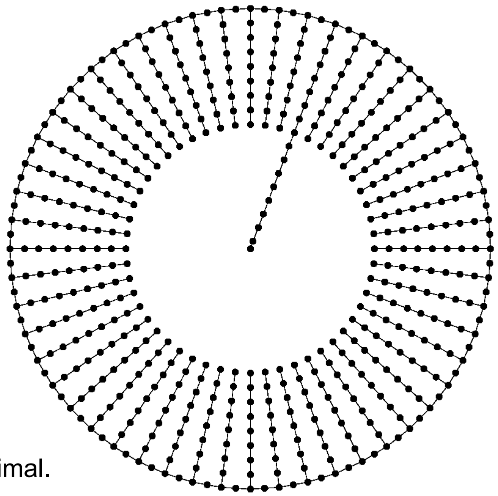
Lower Bound

- The network on the right constructs a lower bound.
- The destination is the center of the circle, the source any node on the ring.
- Finding the right chain costs $\Omega(c^{*2})$, even for randomized algorithms
- Theorem:
  AFR is asymptotically optimal.



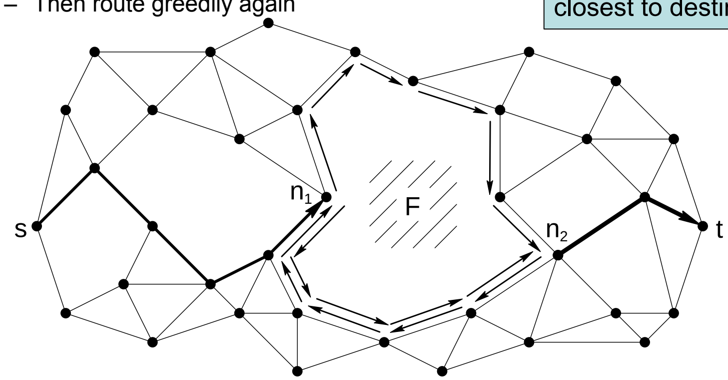
Non-geometric routing algorithms

- In the $\Omega(1)$ model, a standard flooding algorithm enhanced with trick 1 will (for the same reasons) also cost $O(c^{*2})$.
- However, such a flooding algorithm needs O(1) extra storage at each node (a node needs to know whether it has already forwarded a message).
- Therefore, there is a trade-off between O(1) storage at each node or that nodes are location aware, and also location aware about the destination. This is intriguing.

GOAFR – Greedy Other Adaptive Face Routing

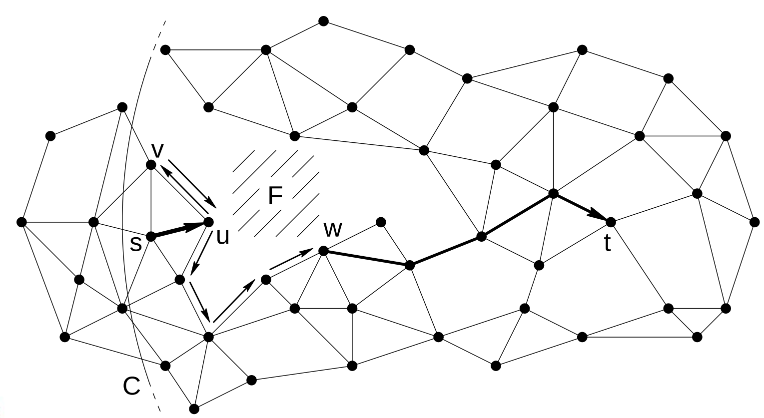
- Back to geometric routing…
- AFR Algorithm is not very efficient (especially in dense graphs)
- Combine Greedy and (Other Adaptive) Face Routing
  - Route greedily as long as possible
  - Circumvent “dead ends” by use of face routing
  - Then route greedily again

Other AFR: In each face proceed to node closest to destination



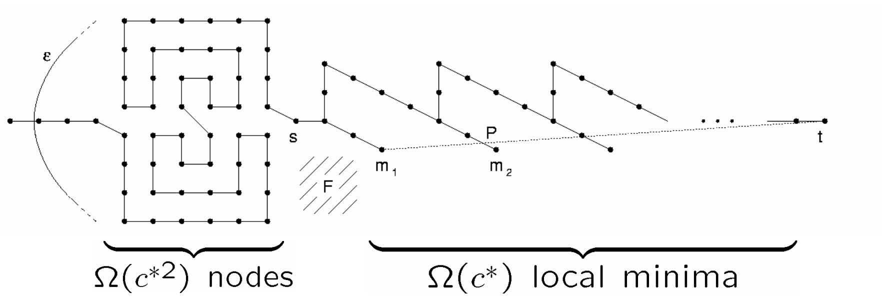
GOAFR+

- GOAFR+ improvements:
  - Early fallback to greedy routing
  - (Circle centered at destination instead of ellipse)



GOAFR+ — Early Fallback

- We could fall back to greedy routing as soon as we are closer to t than the local minimum
- But:



- "Maze" with $\Omega(c^{*2})$ edges is traversed $\Omega(c^*)$ times → $\Omega(c^{*3})$ steps

GOAFR – Greedy Other Adaptive Face Routing

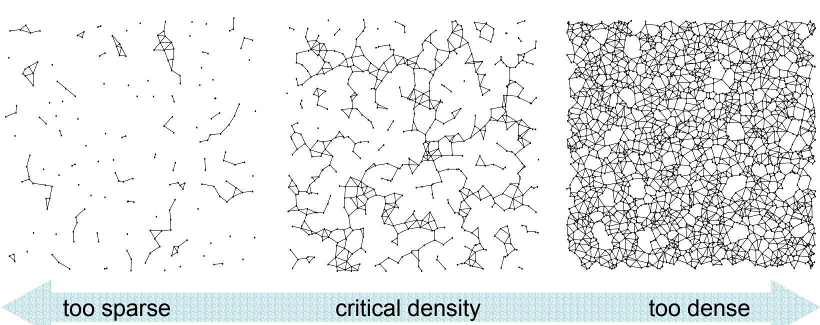
- Early fallback to greedy routing:
  - Use counters p and q. Let u be the node where the exploration of the current face F started
    - p counts the nodes closer to t than u
    - q counts the nodes *not* closer to t than u
  - Fall back to greedy routing as soon as $p > \sigma \cdot q$ (constant $\sigma > 0$)

> Theorem: GOAFR is still asymptotically worst-case optimal…
> …*and* it is efficient in practice, in the average-case.

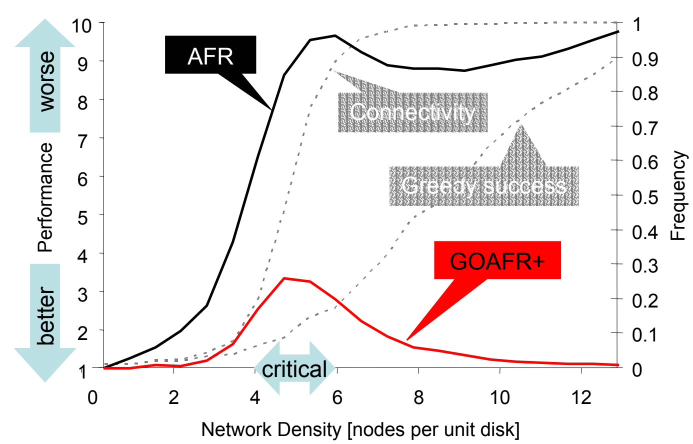
- What does "practice" mean?
  - Usually nodes placed uniformly at random

Average Case

- Not interesting when graph not dense enough
- Not interesting when graph is too dense
- Critical density range ("percolation")
  - Shortest path is significantly longer than Euclidean distance



Simulation on Randomly Generated Graphs



A Word on Performance

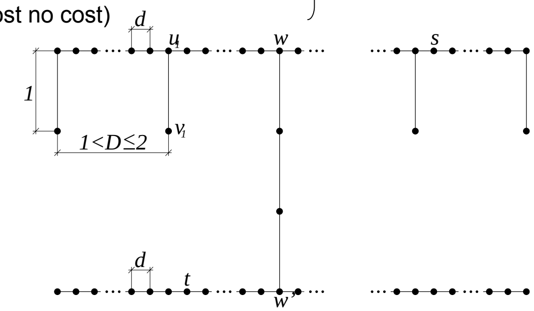
- What does a performance of 3.3 in the critical density range mean?
- If an optimal path (found by Dijkstra) has cost c, then GOAFR+ finds the destination in 3.3·c steps.
- It does *not* mean that the *path* found is 3.3 times as long as the optimal path! The path found can be much smaller…
- Remarks about cost metrics
  - In this lecture "cost" c = c hops
  - There are other results, for instance on distance/energy/hybrid metrics
  - In particular: With energy metric there is no competitive geometric routing algorithm

Energy Metric Lower Bound

Example graph: k "stalks", of which only one leads to t

- any deterministic (randomized) geometric routing algorithm A has to visit all k (at least k/2) "stalks"
- optimal path has constant cost c^* (covering a constant distance at almost no cost)

$$\lim_{k \to \infty} \frac{c(A)}{c^*} = \infty$$



→ With energy metric there is no competitive geometric routing algorithm

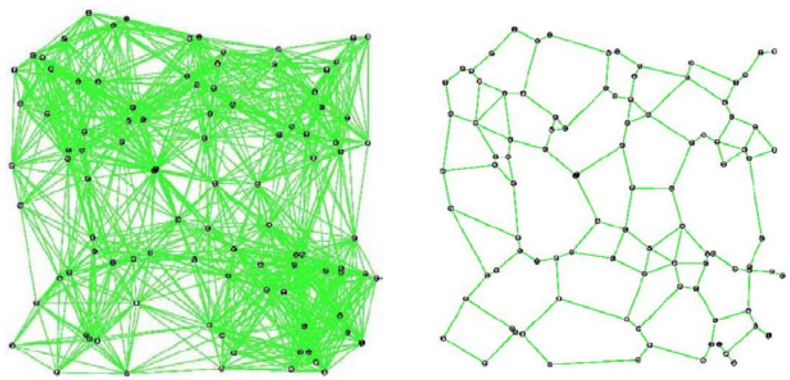
Milestones in Geometric Routing

Kleinrock et al.	Various 1975ff	MFR et al.	Geometric Routing proposed
Kranakis, Singh, Urrutia	CCCG 1999	Face Routing	First correct algorithm
Bose, Morin, Stojmenovic, Urrutia	DialM 1999	GFG	First average-case efficient algorithm (simulation but no proof)
Karp, Kung	MobiCom 2000	GPSR	A new name for GFG
Kuhn, Wattenhofer, Zollinger	DialM 2002	AFR	First worst-case analysis. Tight $\Theta(c^2)$ bound.
Kuhn, Wattenhofer, Zollinger	MobiHoc 2003	GOAFR	Worst-case optimal and average-case efficient, percolation theory
Kuhn, Wattenhofer, Zhang, Zollinger	PODC 2003	GOAFR+	Currently best algorithm, other cost metrics, etc.

Overview – Topology Control

- What is Topology Control?
- Explicit interference model
- Interference in known topologies
- Algorithms
  - Connectivity-preserving and spanner topologies
  - Worst case, average case

Topology Control

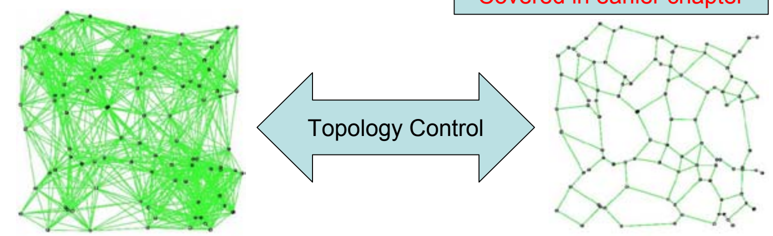


- Drop long-range neighbors: Reduces interference and energy!
- But still stay connected (or even spanner)

Topology Control as a Trade-Off

Sometimes also clustering, Dominating Set construction

Covered in earlier chapter



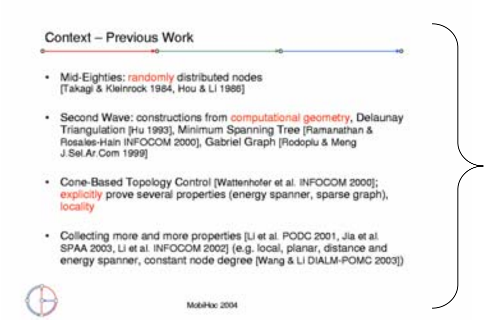
Network Connectivity
Spanner Property

$d(u,v) \cdot t \geq d_{TC}(u,v)$

Conserve Energy
Reduce Interference

Really?!?

Implicit Interference Reduction



Context – Previous Work

- Mid-Eighties: randomly distributed nodes [Takagi & Kleinrock 1984, Hou & Li 1986]
- Second Wave: constructions from computational geometry, Delaunay Triangulation [Hu 1993], Minimum Spanning Tree [Ramanathan & Rosales-Hain INFOCOM 2000], Gabriel Graph [Rodoplu & Meng J.Sel.Ar.Com 1999]
- Cone-Based Topology Control [Wattenhofer et al. INFOCOM 2000]; explicitly prove several properties (energy spanner, sparse graph), locality
- Collecting more and more properties [Li et al. PODC 2001, Jia et al. SPAA 2003, Li et al. INFOCOM 2002] (e.g. local, planar, distance and energy spanner, constant node degree [Wang & Li DIALM-POMC 2003])

MobiHoc 2004

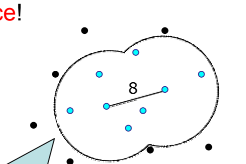
Interference issue "solved" implicitly by graph sparseness or bounded degree

Explicit interference [Meyer auf der Heide et al. SPAA 2002]

- Interference between edges, time-step routing model, congestion
- Trade-offs: congestion, power consumption, dilation
- Interference model based on network traffic

What Is Interference?

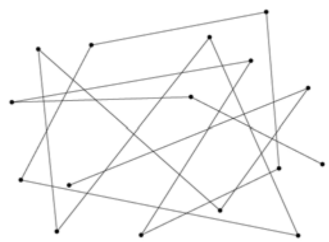
- Model
  - Transmitting edge *e* = (*u*,*v*) disturbs all nodes in vicinity
  - Interference of edge *e* =
    # Nodes covered by union of the two circles with center u and v, respectively, and radius |*e*|
- Problem statement
  - We want to minimize maximum interference!
  - At the same time topology must be connected or a spanner etc.



Exact size of interference range does not change the results

Low Node Degree Topology Control?

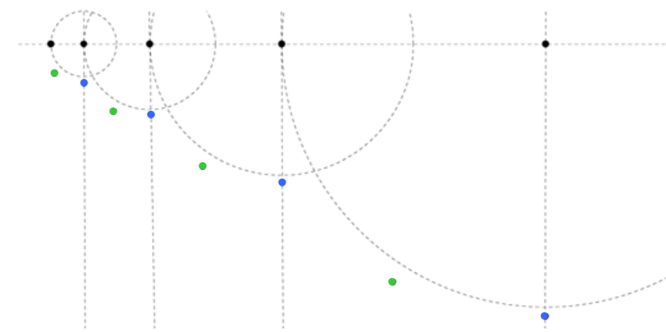
Low node degree does not necessarily imply low interference:



Very low node degree
but huge interference

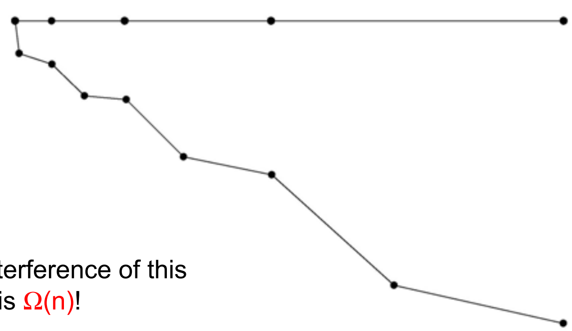
Let's Study the Following Topology!

…from a worst-case perspective



Topology Control Algorithms Produce…

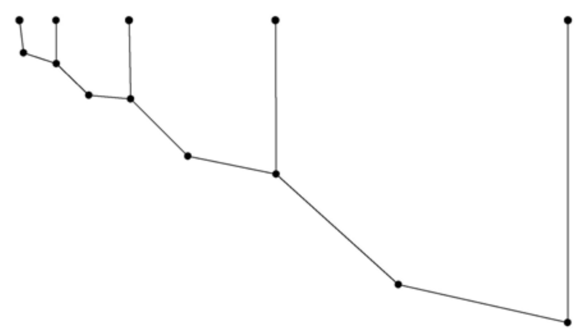
- All known topology control algorithms (with symmetric edges) include the nearest neighbor forest as a subgraph and produce something like this:



- The interference of this graph is $\Omega(n)$!

But Interference…

- Interference does not need to be high…

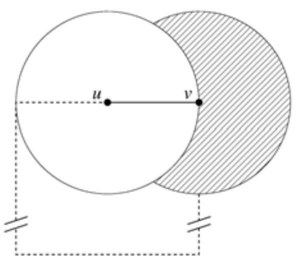


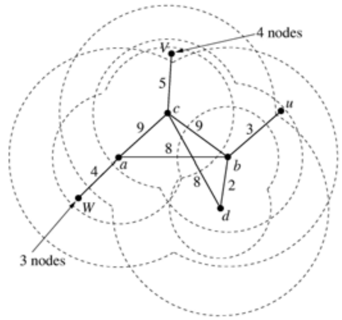
- This topology has interference $O(1)$!!

Interference-Optimal Topology

There is no local algorithm that can find a good interference topology

The optimal topology will not be planar





Algorithms – Requirement: Retain Graph Connectivity

- LIFE (Low Interference Forest Establisher)
- Attribute interference values as weights to edges
- Compute minimum spanning tree/forest (Kruskal's algorithm)

Low Interference Forest Establisher (LIFE)

Input: a set of nodes V, each $v \in V$ having attributed a maximum transmission radius r_v^{max}

```
E = all eligible edges (u,v) (r_u^max ≥ |u,v| and r_v^max ≥ |u,v|) (* unprocessed edges *)
E_LIFE = ∅
G_LIFE = (V, E_LIFE)
while E ≠ ∅ do
    e = (u,v) ∈ E with minimum coverage
    if u, v are not connected in G_LIFE then
        E_LIFE = E_LIFE ∪ {e}
    end if
    E = E \ {e}
end while
```

Output: Graph G_{LIFE}

Theorem: LIFE constructs a Minimum Interference Forest

Proof:
- Algorithm computes forest
- MST also minimizes maximum interference value

Algorithms – Requirement: Construct Spanner

- LISE (Low Interference Spanner Establisher)
- Add edges with increasing interference until spanner property fulfilled

Low Interference Spanner Establisher (LISE)

Input: a set of nodes V, each $v \in V$ having attributed a maximum transmission radius r_v^{max}

```
E = all eligible edges (u,v) (r_u^max ≥ |u,v| and r_v^max ≥ |u,v|) (* unprocessed edges *)
E_LISE = ∅
G_LISE = (V, E_LISE)
while E ≠ ∅ do
    e = (u,v) ∈ E with maximum coverage
    while |p*(u,v) in G_LISE| > t|u,v| do
        f = edge ∈ E with minimum coverage
        move all edges ∈ E with coverage Cov(f) to E_LISE
    end while
    E = E \ {e}
end while
```

Output: Graph G_{LISE}

Theorem: LISE constructs a Minimum Interference t-Spanner

Proof:
- Algorithm computes t-spanner
- Algorithm inserts edges with increasing coverage only "as long as necessary"

Algorithms – Requirement: Construct Spanner Locally

- LLISE
- Local algorithm: scalable
- Nodes collect (t/2)-neighborhood
- Locally compute interference-minimal paths guaranteeing spanner property
- Only request that path to stay in the resulting topology

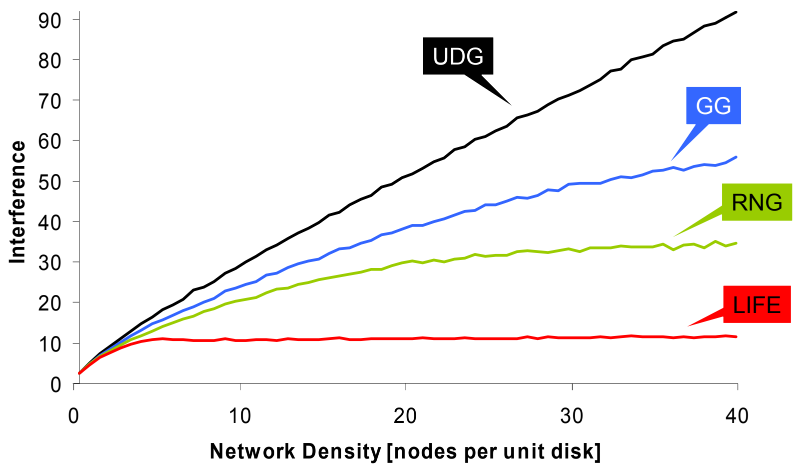
LLISE

```
collect (t/2)-neighborhood G_N = (V_N, E_N) of G = (V, E)
E' = ∅
G' = (V_N, E')
repeat
    f = edge ∈ E_N with minimum coverage
    move all edges ∈ E_N with coverage Cov(f) to E'
    p = shortestPath(u − v) in G'
until |p| ≤ t|u,v|
inform all edges on p to remain in the resulting topology.
```

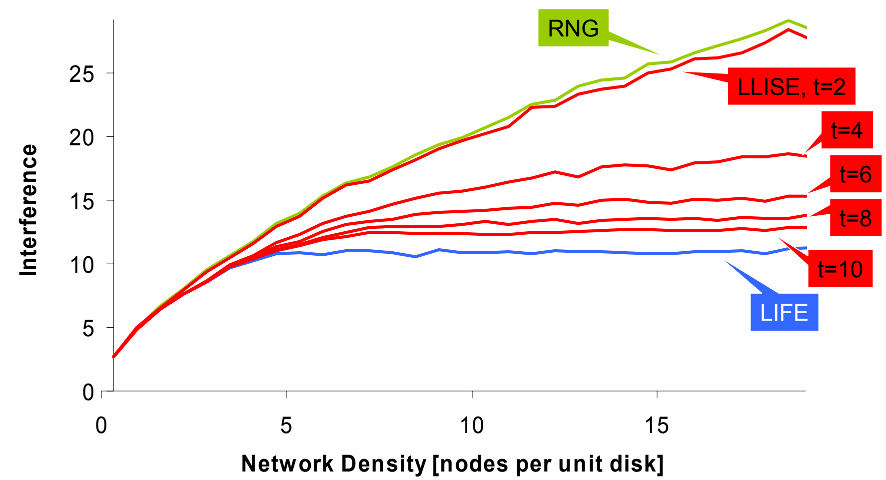
Note: $G_{LL} = (V, E_{LL})$ consists of all edges eventually informed to remain in the resulting topology.

Theorem: LLISE constructs a Minimum Interference t-Spanner

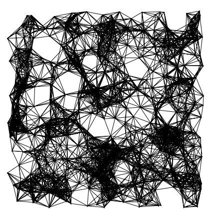
Average-Case Interference: Preserve Connectivity



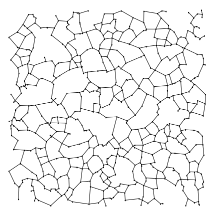
Average-Case Interference: Spanners



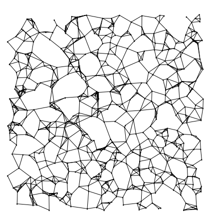
Simulation



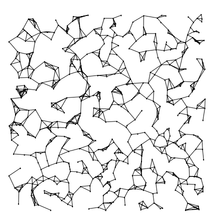
UDG, I = 50



RNG, I = 25



$LLISE_2$, I = 23

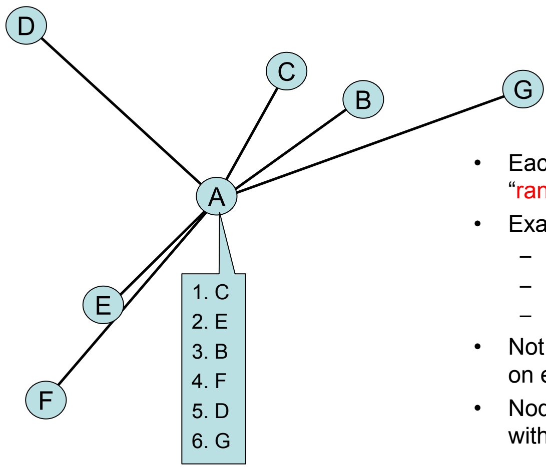


$LLISE_{10}$, I = 12

Overview – Lightweight Topology Control

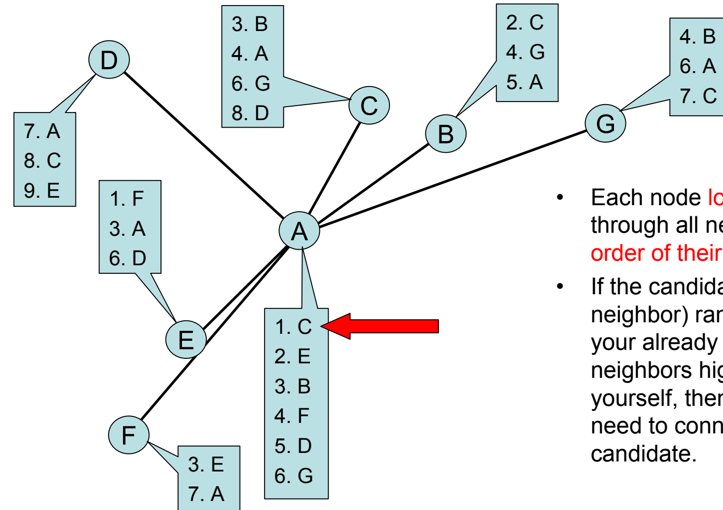
- Topology Control commonly assumes that the node positions are known.
- What if we do not have access to position information?
- XTC algorithm
- XTC analysis
  - Worst case
  - Average case

XTC Algorithm



- Each node produces "ranking" of neighbors.
- Examples
  - Distance (closest)
  - Energy (lowest)
  - Link quality (best)
- Not necessarily depending on explicit positions
- Nodes exchange rankings with neighbors

XTC Algorithm (Part 2)



- Each node locally goes through all neighbors in order of their ranking
- If the candidate (current neighbor) ranks any of your already processed neighbors higher than yourself, then you do not need to connect to the candidate.

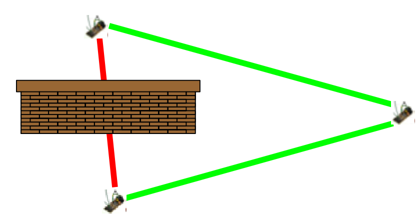
XTC Analysis (Part 1)

- Symmetry: A node u wants a node v as a neighbor if and only if v wants u.
- Proof:
  - Assume 1) $u \rightarrow v$ and 2) $u \not\leftarrow v$
  - Assumption 2) $\Rightarrow \exists w$: (i) $w \prec_v u$ and (ii) $w \prec_u v$

Contradicts Assumption 1)

XTC Analysis (Part 1)

- Symmetry: A node u wants a node v as a neighbor if and only if v wants u.
- Connectivity: If two nodes are connected originally, they will stay so (provided that rankings are based on symmetric link-weights).
- If the ranking is energy or link quality based, then XTC will choose a topology that routes around walls and obstacles.

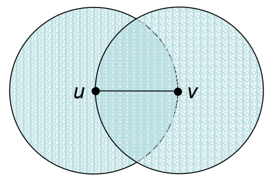


XTC Analysis (Part 2)

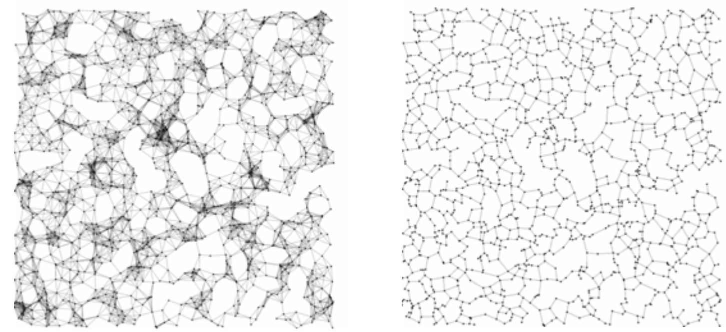
- If the given graph is a Unit Disk Graph (no obstacles, nodes homogeneous, but not necessarily uniformly distributed), then …

- The degree of each node is at most 6.
- The topology is planar.
- The graph is a subgraph of the RNG.

- Relative Neighborhood Graph RNG(V):
- An edge e = (u,v) is in the RNG(V) iff there is no node w with (u,w) < (u,v) and (v,w) < (u,v).

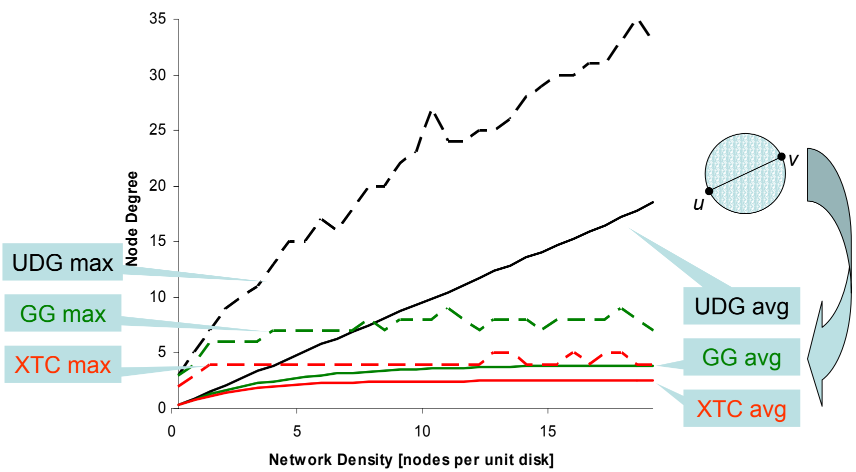


XTC Average-Case

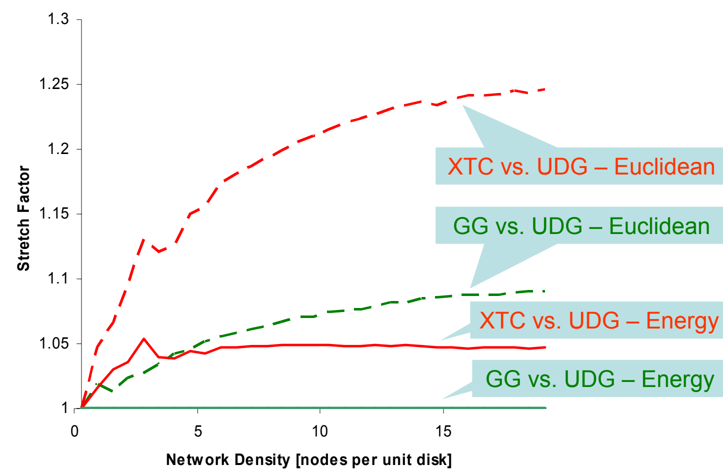


Unit Disk Graph XTC

XTC Average-Case (Degrees)



XTC Average-Case (Stretch Factor)



XTC Average-Case (Geometric Routing)