \geq (1+\epsilon)\beta\mathcal{N}$

Notations

1. Range of a station u (r) - the distance from u within which another station can hear a message from u if all other stations are silent.
2. Communication graph G(V,E) - Wireless stations are nodes. If station v within range of u, there is an edge from u to v. (*Weak links*.)
3. Uniform network - ranges of all stations (and by extension powers) same and equal to r.

Settings

Network & Clock Set Up

1. Alg. works synchronously in rounds.
2. Nodes located on 2D Euclidean plane.
3. Transmission power - fixed & uniform.
4. Size of message - $O((\Delta \log N + n) \log N)$ bits
   Δ - max. degree of any node
   n - no. of nodes
   N - $[N]$ is the range from which node labels (IDs) are taken
5. Assume even with weak devices, communication graph connected.

Settings

Properties of Stations

1. A node can act either as a receiver or sender in a round, but not both.
2. No collision detection, i.e. in a given round, a receiving node can't tell if no one sent a message or too many sent a message.
3. Labels (IDs) of all nodes unique and taken from $[1, N]$.
4. Every node knows value of N, the no. of nodes in the network n, and own label.
5. No idea about Euclidean coordinates. No idea about nbrs in comm. graph.
6. At start of multi-broadcast - all nodes initially active (awake).
7. At start of multi-wakeup - some nodes initially active (awake).

Outline

Outline

Problem Statement

Problem Statement

Initially, there are several nodes, each with a different piece of information. Every piece of info must be transmitted to all other nodes within the network.

Motivation

Related Work

- SINR model has been looked at quite a bit [Avin et al., 2009, Fanghänel et al., 2009, Kesselheim and Vöcking, 2010, Kesselheim, 2011].
- Specifically the problem of broadcast [Goussevskaia et al., 2008, Yu et al., 2011, Jurdzinski et al., 2013].
- [Jurdzinski et al., 2013] model is very close, except they have knowledge of coordinates. Alg. running time - $O(D\Delta \log^2 N)$ rounds, D - diameter of communication graph.

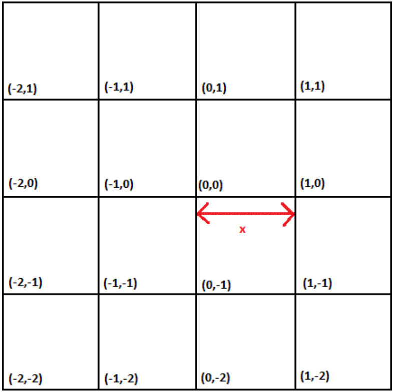
Our Contribution

Our Contribution

- We achieve deterministic multi-broadcast w/o knowledge of coordinates in $O(n \log^2 N)$ rounds, assuming all nodes awake initially.
- To the best of our knowledge, no work before ours has been able to achieve efficient deterministic multi-broadcast w/o knowledge of coordinates or knowledge of neighbors' labels.

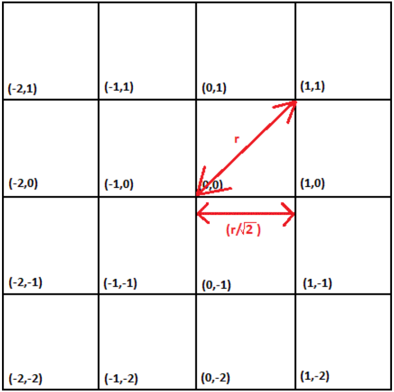
Outline

Grid



- Consider plane as a grid.
- Length of each side x.

Pivotal Grid

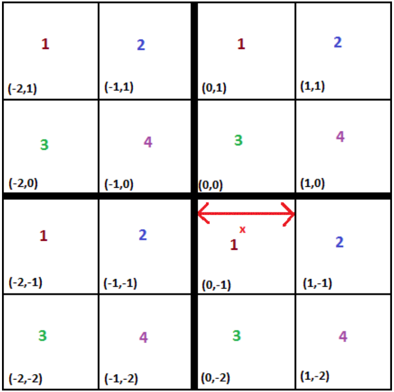


- $x = \frac{r}{\sqrt{2}}$.
- Significance: If two nodes within same box, they can hear each other.

A Simple Way to Hear a Message

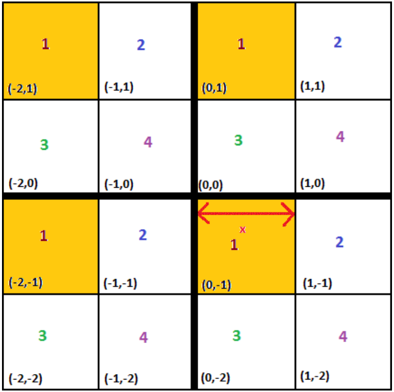
- According to SINR model, only if your signal at destination beats out interference plus noise can your message be heard.
- If there are lots of nodes, this becomes difficult.
- Solution: Develop a way to limit the number of people actually transmitting within some distance of you at any given time. Dilution.

Dilution



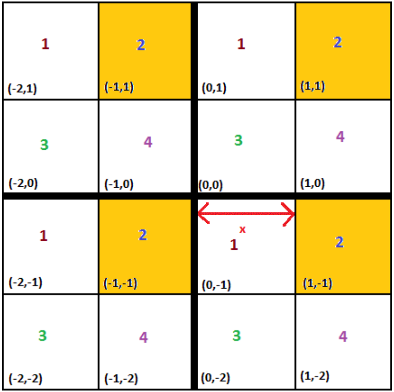
- Suppose we know grid coordinates of nodes.
- Group smaller boxes together into larger boxes.
- Example is 2-dilution.

Dilution

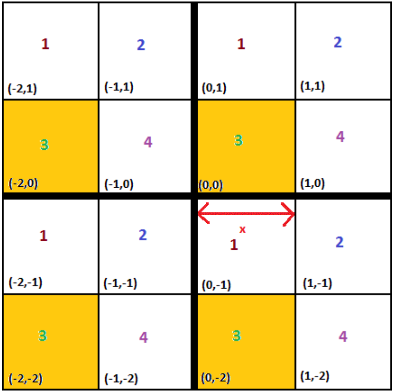


- In a given round, only one box out of 4 participates. Rest are silent.
- This chosen box is the same across all bigger boxes.

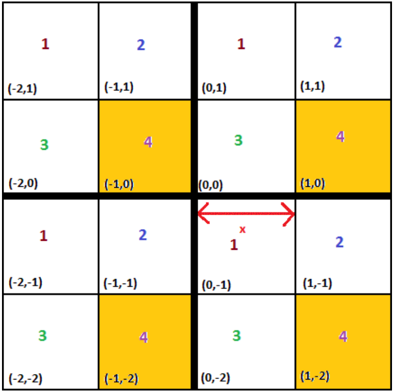
Dilution



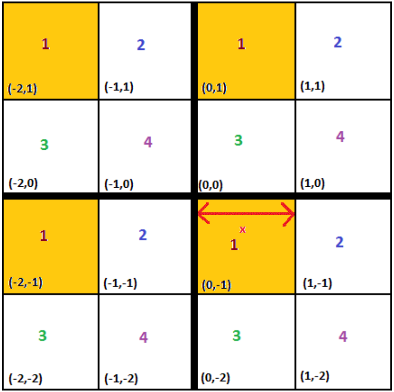
Dilution



Dilution

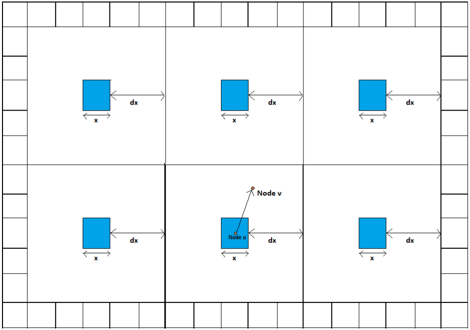


Dilution



- And this cycle of active boxes is repeated.
- We can guarantee a box is active once every 4 rounds.

Recap of Dilution



- $(2d + 1)$-dilution.
- Consider plane as a grid. Group $(2d + 1)^2$ boxes into a larger box.
- In one round, only one of those smaller boxes active per larger box. Same box in every larger box.
- v can now hear message of u.

Grids & Dilution

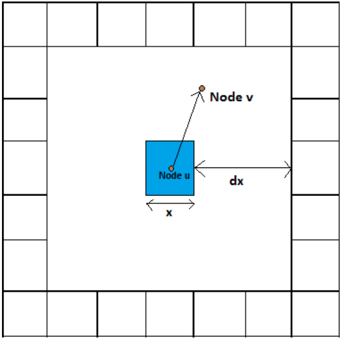
- We divide the plane into a grid. If we enforce a δ-dilution, it means that for a group of δ^2 boxes, only one of them is active at any given time. Moreover, it is the same box in each group of δ^2 boxes.
- Make the boxes small enough so that only node per box.
- Q: We're done then, right?
- A: No. Grid-based dilution is only possible when we know coordinates of each node. We don't.
- So our goal then: Silence boxes within some distance of a given box.
- How: Strongly selective families.

Strongly Selective Family

(N, c)-ssf

A family F of subsets of $[N]$ is an (N, c) strongly selective family if for every non-empty subset Z of $[N]$, such that $|Z| <= c$, and for every $z \in Z$, there is a set $f \in F$ that intersects Z at only element z. The number of subsets in the family is $O(c^2 \log N)$.

- Each subset represents set of nodes transmitting in that round.
- To complete one execution of an (N, c)-ssf, it takes $O(c^2 \log N)$ rounds.



- If we know there is at most one node per box, it's sufficient.
- We are now able to silence area of $(2d + 1)^2$ boxes except box with u.
- v can now hear message of u.

Comparison

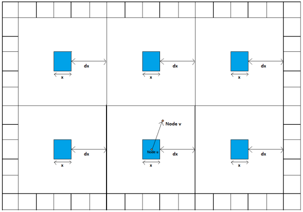


Figure: Grid-Based Dilution

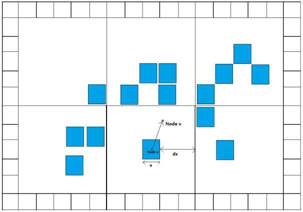


Figure: SSF-Based Dilution

SSF-Based Dilution

Issues

- Issue 1: We don't know in which round v will be able to hear u.
- Issue 2: There may be more than one node per box.
- Issue 3: Outside interference may complicate things.

Solutions

- Issue 1 & 2 taken care of by our algorithm.
- Issue 2 & 3 addressed by our lemma (modified form of proposition from [Jurdzinski et al., 2013]).

SSF-Based Dilution

Lemma

For stations with same range r, sensitivity $\epsilon > 0$, and transmission power, for each $\alpha > 2$, there exists a constant d, which depends only on the parameters α, β, and ϵ of the model and a constant k, satisfying the following property.

Let W be the set of stations such that at most a constant k of them are present in any grid box of the grid G_x, $x \leq \frac{r}{\sqrt{2}}$. Let u and v be two stations in different grid boxes such that the distance between them, $\sqrt{2}x$, is the minimum distance between any two stations in different grid boxes in G_x. Let A be the set of stations in u's grid box.

If u is transmitting in a round t and no other station within its box or a box less than d box distance away from its box is transmitting in that round, then v and all stations in A can hear the message from u in round t.

Proof Sketch: If d is large enough, the interference by outside nodes will not cause problems.

SSF-Based Dilution

Significance of Lemma

- No matter how small you make the grid boxes, the number of grid boxes you need to silence, $(2d+1)^2 - 1$, only depends on α, β, ϵ, and k.
- The first 3 are parameters of the model. If you can ensure k is a constant, then you only need to shut down a constant number of boxes.
- $\therefore$ (N, c)-ssf, where $c = k^2(2d+1)^2$, takes $O(c^2 \log N) = O(\log N)$ rounds to execute.
- $\therefore$ Can replace any dilution scheme (req. knowledge of coordinates) with our scheme with an additional factor of $O(\log N)$ rounds.

Very Important Theorem

For a grid G_x, $x \leq \frac{r}{\sqrt{2}}$, let the set of all nodes that want to transmit satisfy the properties of the Lemma. Every node in this set can successfully transmit a message to its neighbors within $\sqrt{2}x$ distance of it in $O(\lg N)$ rounds by executing one (N, c)-ssf, where $c = k^2(2d+1)^2$ where d is taken from the Lemma and k is an upper limit on the number of nodes from the set in any box of the grid.

Outline

Overview of Approach

1. Tree Grower
   - Creates trees which may span multiple grid boxes with at most one root per grid box.
   - Running time - $O(n \log N)$.
2. Tree Cutter
   - Cuts the trees to height at most 1.
   - Running time - $O(n \log^2 N)$.
3. Broadcast
   - Takes a message at one of the nodes and spreads it throughout the network.
   - Uses Tree Grower & Tree Cutter as subroutines.
   - Running time - $O(n \log^2 N)$.

Outline

Tree Grower

- Input: A network of connected nodes.
- Output: A forest of trees such that:
  - Every node is either a leader (root) or a child.
  - There is at most one leader per grid box of the pivotal grid (length of side of grid box $= \frac{r}{\sqrt{2}}$).

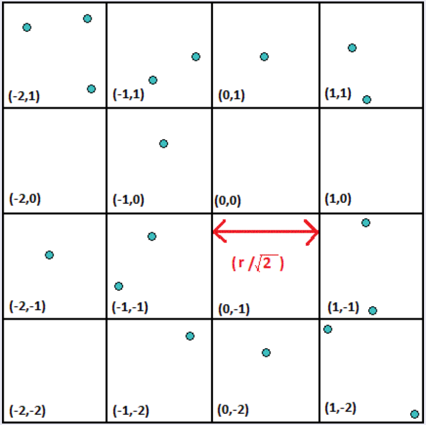
Tree Grower

Tree Grower, run by each node u

```
for cnt ← 1, n do
    if u is active then
        Execute SSF:
            Transmit u's label. Listen for others' labels.
        Execute SSF:
            Transmit info about everyone u heard. Figure out who u can bidirectionally communicate with.
        Execute SSF:
            Transmit u's label in active rounds. Listen for others who u can bidirectionally communicate with.
        Execute SSF:
            Transmit info about who might be u's potential parent or children. Lower labels becomes parents.
        Form links. If u becomes a child, become inactive.
    end if
end for
If active, become leader.
```

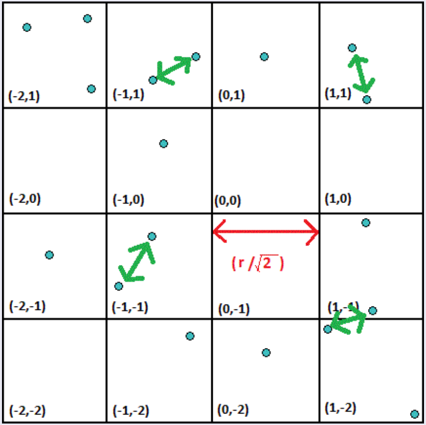
Tree Grower

TG - Initially



Tree Grower

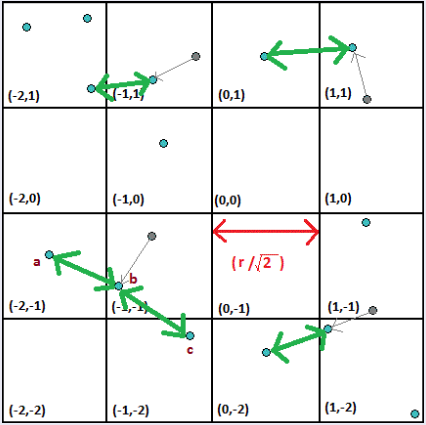
TG - Step 1



- Nodes which are close enough, hear each other thanks to SSF-based dilution and our lemma.

Tree Grower

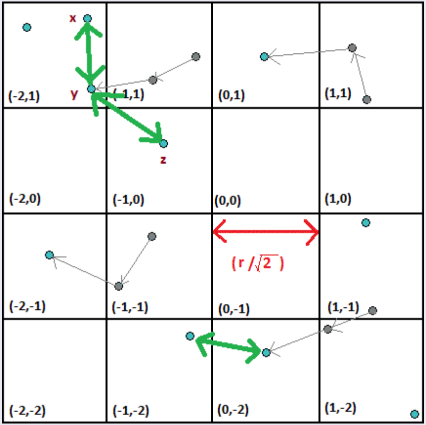
TG - Step 2



- A series of connections, where $a < b$ and $b > c$.

Tree Grower

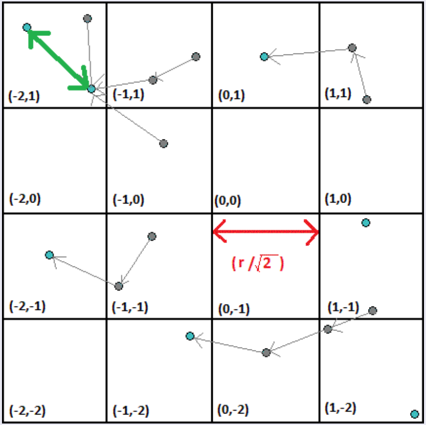
TG - Step 3



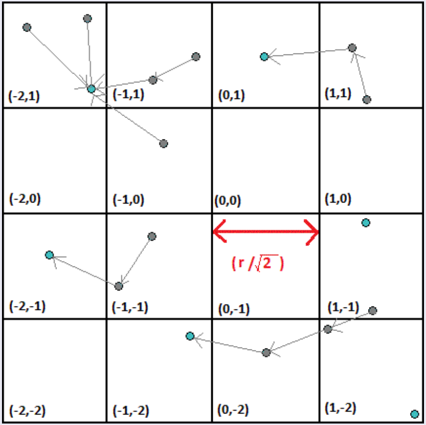
- A series of connections, where $x > y$ and $y < z$.

Tree Grower

TG - Step 4



TG - Finally



- Every node either leader or child.
- At most one leader per grid box.

Tree Grower - Proof of Claims

Claims

1. Creates trees which may span multiple grid boxes with at most one root per grid box.
2. Running time - $O(n \log N)$ rounds.

Proof Sketch of Correctness (Claim 1)

- If two nodes are both active and distance between them is min. among all distances in graph, they will be able to communicate because of ssf-based dilution.
- At the beginning of every phase i of the alg., either at least $i - 1$ nodes have become children or all nodes which are not children will become leaders.

Tree Grower - Proof of Claims

Claims

1. Creates trees which may span multiple grid boxes with at most one root per grid box.
2. Running time - $O(n \log N)$ rounds.

Proof of Running Time (Claim 2)

- Each execution of (N, c)-ssf takes $O(c^2 \log N) = O(\log N)$ rounds, since c is a constant.
- There are n such executions of 2 ssfs, so running time $= O(n \log N)$ rounds.

Now what?

- Why did we build trees?
- So that at most there are a constant number of nodes that want to transmit per grid box.
- What could we do now to broadcast?
- Use token passing, DFS style. Tree defined by leader. One token per tree. Transmit when you get token.
- Will it work? No.
- Why? Several trees pass through single grid box. No bound on them. What if all tokens end up in one grid box at same time. (N, c)-ssf-based dilution with constant c fails.
- Solution: Cut the trees down to height 1 or less.

Outline

Tree Cutter

- Input: Trees such that there is at most one leader per grid box.
- Output: Trees of height at most 1 such that there is at most one leader per grid box.

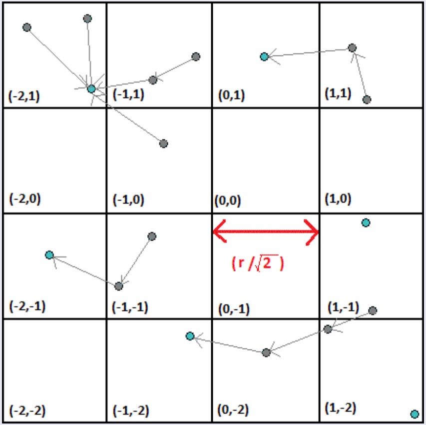
Tree Cutter

Tree Cutter, run by each node u

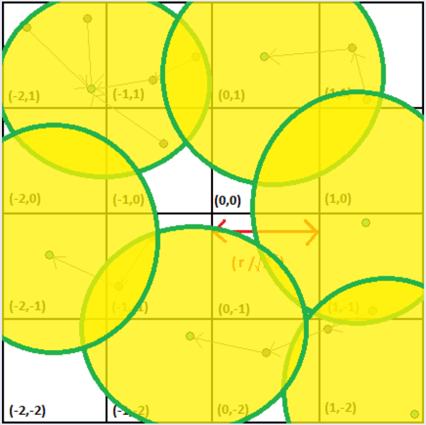
Initially, leaders have tokens.
for $cnt \leftarrow 1, 2 \cdot 947 \cdot (n+1) - 1$ **do**
    Execute Potential Leader Election.
    Execute SSF:
        Transmit u's status (leader/follower) and update u's tree.
    Execute SSF:
        Pass token, if any, in DFS manner.
end for

Tree Cutter

TC - Initially



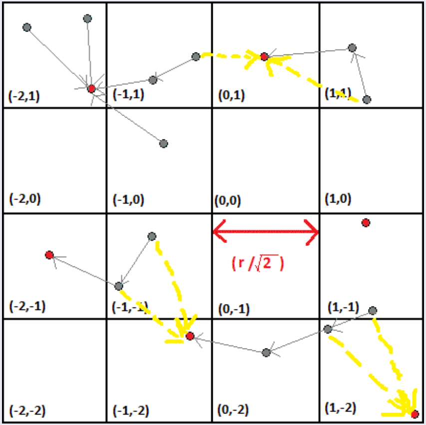
TC - Step 1



- Leaders transmit.

Tree Cutter

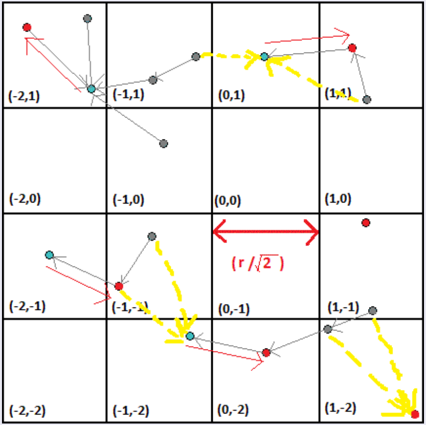
TC - Step 2



- Non-leaders orient themselves to whichever leader they hear first. If they hear from someone other than original leader first, then they reorient themselves to new leader.
- Note that non-leaders may hear from no leader initially, in which case they stay unoriented.

Tree Cutter

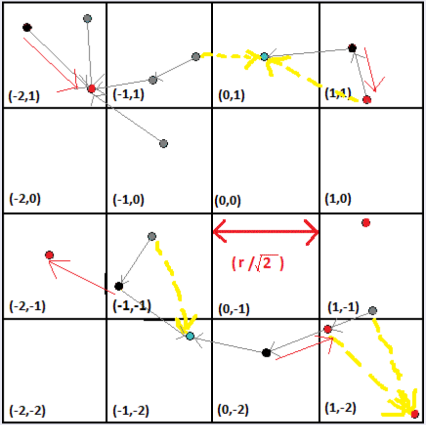
TC - Step 3



- Once a node who reorients itself gets a token from its old leader, it can declare its new allegiance.

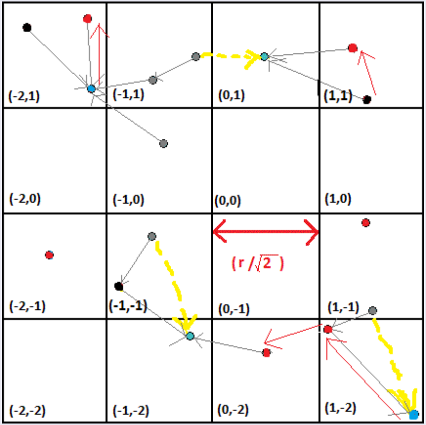
Tree Cutter

TC - Step 4



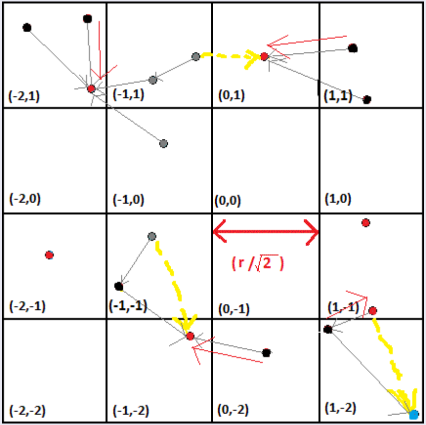
Tree Cutter

TC - Step 5

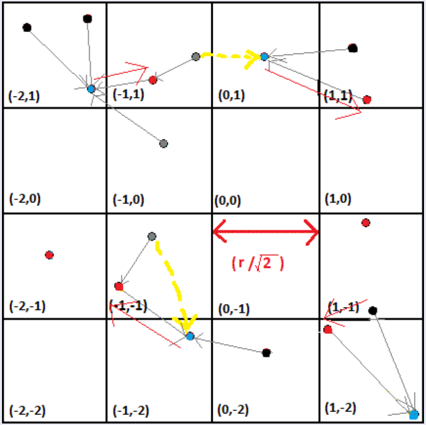


Tree Cutter

TC - Step 6

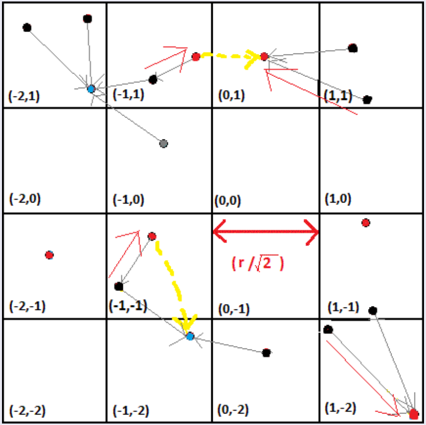


TC - Step 7



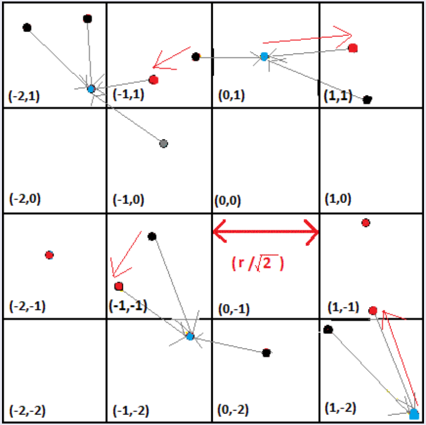
Tree Cutter

TC - Step 8



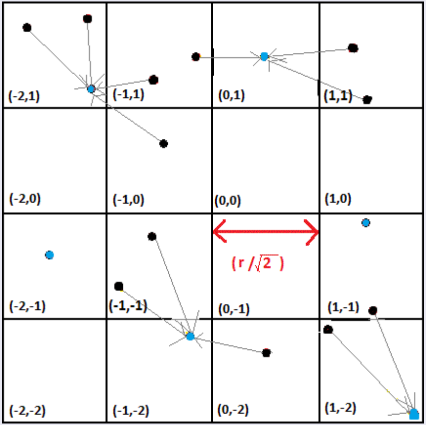
Tree Cutter

TC - Step 9



Tree Cutter

TC - Finally



- All nodes either leaders or followers.
- At most one leader per grid box.
- All trees of height 1 or less.

Tree Cutter - Potential Leader Election

- Till now we have not made use of the Potential Leader Election part of the algorithm.
- We present an example which shows how it works.
- It is only used by nodes who get the token but have not oriented themselves to a leader yet.

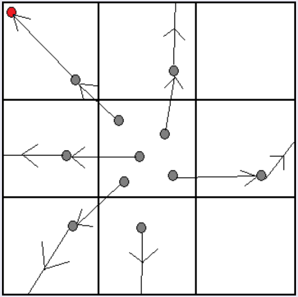
Tree Cutter - Potential Leader Election

Potential Leader Election, run by each node u

```
Execute SSF:
    Transmit u's label. Record other nodes heard in every round.
Execute SSF:
    Transmit info about which nodes u heard and which round heard in during previous ssf.
Determine which round is the round in which u **alone** transmits in the ssf. Call it R.
for i ← 1, c_1 log N do
    Execute SSF:
        If i = R and u is not follower, transmit, else stay silent.
        If u transmits, become leader. If u hears from another node v and u is not a leader/follower, become v's follower.
end for
```

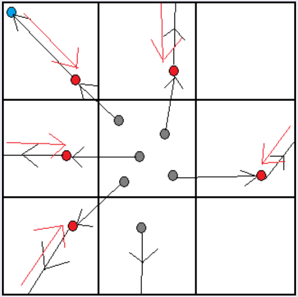
Potential Leader Election

PLE - Initially



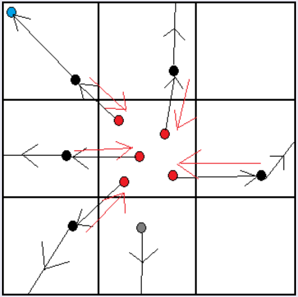
Potential Leader Election

PLE - Step 1



Potential Leader Election

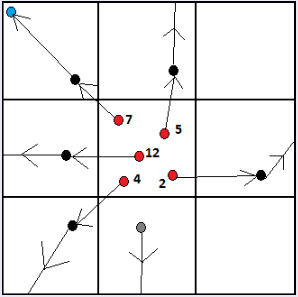
PLE - Step 2



- Now, these nodes with tokens haven't oriented themselves to any leader. So now they participate in Potential Leader Election.

Potential Leader Election

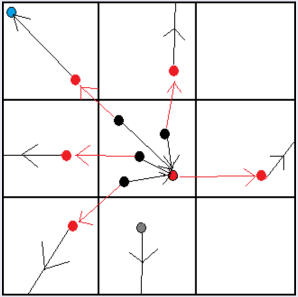
PLE - Step 3



- They've figured out the number of the round in which they alone transmit to the others.
- Notice that the grey node outside does not get involved. Reason - no token.

Potential Leader Election

PLE - Finally



- Tokens sent back. Now the leader has his own token. Others recognize him as leader and have already transmitted the same.

Tree Cutter - Proof of Claims

Claims

1. There is at most one leader per grid box of pivotal grid.
2. Cuts the trees to height at most one.
3. Running time - $O(n \log^2 N)$.

Tree Cutter - Proof of Claims

Proof Sketch of Correctness (Claim 1)

- **Goal:** At most one leader per grid box of pivotal grid (**Invariant 1** across phases of Tree Cutter).
- When is this true? When Potential Leader Election works as desired, i.e. no more than one leader created per grid box (Lemma 8).
- But for Potential Leader Election and Tree Cutter to work properly, there must be a constant no. of tokens in any grid box of pivotal grid at the beginning of every phase of Tree-Cutter (**Invariant 2**).
- Intertwined proof.
  Step 1: Assume Inv. 2 holds in every phase $\leq i$ of Tree-Cutter, then Inv. 1 holds in phase i (Lemma 9).
  Step 2: To prove Inv. 2, we need to bound the number of tokens that can get into a grid box. We do this by bounding the distance a token can move away from its leader (Lemma 10).
  Step 3: Strong induction on Inv. 2, to show it holds with help from Lemma 9 and Lemma 10.

Tree Cutter - Proof of Claims

Proof Sketch of Correctness (Claim 2)

- **Goal:** All trees cut down to height at most one.
- Using Tree Cutter & Potential Leader Election, a node will be either a leader or declare itself a follower of a leader within range of it.
- We show that after $2 \cdot 947 \cdot (n + 1) - 1$ phases of Tree Cutter, every node has a chance to transmit its status (i.e. gets to participate in Potential Leader Election if necessary).
- $\therefore$ After so many phases, all trees cut down to size.

Tree Cutter - Proof of Claims

Proof of Running Time (Claim 3)

- Potential Leader Election performs $O(c^2 \log N)$ executions of an (N, c)-ssf of size $O(c^2 \log N)$. Since c is a constant, it takes $O(\log^2 N)$ rounds.
- We perform $O(n)$ executions of Potential Leader Election. Total running time $= O(n \log^2 N)$ rounds.

Now what?

- Now we can actually perform multi-broadcast.

Outline

Multi-Broadcast

- Input: Several nodes, each with a different piece of information.
- Output: Every node has every piece of information.

- First we look at the problem of Broadcast, where only one node has info and it needs to be spread.
- After showing how this is done, we show how Multi-Broadcast is achieved in a similar manner.

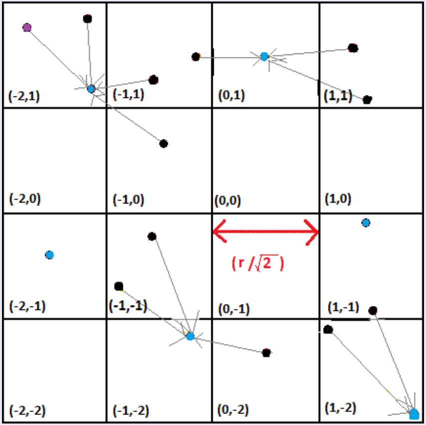
Broadcast

Broadcast, run by each node u

```
Run Tree Grower.
Run Tree Cutter.
Initially leaders have tokens.
for cnt ← 1, 4 * n do
    Execute SSF:
        If u has token and info, transmit it. Else stay silent.
    Execute SSF:
        Pass token, if any, in DFS manner.
end for
```

Broadcast

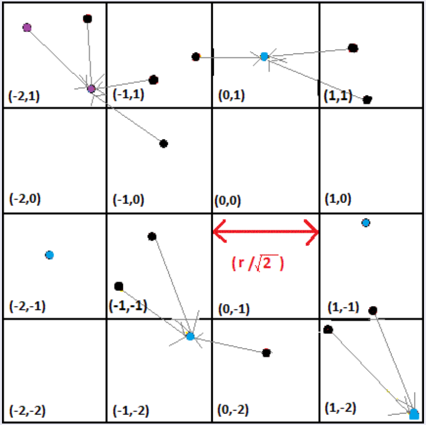
Broadcast - Initially



- Initially one node has info.

Broadcast

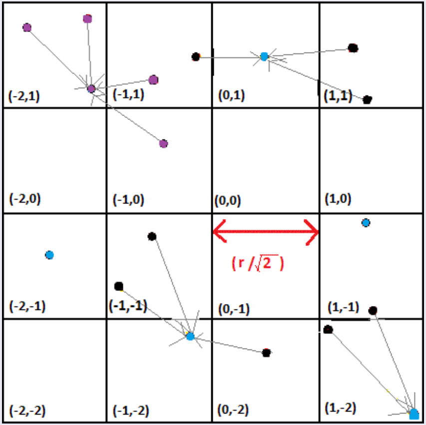
Broadcast - Step 1



- When token comes to node, it transmits and its leader hears the info.

Broadcast

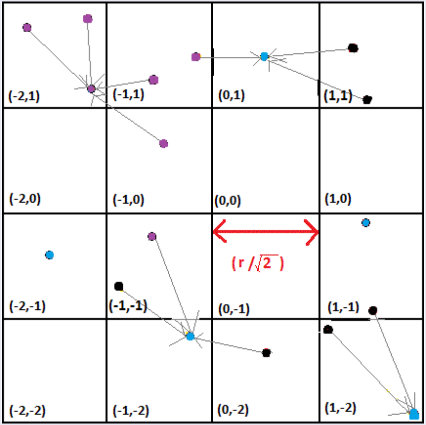
Broadcast - Step 2



- Leader gets token, transmits, and all its children get info.

Broadcast

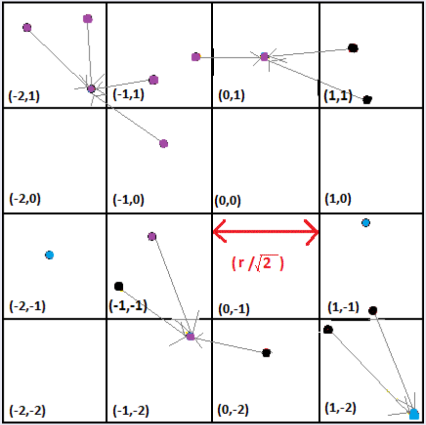
Broadcast - Step 3



- Once all children transmit, since connected graph, someone new will hear.

Broadcast

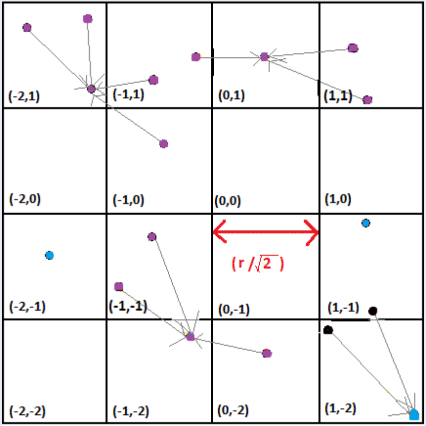
Broadcast - Step 4



- And the cycle continues...

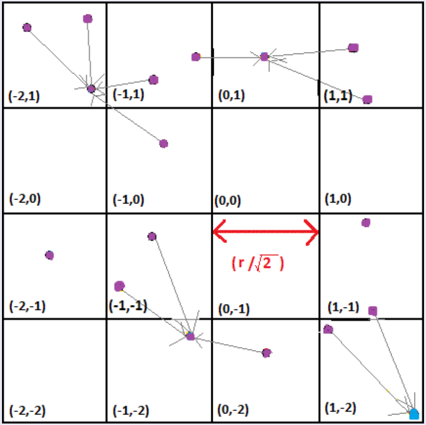
Broadcast

Broadcast - Step 5

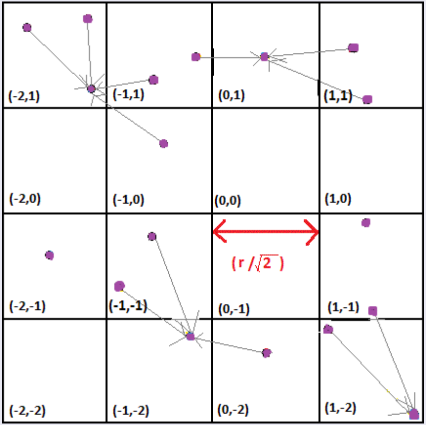


Broadcast

Broadcast - Step 6



Broadcast - Finally



- Until finally all nodes have info.

Broadcast - Proof of Claims

Claims

1. Takes a message at one of the nodes and spreads it throughout the network.
2. Running time - $O(n \log^2 N)$.

Proof Sketch of Correctness (Claim 1)

- If a node with message belongs to a tree of size s_i, it takes at most $O(s_i \log N)$ time for the node to get a token.
- Once that node transmits, it takes at most $O(s_i \log N)$ time for all nodes in that tree to transmit.
- Let us say that a tree has a message if one of its nodes has a message. If all nodes belonging to trees with messages transmit, and if there still exist nodes in the network without the message, then at least one node belonging to a new tree will now have the message because the underlying communication graph is connected.

Broadcast - Proof of Claims

Claims

1. Takes a message at one of the nodes and spreads it throughout the network.
2. Running time - $O(n \log^2 N)$.

Proof of Running Time (Claim 2)

- Each node executes two (N, c)-ssfs $4n$ times. Total running time $=$ $O(n \log N)$ rounds.
- Overall running time from start to finish $=$ $O(n \log N) + O(n \log^2 N) + O(n \log N) = O(n \log^2 N)$ rounds.

Multi-Broadcast

- In Multi-Broadcast, more messages.
- Large enough message size ensures that even if there are n pieces of information, still only one round reqd. to transmit them.
- Each piece of info takes same time to traverse network.
- Therefore, just have nodes always transmit messages (if they have one) when they get token.
- Running time is same, i.e. $O(n \log^2 N)$ rounds from start to finish.

Outline

Outline

Problem Statement

Problem Statement

Initially, only a subset of the total nodes are awake. Our goal is to wake them all up.

Our Contribution

Our Contribution

We achieve deterministic non-spontaneous wakeup w/o knowledge of coordinates in $O(n \log^2 N \log n)$ rounds.

Outline

Overview

- We use Tree Grower, Tree Cutter, and another algorithm called Token Passing Transfer as subalgorithms in our main algorithm, which we call Multi-Wakeup.
- We also use techniques of slotting and multiplexing and the idea of epochs.

Token Passing Transfer

- Input: The trees obtained from running Tree Grower and Tree Cutter on participating nodes.
- Output: Any sleeping nodes within range are woken up.

Token Passing Transfer

Token Passing Transfer, run by each node u

1: Initially leaders have tokens.
2: **for** $cnt \leftarrow 1, 4 \cdot n$ **do**
3: Execute SSF:
4: If u has token, transmit wakeup message. Else stay silent.
5: Execute SSF:
6: Pass token, if any, in DFS manner.
7: **end for**

Slots

- When we run an algorithm in slot i, it means that we are assuming at most 2^i nodes will participate in that algorithm in that slot.
- We have $\lfloor \log n \rfloor + 1$ slots in Multi-Wakeup.

Phases

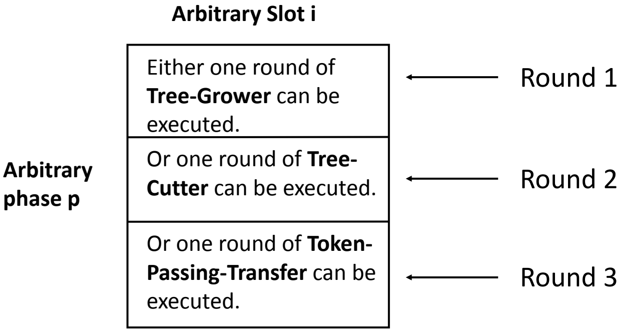


Figure: One phase of one slot for a given awake node.

Epochs

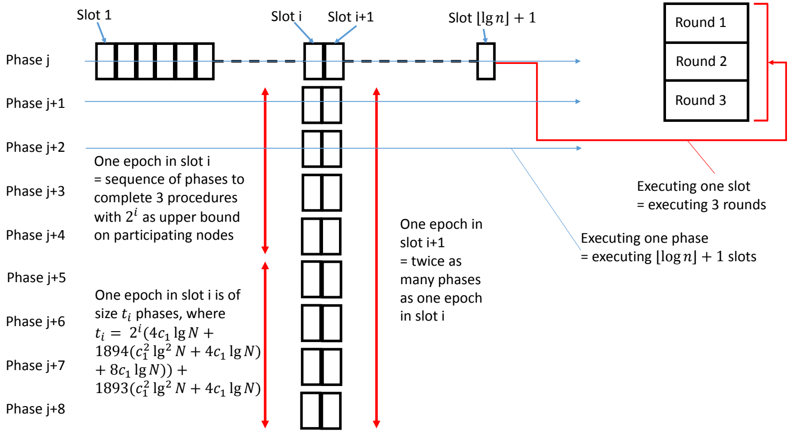


Figure: One epoch across multiple phases for a given slot.

Epochs (contd.)

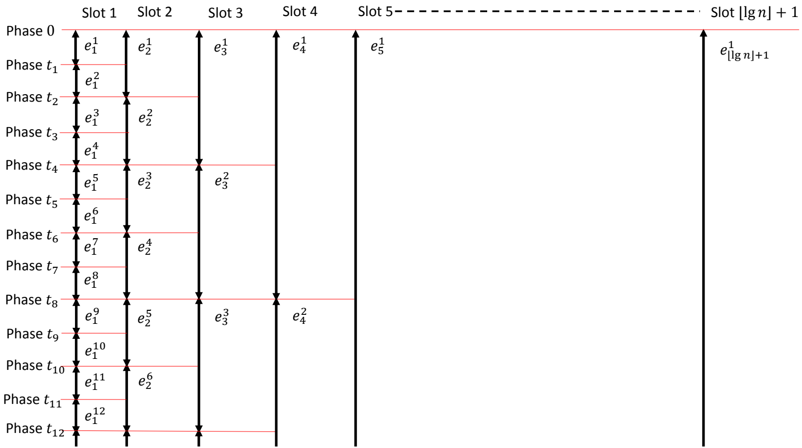


Figure: Phases, slots, & epochs together.

Outline

Overview of Approach

- Consider Tree Grower, Tree Cutter and Token Passing Transfer as one combo pack $T3$.
- As soon as a node wakes up, it begins $T3$ in each slot at the first opportunity (next epoch of that slot).
- In one of the slots, the correct assumption on no. of nodes participating will be made.
- After $T3$ finishes executing in that slot, if there are any nodes within range that are asleep, they will be woken up.
- The cycle continues until all nodes are woken up.

Outline

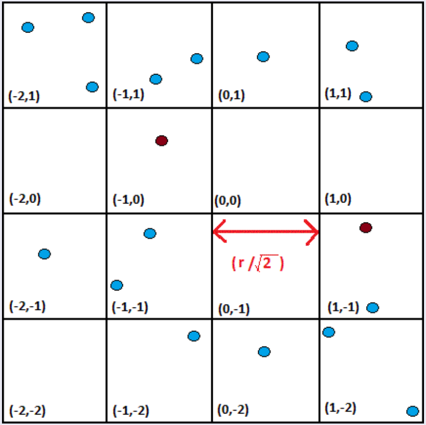
Multi-Wakeup

Multi-Wakeup, run by each node u

Initially leaders have tokens.
for *phase* $\leftarrow 0, 4nt_1 - 1$ **do**
  for *slot* $\leftarrow 1, \lfloor \log n \rfloor + 1$ **do**
    If asleep, stay silent for 3 rounds.
    Else if awake and Tree Grower not started yet or going on
      If not executing anything yet, wait (stay silent) till the start of next epoch.
      If waiting over, start executing Tree Grower. Execute one round of it.
    Else if Tree Grower over, start executing Tree Cutter. Execute one round of it.
    Else if Tree Cutter over, start executing Token Passing Transfer. Execute one round of it.
    Else if Token Passing Transfer over, stay silent for 3 rounds.
  end for
end for

Multi-Wakeup

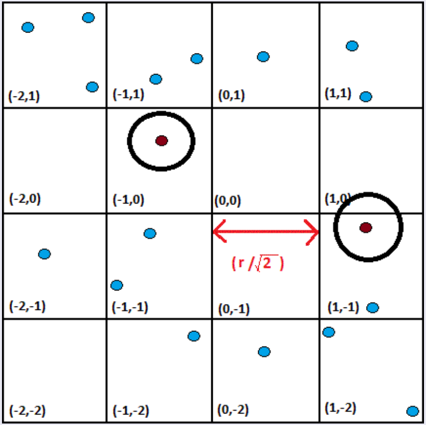
Multi-Wakeup - Initially



- Initially some nodes awake.

Multi-Wakeup

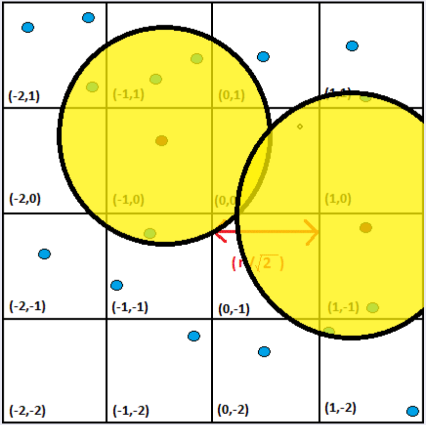
Multi-Wakeup - Step 1



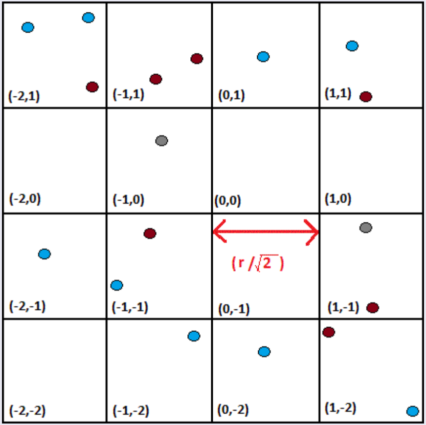
- They execute in slot 1.

Multi-Wakeup

Multi-Wakeup - Step 2



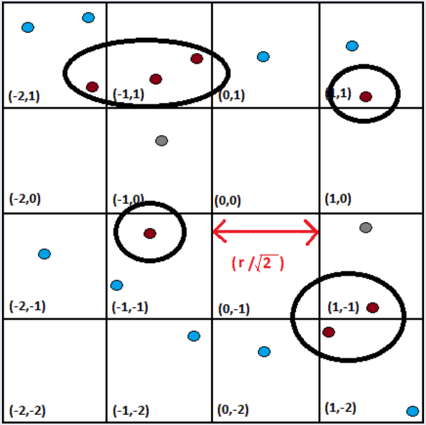
Multi-Wakeup - Step 3



- New nodes woken up.

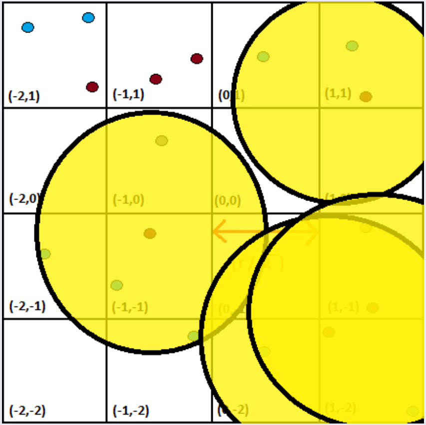
Multi-Wakeup

Multi-Wakeup - Step 4



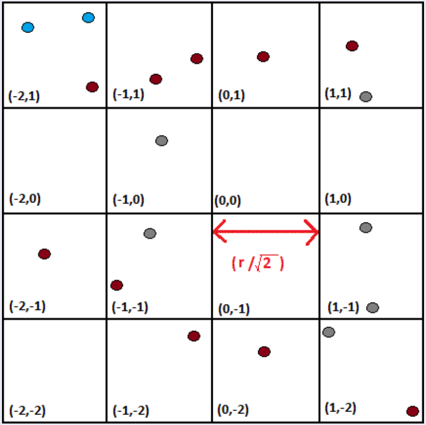
Multi-Wakeup

Multi-Wakeup - Step 5

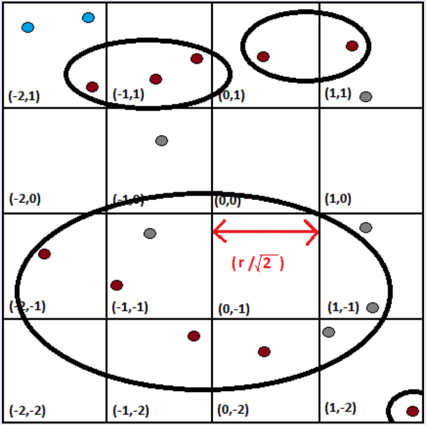


- Only those in slot 1 can execute immediately.

Multi-Wakeup - Step 6



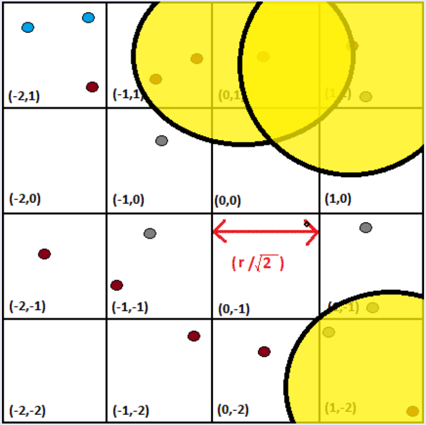
Multi-Wakeup - Step 7



- Larger group of woken up nodes absorbs nodes which haven't executed in a higher slot yet.

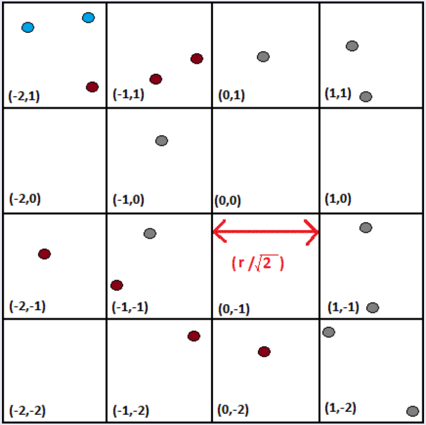
Multi-Wakeup

Multi-Wakeup - Step 8



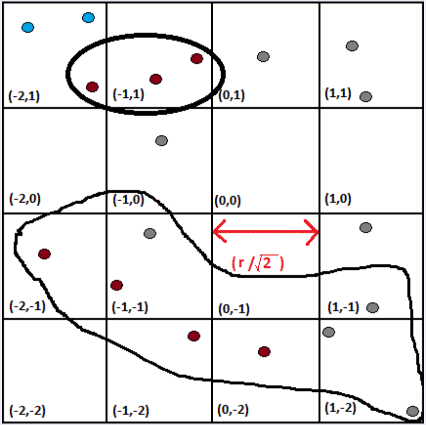
- Slot 1 execution.

Multi-Wakeup - Step 9



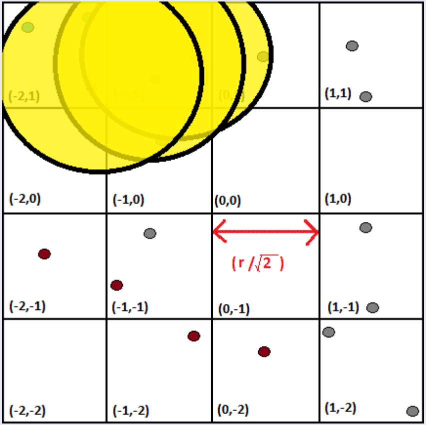
Multi-Wakeup

Multi-Wakeup - Step 10



Multi-Wakeup

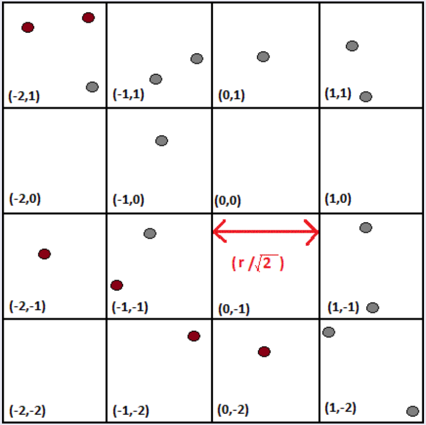
Multi-Wakeup - Step 11



- Slot 2 execution.

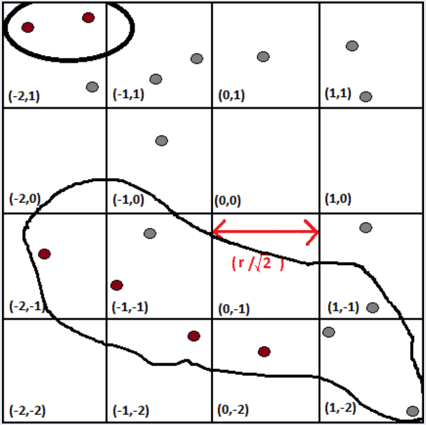
Multi-Wakeup

Multi-Wakeup - Step 12



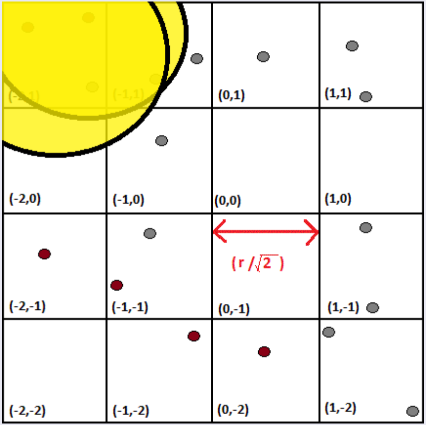
Multi-Wakeup

Multi-Wakeup - Step 13



Multi-Wakeup

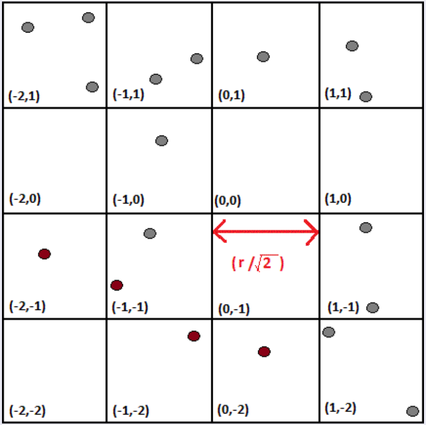
Multi-Wakeup - Step 14



- Slot 1 execution.

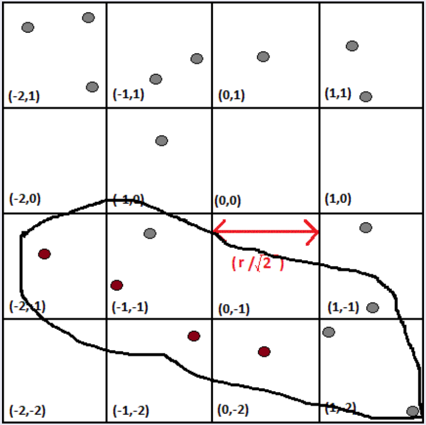
Multi-Wakeup

Multi-Wakeup - Step 15



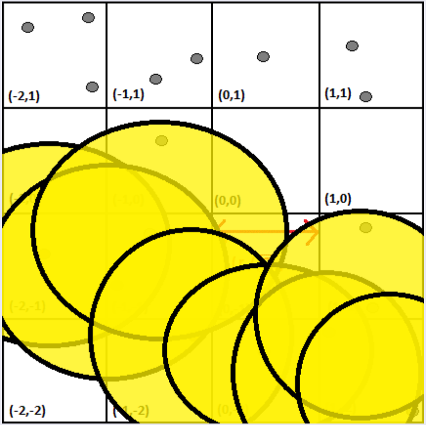
Multi-Wakeup

Multi-Wakeup - Step 16



Multi-Wakeup

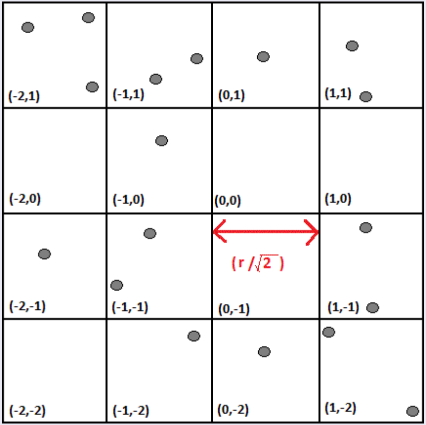
Multi-Wakeup - Step 17



- Next epoch for slot 3 arrives. Slot 3 execution.

Multi-Wakeup

Multi-Wakeup - Finally



- All nodes awake.

Multi-Wakeup - Proof of Claims

Proof Sketch of Correctness & Running Time

- The worst case scenario for running time is when nodes wake up in increasingly larger batches and cause other nodes to keep participating in larger slots.
- This worst case is covered by waiting the time period required for 2 executions of T3 in the largest slot ($\lfloor \log n \rfloor + 1$) to complete.
- This takes at most $O(n \log^2 N \log n)$ rounds.
- Running our algorithm that long ensures all nodes wake up. Also, running time of algorithm is fixed.

Outline

Other Results Obtained using SSF-based Dilution

- When knowledge of N, n, and your label is known, solved problem of deterministic creation of backbone. Running time $= O(n \log^2 N)$ rounds.
- When knowledge of N, n, your own label and your neighborhood is known, solved problem of deterministic creation of backbone. Running time $= O(\Delta \log^2 N)$ rounds.

Possible Future Research

- When only knowledge of N and your label is known, solve the following:
  1. Broadcast.
  2. Backbone creation.
  3. Non-spontaneous wake-up.

Outline

References

Olga Goussevskaia and Thomas Moscibroda and Roger Wattenhofer
Local Broadcasting in the Physical Interference Model
Foundations of Mobile Computing, 35 – 44.

Chen Avin and Zvi Lotker and Francesco Pasquale and Yvonne-Anne Pignolet
A Note on Uniform Power Connectivity in the SINR Model
Algorithmic Aspects of Wireless Sensor Networks, 116 – 127.

Alexander Fanghänel and Thomas Kesselheim and Harald Räcke and Berthold Vöcking
Oblivious Interference Scheduling
Principles of Distributed Computing, 220 – 229.

Thomas Kesselheim and Berthold Vöcking
Distributed Contention Resolution in Wireless Networks
Distributed Computing, 163 – 178.

References

Thomas Kesselheim
A Constant-Factor Approximation for Wireless Capacity Maximization with Power Control in the SINR Model
Symposium on Discrete Algorithms 1549 – 1559.

Dongxiao Yu and Yuexuan Wang and Qiang-Sheng Hua and Francis CM Lau
Distributed Local Broadcasting Algorithms in the Physical Interference Model
Distributed Computing in Sensor Systems and Workshops, 1 – 8.

Tomasz Jurdzinski and Dariusz R. Kowalski and Grzegorz Stachowiak
Distributed Deterministic Broadcasting in Wireless Networks of Weak Devices
Automata, Languages, and Programming, 632 – 644.

The End