Connectivity-Based Multi-Hop Ad hoc Positioning

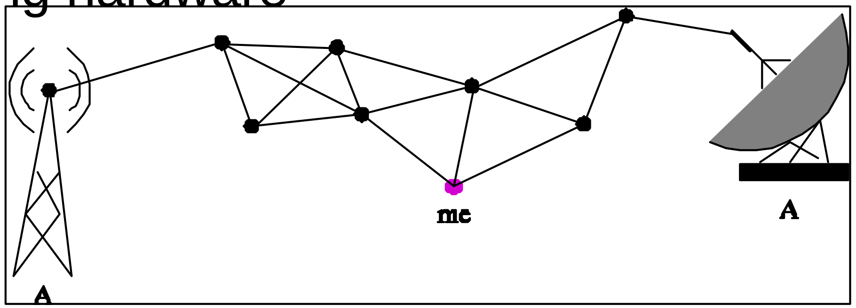
Distributed Computing Group

Regina Bischoff (O'Dell)

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Brief Introduction to Positioning

- Why positioning?
  - Sensible sensor networks
  - Heavy and/or costly positioning hardware
  - Smart dust
  - Geo-routing

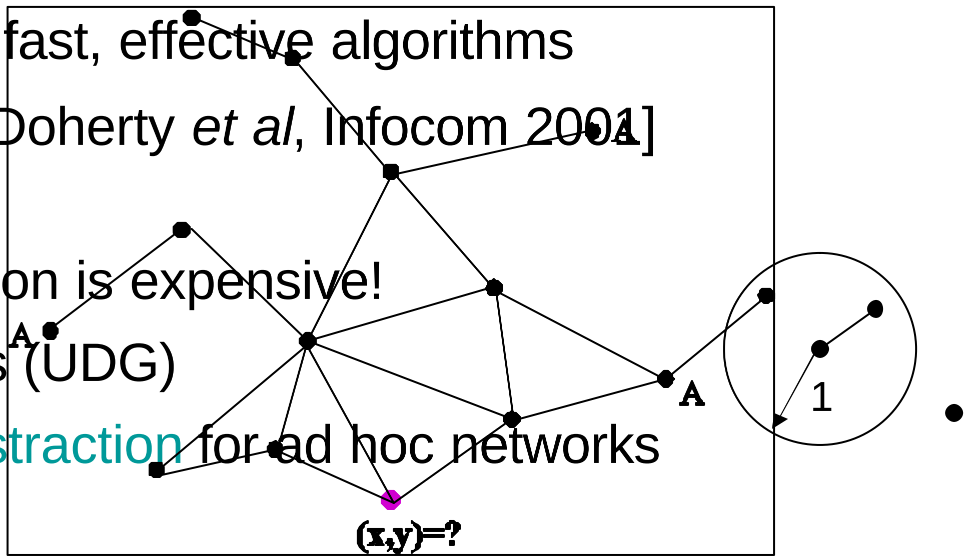


- Why not GPS?
  - Heavy, large, and expensive (as of yet)
  - Battery drain
  - Not indoors or remote regions
  - Accuracy?

- Solution: equip small fraction with GPS (anchors)

Model

- Anchors (A) know position
  - ? Virtual coordinates
- Multiple hops
  - ? Single hop: nodes hear anchors directly
  - – Allows small percentage of anchors
  - – Unavoidable?
- Ad hoc network: fast, effective algorithms
  - ? Centralized [Doherty *et al*, Infocom 2001]
  - – Scalability
  - – Communication is expensive!
- Unit Disk Graphs (UDG)
  - – Common abstraction for ad hoc networks



Model … cont'd

- Connectivity information only
  - ? T[D]oA (in GPS)
  - ? RSSI (in RADAR)
  - ? AoA (APS using AoA)
  - ? Relative distance to anchors [He *et al*, Mobicom 2003]
  - – Cheaper!
  - – Weak measuring instruments are not better:
    - [Beutel, Handbook on Sensor Networks, 2004]
    - Recent submission to [MobiHoc 2004]
- Maximum error
  - ? Average or least-squares error
  - – Worst-case analysis

Positioning Goals – In this Talk

- Hop algorithms are not enough
- Optimal algorithm in 1 dimension
  - HS algorithm
- Improved hop-based algorithm in 2 dimensions
  - GHoST algorithm framework
- Ultimate Goal ➔ better understanding of positioning

But first… General Positioning Algorithms

- Structure of connectivity-based multi-hop algorithms
  - Obtain distance in hops to (all/some) anchors – multi-hop
    - In general: obtain connectivity information
  - Local computation to estimate posi[illegible] based [illegible]
    - Gives area of all possible locations
  - Can be done incrementally
  - Can be done iteratively [Savarese *et al*, USENIX 2002]

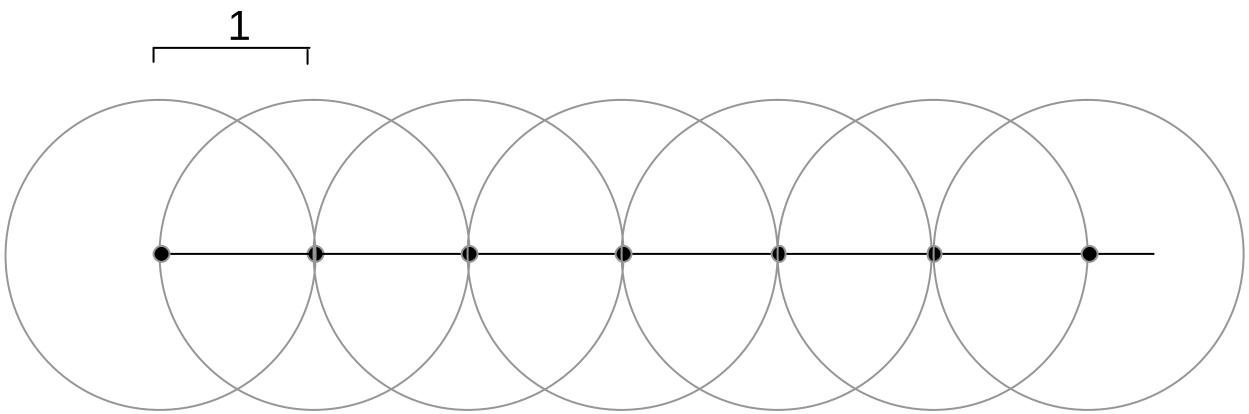
Chance for Improvement

HOP Algorithm

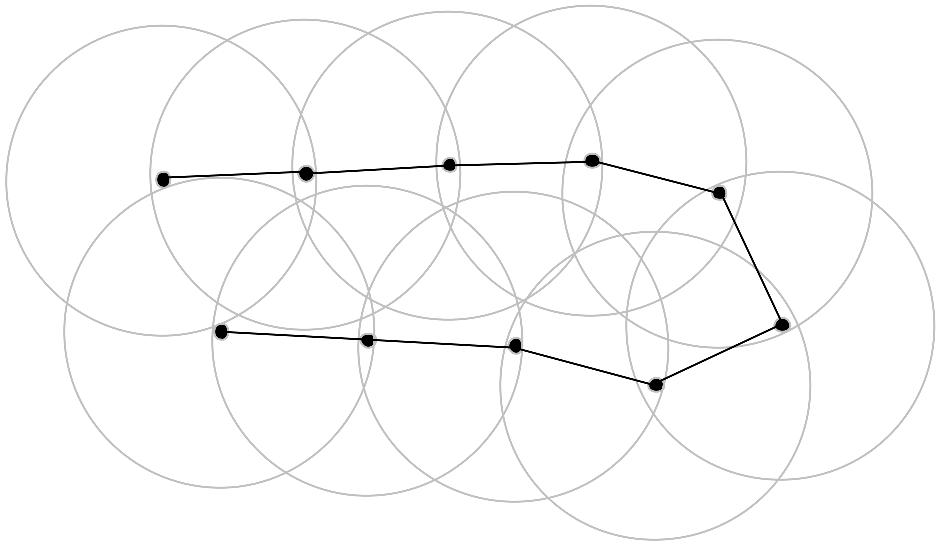
- Simple HOP algorithm:
  - Get graph distance h to anchor(s)
  - Intersect circles around anchors
    - radius = distance to anchor
  - Choose point such that maximum error is minimal
    - Find enclosing circle (ball) of minimal radius
    - Center is calculated location

HOP Algorithm … cont'd

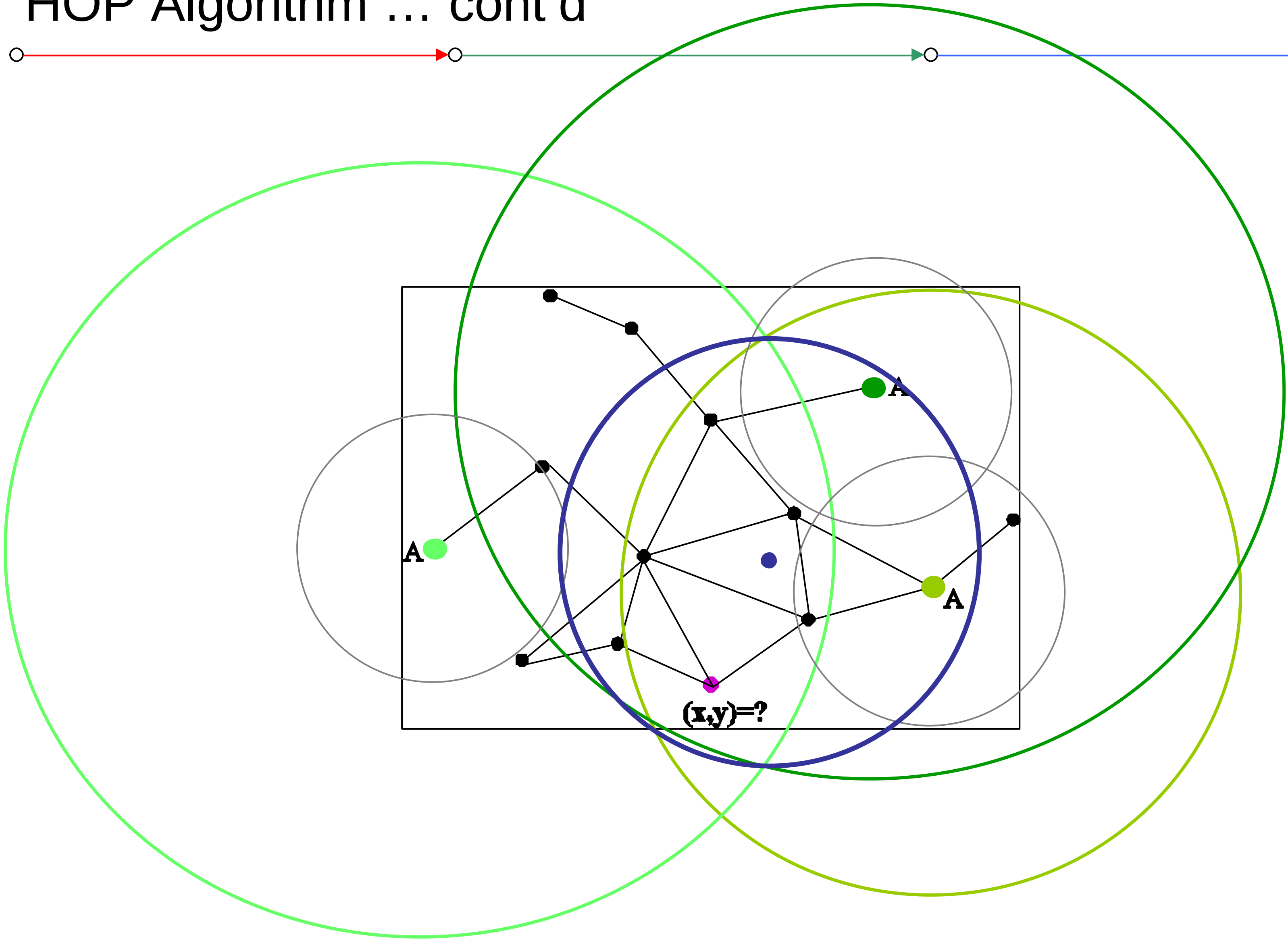
- In 1D: Euclidean distance d is bounded by $h/2 < d \leq h$



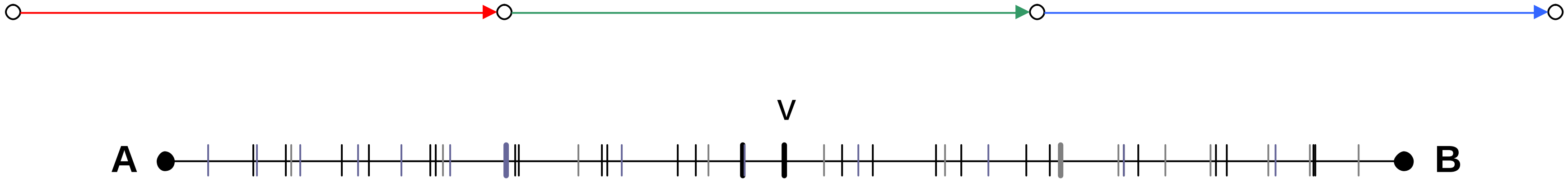
- In higher dimensions: $1 < d \leq h$



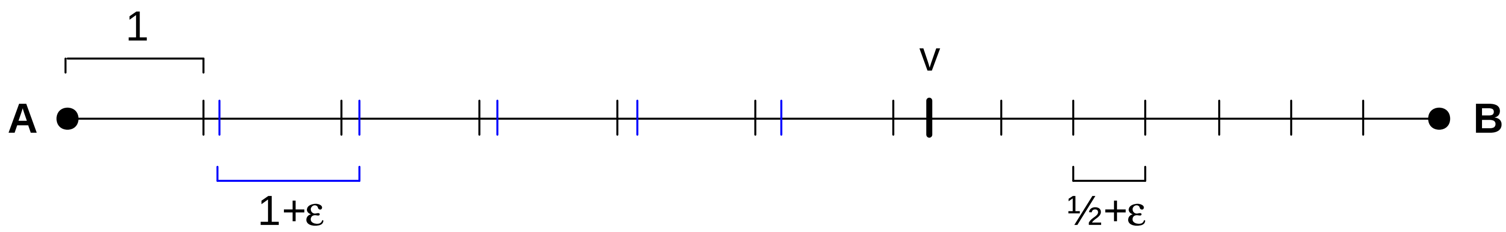
HOP Algorithm … cont’d

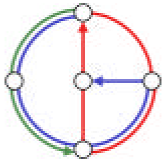


HOP is Bad

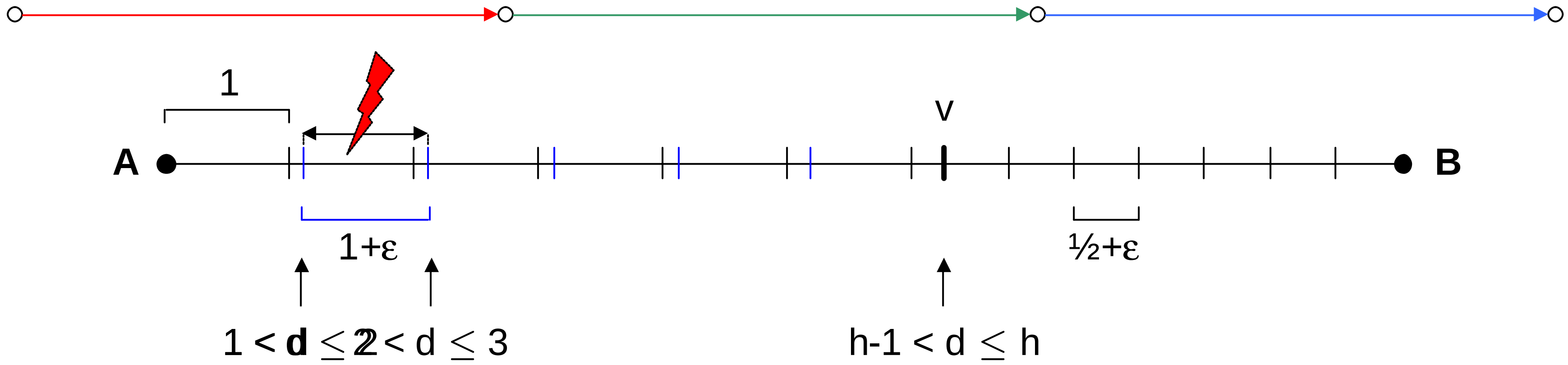


- HOP algorithm
  - Symmetric hop information ➔ place v in the middle at position d/2
  - True position ≈ h, about 2/3 d ➔ Error is almost d/6





OPT is better



- Optimal algorithm OPT (knows entire graph G = (V,E))
  - Deduces that blue nodes are Euclidean distance at least 1 apart
  - But they are also hop distance +1 from anchor **A**
  - Conclusion: actual distance d_v from **A** is $h-1 < v \leq h$

Combine hop with graph knowledge!

$Error_{HOP}$ grows with h while $Error_{OPT}$ bounded by 1

Positioning Goals – In this Talk

- Hop algorithms are not enough
- Optimal algorithm in 1 dimension
  - HS algorithm
- Improved hop-based algorithm in 2 dimensions
  - GHoST algorithm framework
- Ultimate Goal ➔ better understanding of positioning

Lessons Learned in 1D

- Define a skip in a graph $G = (V,E)$ between nodes u, w if
  - $\{u,w\} \notin E$
  - $\exists v$ such that $\{u,v\}$ and $\{v,w\} \in E$
- Define a skip path $v_0 v_1 \ldots v_k$ of length k if
  - $\{v_i,v_j\} \notin E$ for $i \neq j$
  - $\exists u_i$ such that $v_0 u_1 v_1 \ldots u_k v_k$ is a path
- Define the skip distance between $u, v \in V$ as
  - the length of the longest skip path between u and v

- Lemma: for $v \in V$ at h hops and s skips from anchor A

  $\lfloor h/2 \rfloor \leq s \leq h - 1$

Observation: for the Euclidean distance d from v to A in 1D

$s < d \leq h$

HS Algorithm – 1 Dimension

- HS algorithm:
  - Compute hop and skip distances
  - In packet from anchor A: (pos(A), hops) and (u, skips)
    - u is the last node on the skip path

- Has same asymptotic time complexity as HOP
  - At most h asynchronous time units for correct distance
  - One of those will be the one with maximal skip distance

> Theorem: In 1D, knowing h and s gives an optimal location estimate.

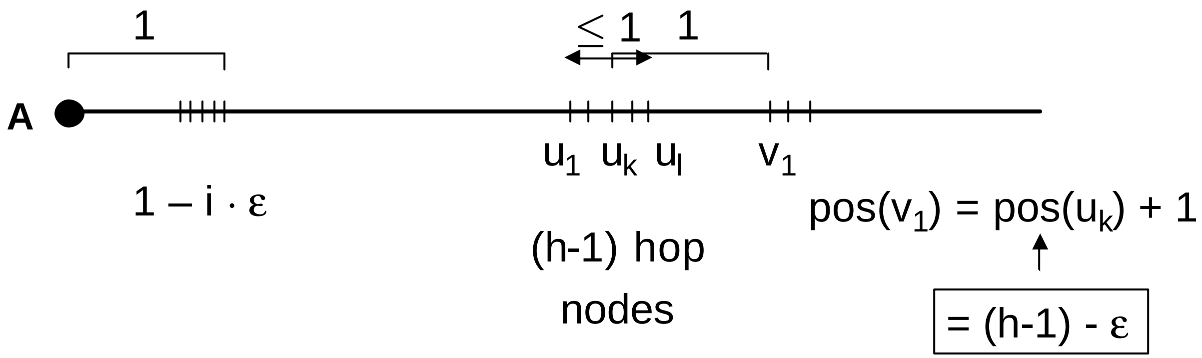
- Recall:
  - Compared to an omniscient algorithm
  - Maximum error is minimized
  - Up to an additive constant

Proof of HS Optimality in 1D

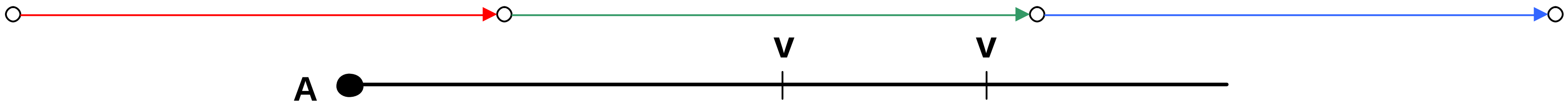
- Set up: anchor A at pos = 0, all nodes are to the right
  1. Show that it works for one anchor
  2. Show that no “hidden information” with multiple anchors

Lemma 1: If a node v is h hops from A, then there is a UDG based on G = (V,E) such that pos(v) = h - ε ($\varepsilon \to 0^+$).

Proof: Idea: Stretch graph as much as possible to the right. Use induction on h. (Nodes with same neighborhood get same position.)



Proof of HS Optimality in 1D … cont'd



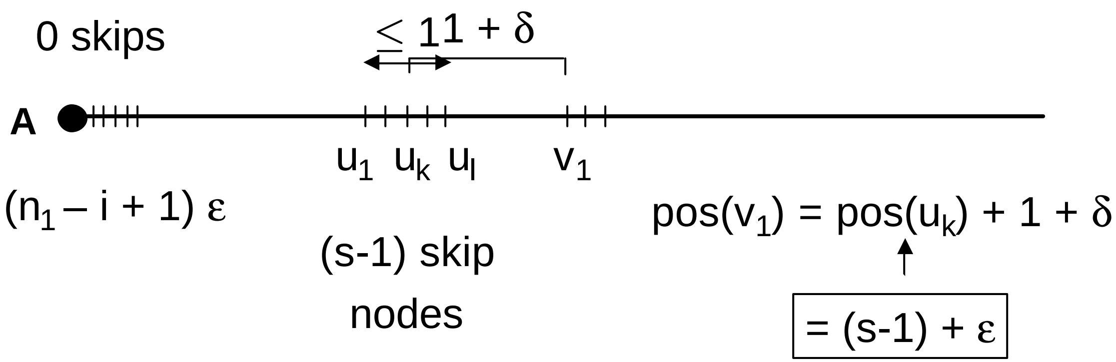
Lemma 2: If a node v is s skips from A, then there is a UDG based on $G = (V,E)$ such that $pos(v) = s + \varepsilon$ ($\varepsilon \to 0^+$).

Proof: Compress graph as much as possible to the left.

Use induction on s.

Idea: Place skip nodes as close as possible: $1 + \delta$ for $\delta \to 0$.

All s-skip nodes are neighbors: compact embedding possible.



Proof of HS Optimality in 1D … cont'd

- Lemma 3: Given a graph G = (V,E) ➔ construct U_1 = UDG(G) where pos(v) = h - ε_1 and U_2 where pos(v) = s + ε_2.
  Therefore, OPT cannot do better than HS in this case.

> Theorem: HS is optimal in 1D up to an additive constant.

Proof:

1. Interval I = [L,R] defined by borders above. Vary ε's and δ's in Lemmas 1 and 2 ➔ v anywhere in I.
2. Anchors A and B to left and right of v, respectively. Only difficulty in connection of two "chains" at v ➔ lose at most 1 unit at v's neighbors on both sides. Others are independent.
3. Multiple anchors to one side: Shortest (skip) path either goes through A_{inner}. Else, going through A_{inner} adds a hop/drops a skip ➔ at most 1 unit.

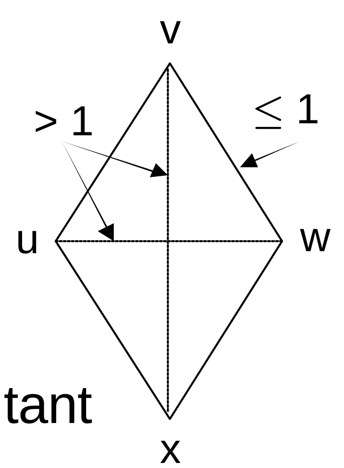
Positioning Goals – In this Talk

- Hop algorithms are not enough
- Optimal algorithm in 1 dimension
  - HS algorithm
- Improved hop-based algorithm in 2 dimensions
  - GHoST algorithm framework
- Ultimate Goal ➔ better understanding of positioning

Lessons Learned from 1D

- Do not need centralized algorithms to improve HOP!
- Local structures exist
  - Bound the length of a hop
  - Computationally cheap
  - Classify into stretchers and trimmers of hops
    - A skip (in 1D) is a stretcher: imposes minimal distance

- Trimmer T_0: $dist_E(u, w) \leq \sqrt{3} < 2$
- Trimmer T_k: paths of length k at v and x
- Trimmer MT_{k_1,k_2}: merging paths after k_1 and k_2 hops
  - $MT_{1,1}$: $dist_E(A, v) \leq \sqrt{1 + (h-1)^2} < h$ up to constant

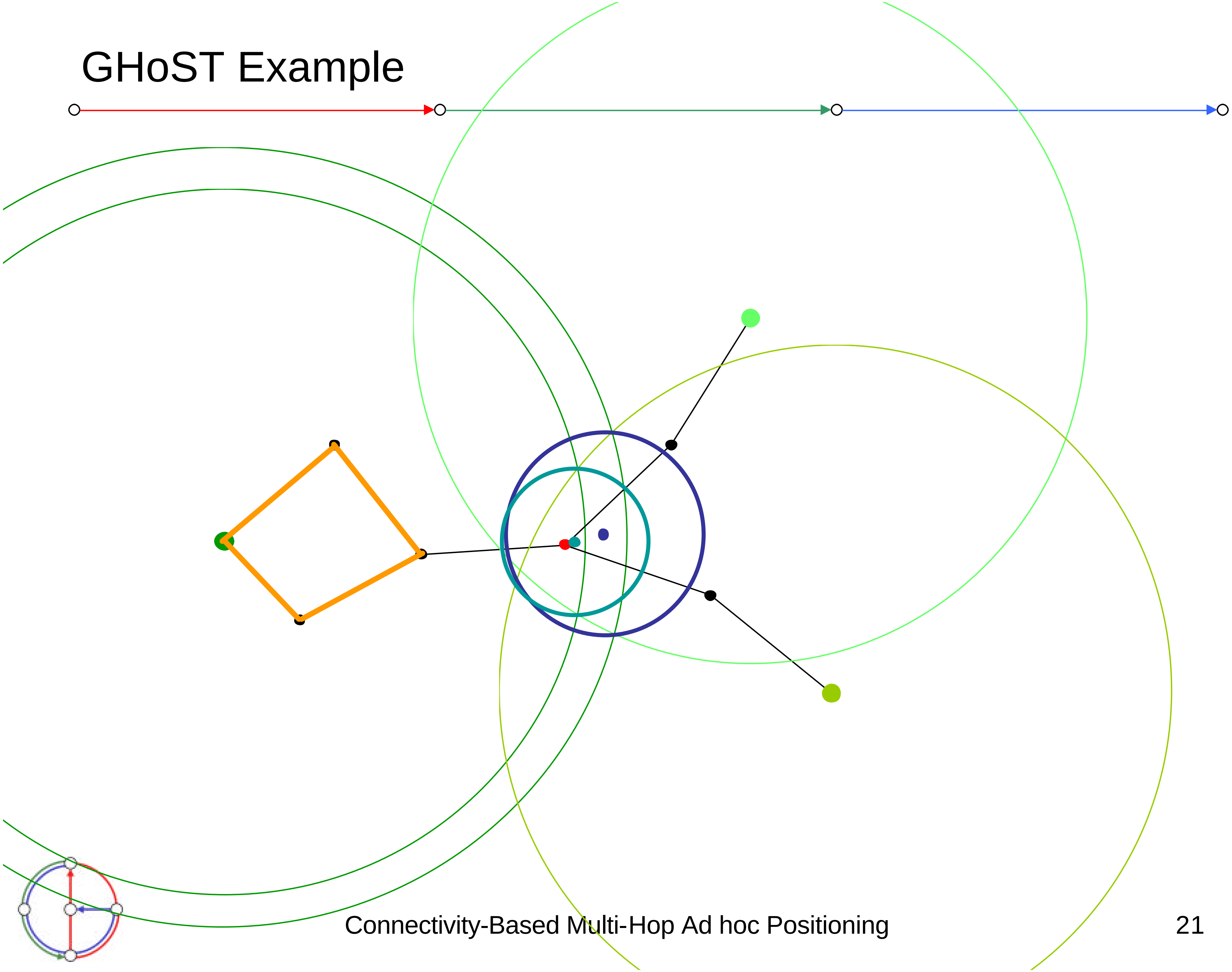


GHoST

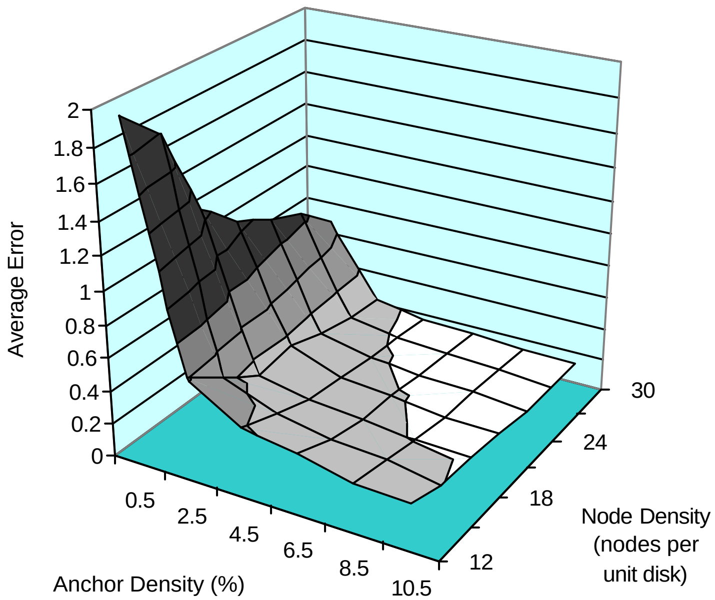
- **G**eneral **Ho**p **S**tretcher **T**rimmer Algorithm
  - Examine local neighborhood
  - Extract necessary info about local structures
  - Incorporate info to pass on upper/lower hop bounds
  - Alternatively, collect paths in messages, compute at v locally
  - Sometimes, more paths (other than shortest) are necessary
  - Possible to use heuristics or measurements
- Time complexity
  - Using shortest paths: O(h)
- Accuracy
  - Max error is smaller or equal to HOP
- Substitute into other hop-based algorithms (i.e. APS)

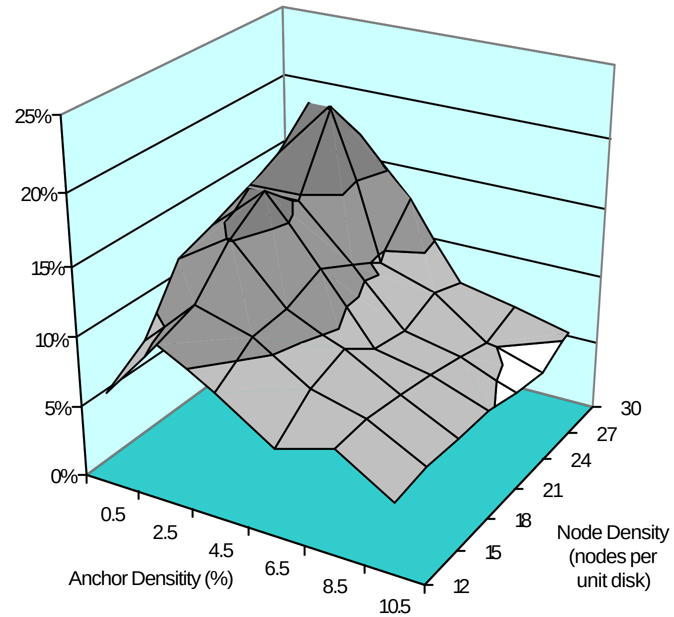
Framework

GHoST Example



GHoST in Simulation





- GHoST with T_0
  - 20 by 20 units
  - Node densities: 12 – 30 nodes per unit disk (up to 4000 nodes)
  - Anchor densities: 0.5 – 10% of the nodes
  - 300 trials per combination

Positioning Goals – In this Talk

- Hop algorithms are not enough
- Optimal algorithm in 1 dimension
  - HS algorithm
- Improved hop-based algorithm in 2 dimensions
  - GHoST algorithm framework
- Ultimate Goal ➔ better understanding of positioning
  - More trimmers & stretchers
  - Optimal 2D distributed algorithm?
  - Theoretical study of tradeoff:

    cost effectiveness of measuring instruments

Distributed
Computing
Group

Questions?
Comments?

More Work in our Group

- Ad hoc networks
  - Geometric Routing
  - Backbone Construction (Dominating Sets)
  - Mobile Routing
  - Topology Control and Interference
  - Models (Quasi-UDG)
  - Distributed Linear Programming
  - Initialization
  - Connection to peer-to-peer networks

- Peer-to-Peer networks

  … beyond information sharing
  - Systems & Theory