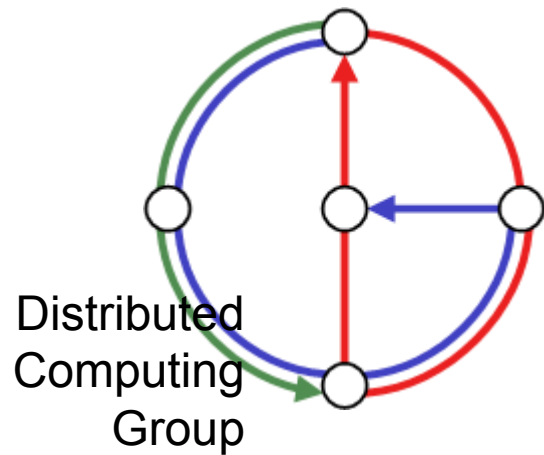


# Chapter 7

# GEOMETRIC

# ROUTING

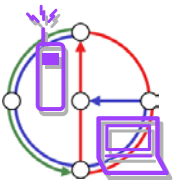


Mobile Computing  
Summer 2003

# Overview



- Geometric routing
- Greedy geometric routing
  
- Euclidean and planar graphs
- Unit disk graph
- Gabriel graph and other planar graphs
  
- Face Routing
- Adaptive Face Routing
- Lower bound
- Greedy (Other) Adaptive Face Routing

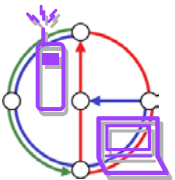
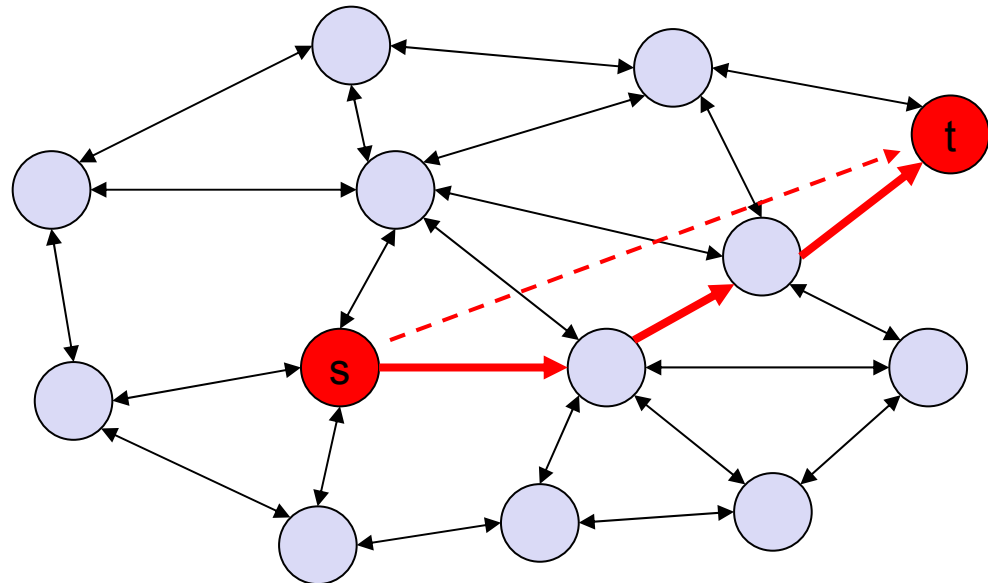


# Geometric (Directional, Position-based) routing



- ...even with all the tricks there will be flooding every now and then.
- In this chapter we will assume that the nodes are location aware (they have GPS, Galileo, or an ad-hoc way to figure out their coordinates), and that we know where the destination is.

- Then we simply route towards the destination

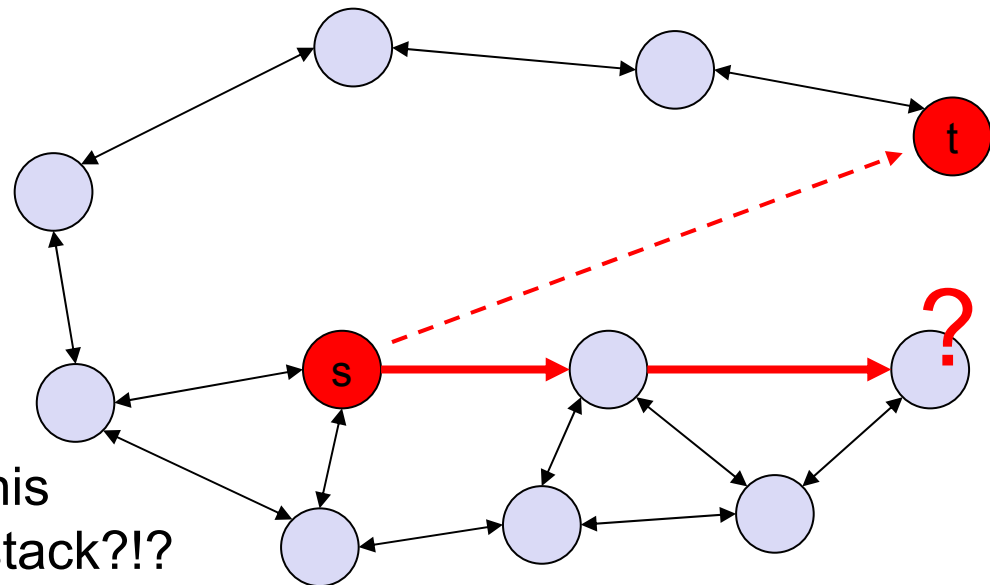


# Geometric routing

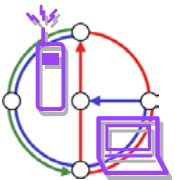


- Problem: What if there is no path in the right direction?
- We need a guaranteed way to reach a destination even in the case when there is no directional path...

- Hack: as in flooding nodes keep track of the messages they have already seen, and then they backtrack\* from there



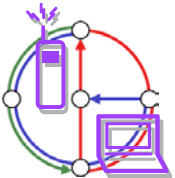
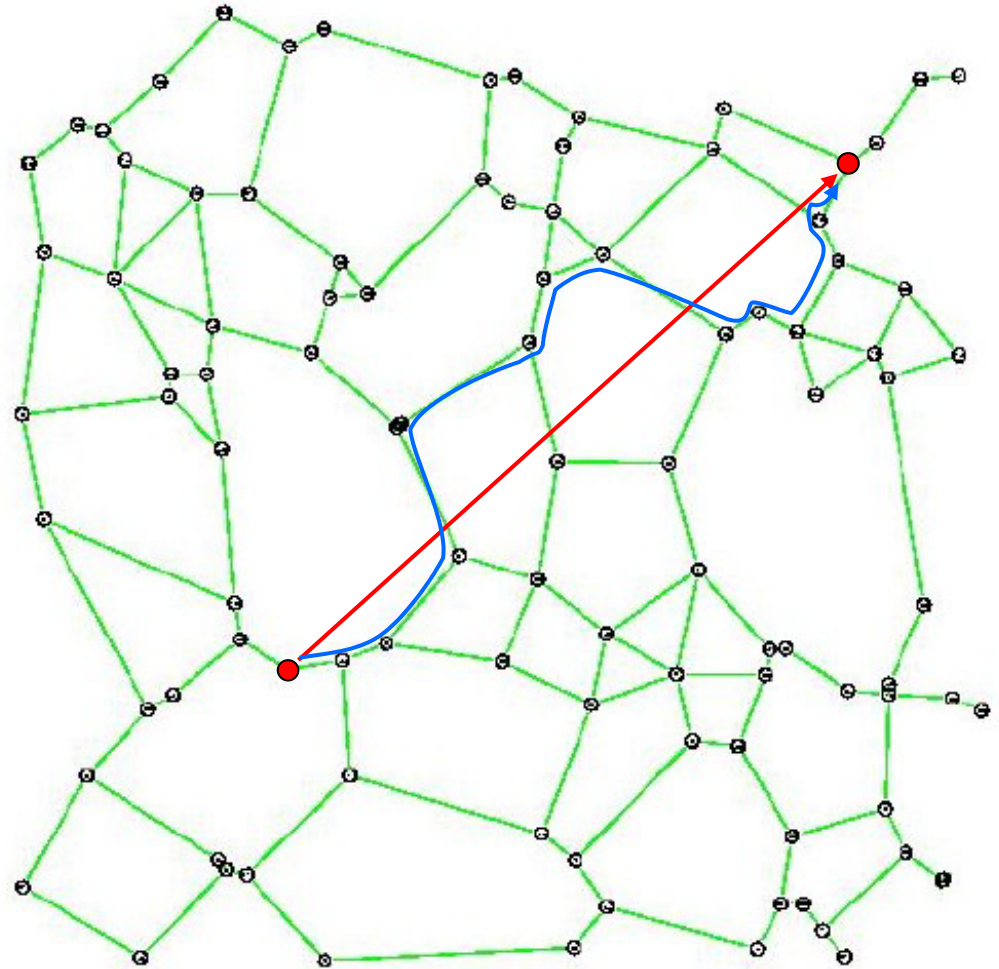
\*backtracking? Does this mean that we need a stack?!?



# Greedy routing



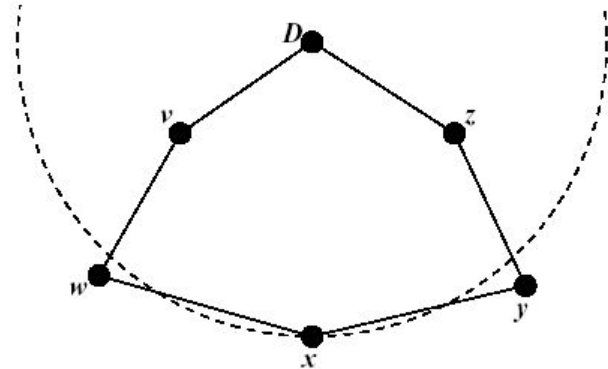
- Greedy routing looks promising.
- Maybe there is a way to choose the next neighbor and a particular graph where we always reach the destination?



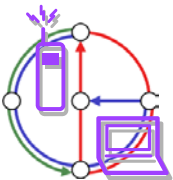
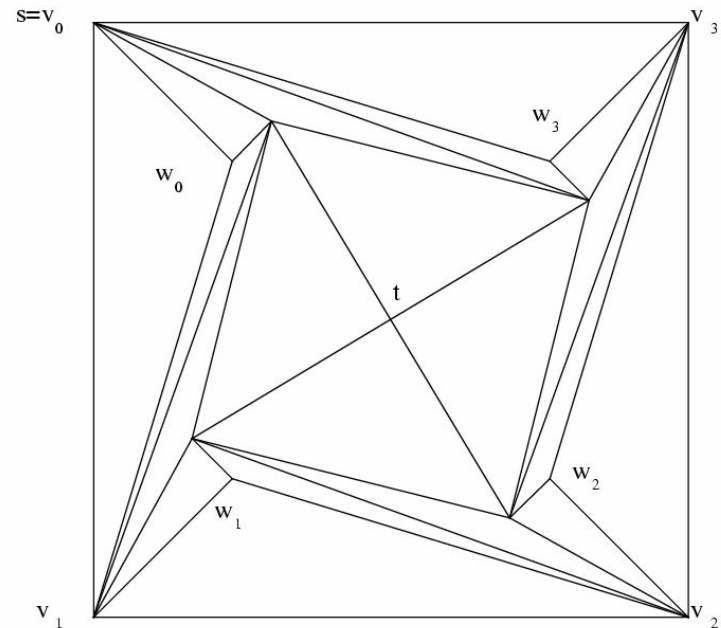
# Examples why greedy algorithms fail



- We greedily route to the neighbor which is closest to the destination:  
But both neighbors of  $x$  are not closer to destination  $D$



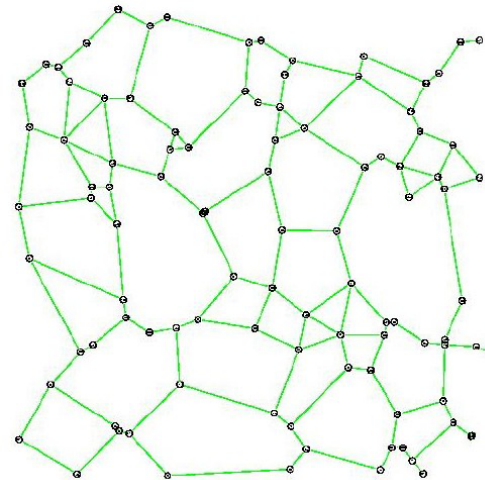
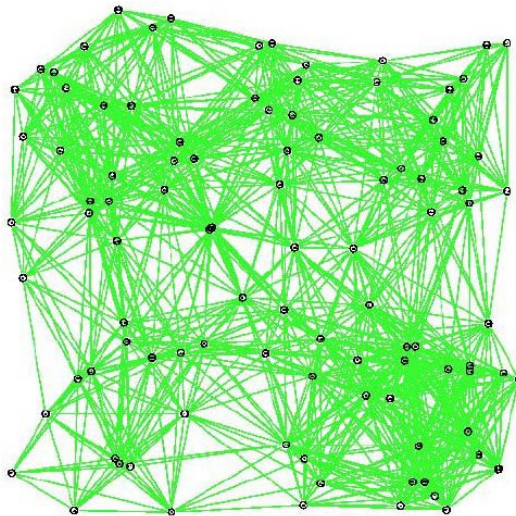
- Also the best angle approach might fail, even in a triangulation:  
if, in the example on the right, you always follow the edge with the narrowest angle to destination  $t$ , you will forward on a loop  $v_0, w_0, v_1, w_1, \dots, v_3, w_3, v_0, \dots$



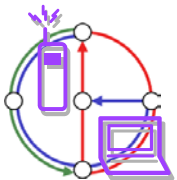
# Euclidean and Planar Graphs



- Euclidean: Points in the plane, with coordinates
- Planar: can be drawn without “edge crossings” in a plane



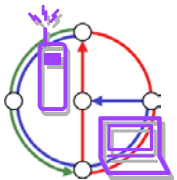
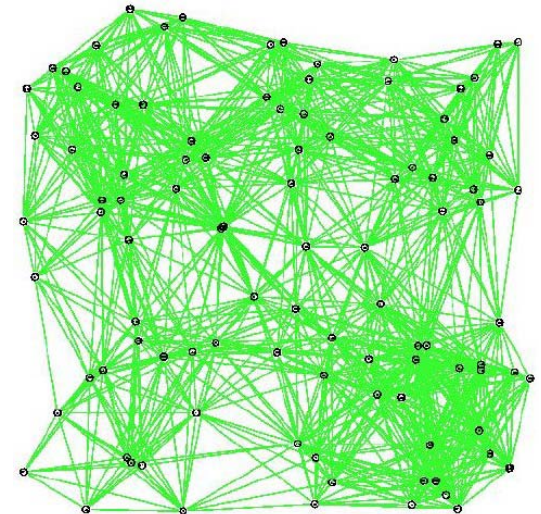
- Euclidean planar graphs (planar embedding) simplify geometric routing.



# Unit disk graph



- We are given a set  $V$  of nodes in the plane (points with coordinates).
- The unit disk graph  $UDG(V)$  is defined as an undirected graph (with  $E$  being a set of undirected edges). There is an edge between two nodes  $u, v$  iff the Euclidean distance between  $u$  and  $v$  is at most 1.
- Think of the unit distance as the maximum transmission range.
- We assume that the unit disk graph  $UDG$  is connected (that is, there is a path between each pair of nodes)
- The unit disk graph has many edges.
- Can we drop some edges in the  $UDG$  to reduced complexity and interference?

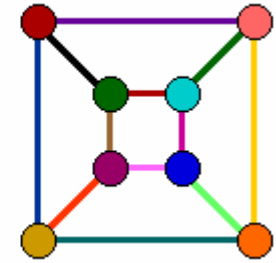
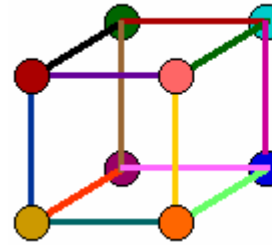




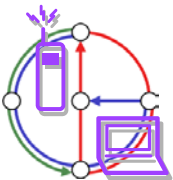
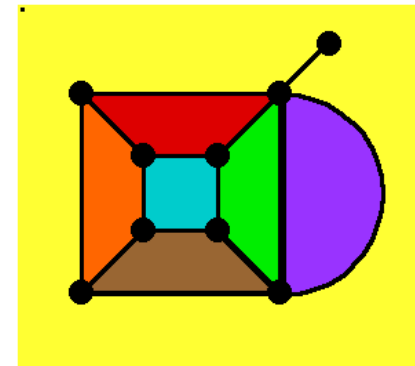
# Planar graphs



- Definition: A planar graph is a graph that can be drawn in the plane such that its edges only intersect at their common end-vertices.



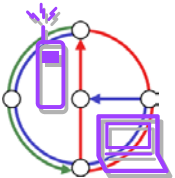
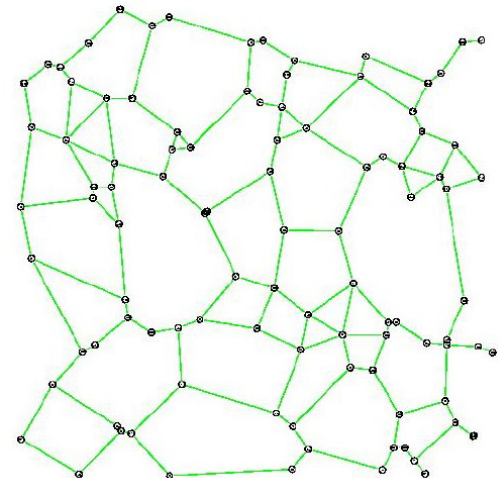
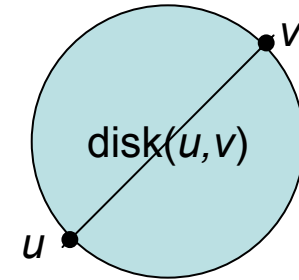
- Kuratowski's Theorem: A graph is planar iff it contains no subgraph that is edge contractible to  $K_5$  or  $K_{3,3}$ .
- Euler's Polyhedron Formula: A connected planar graph with  $n$  nodes,  $m$  edges, and  $f$  faces has  $n - m + f = 2$ .
- Right: Example with 9 vertices, 14 edges, and 7 faces (the yellow "outside" face is called the infinite face)
- Theorem: A simple planar graph with  $n$  nodes has at most  $3n - 6$  edges, for  $n \geq 3$ .



# Gabriel Graph



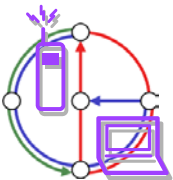
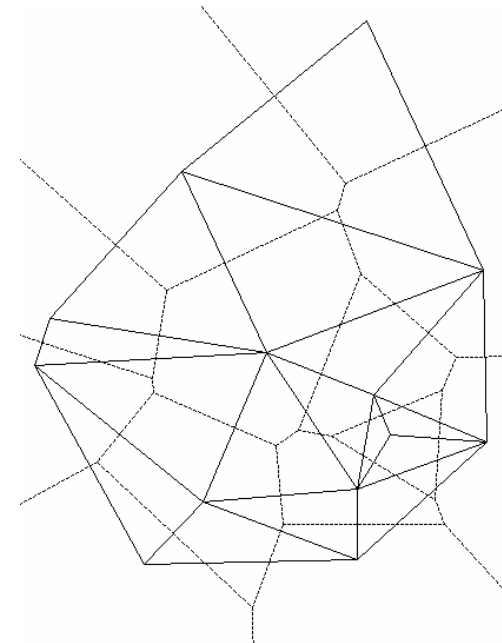
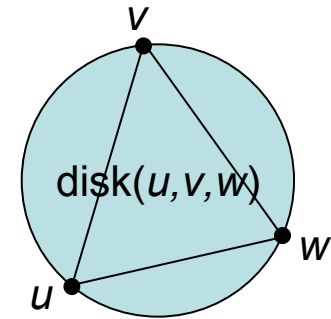
- Let  $\text{disk}(u,v)$  be a disk with diameter  $(u,v)$  that is determined by the two points  $u,v$ .
- The Gabriel Graph  $\text{GG}(V)$  is defined as an undirected graph (with  $E$  being a set of undirected edges). There is an edge between two nodes  $u,v$  iff the disk  $(u,v)$  including boundary contains no other points.
- As we will see the Gabriel Graph has interesting properties.



# Delaunay Triangulation



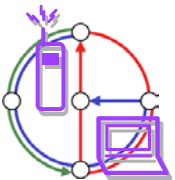
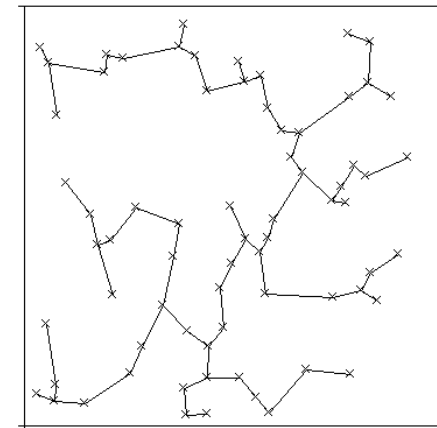
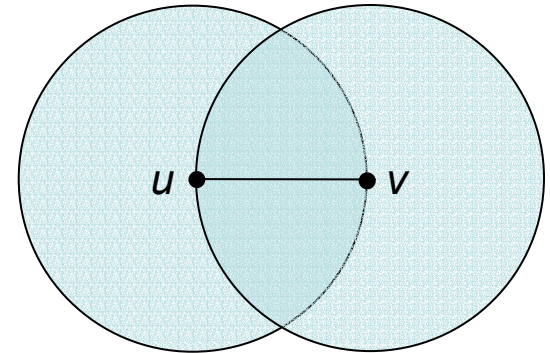
- Let  $\text{disk}(u,v,w)$  be a disk defined by the three points  $u,v,w$ .
- The Delaunay Triangulation (Graph)  $\text{DT}(V)$  is defined as an undirected graph (with  $E$  being a set of undirected edges). There is a triangle of edges between three nodes  $u,v,w$  iff the  $\text{disk}(u,v,w)$  contains no other points.
- The Delaunay Triangulation is the dual of the Voronoi diagram, and widely used in various CS areas; the DT is planar; the distance of a path  $(s,\dots,t)$  on the DT is within a constant factor of the  $s$ - $t$  distance.



# Other planar graphs



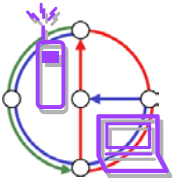
- Relative Neighborhood Graph  $RNG(V)$
- An edge  $e = (u,v)$  is in the  $RNG(V)$  iff there is no node  $w$  with  $(u,w) < (u,v)$  and  $(v,w) < (u,v)$ .
- Minimum Spanning Tree  $MST(V)$
- A subset of  $E$  of  $G$  of minimum weight which forms a tree on  $V$ .



# Properties of planar graphs



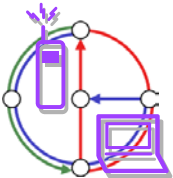
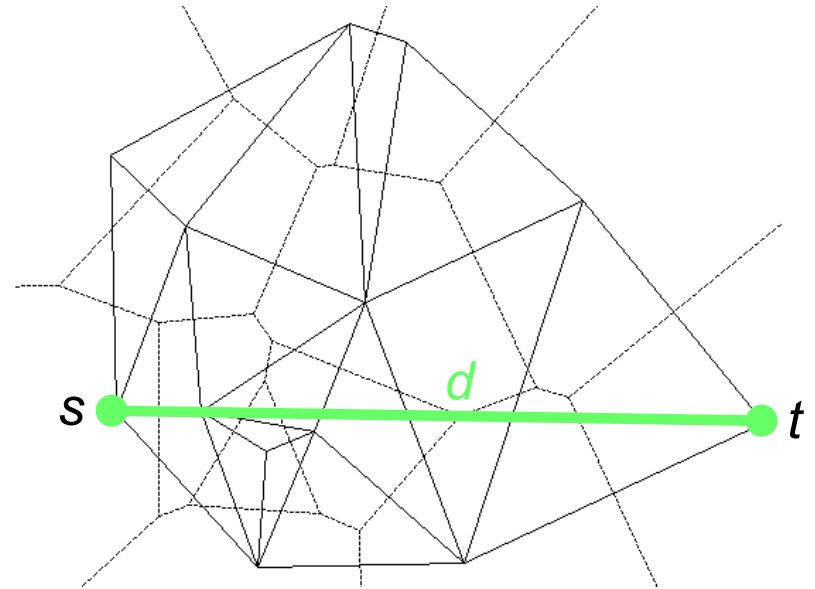
- Theorem 1:  
 $MST(V) \subseteq RNG(V) \subseteq GG(V) \subseteq DT(V)$
- Corollary:  
Since the  $MST(V)$  is connected and the  $DT(V)$  is planar, all the planar graphs in Theorem 1 are connected and planar.
- Theorem 2:  
The Gabriel Graph contains the Minimum Energy Path  
(for any path loss exponent  $\alpha \geq 2$ )
- Corollary:  
 $GG(V) \cap UDG(V)$  contains the Minimum Energy Path in  $UDG(V)$



# Routing on Delaunay Triangulation?



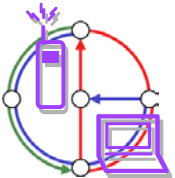
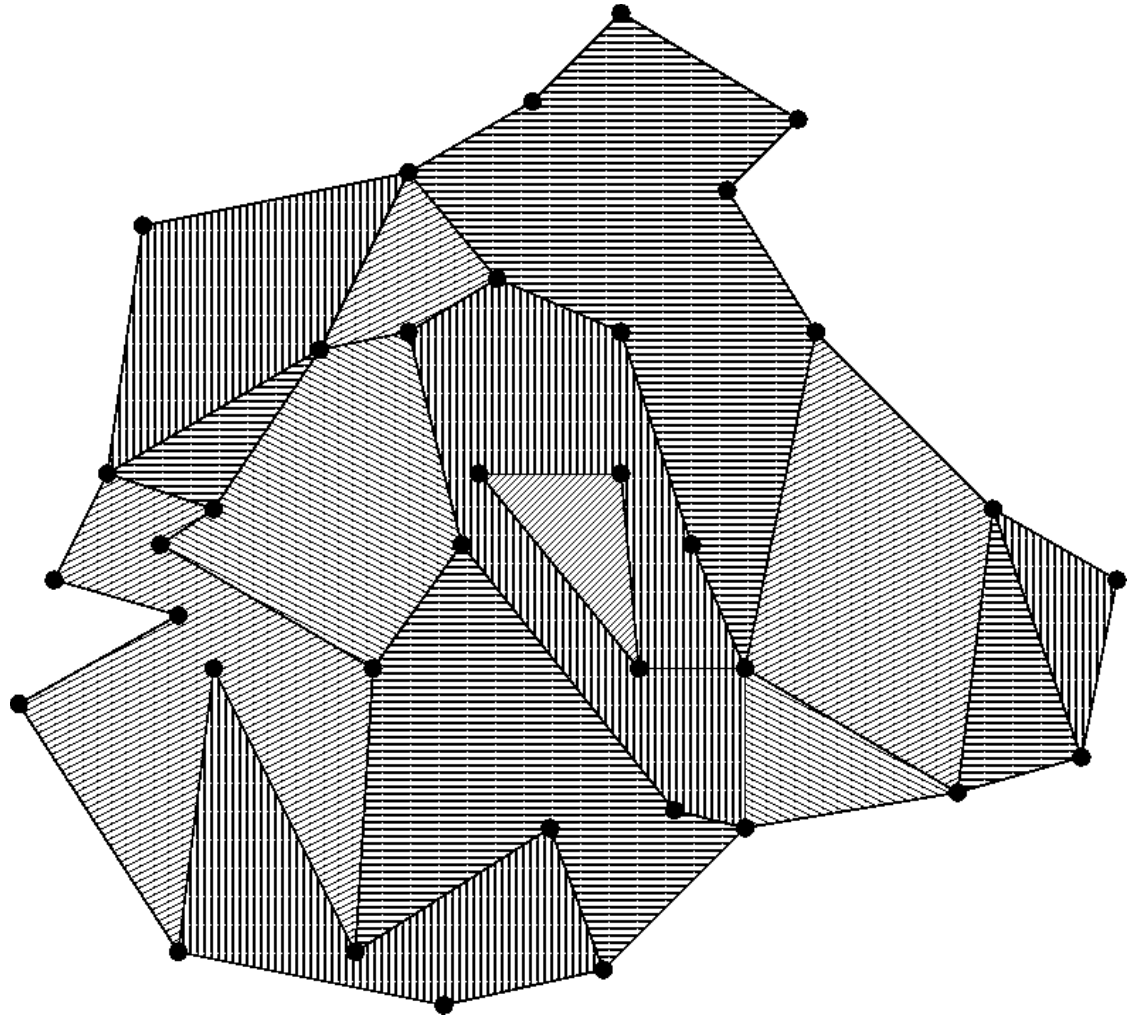
- Let  $d$  be the Euclidean distance of source  $s$  and destination  $t$
- Let  $c$  be the sum of the distances of the links of the shortest path in the Delaunay Triangulation
- It was shown that  $c = \Theta(d)$
- Two problems:
  - 1) How do we find this best route in the DT? With flooding?!?
  - 2) How do we find the DT at all in a distributed fashion?
- ... and even worse: The DT contains edges that are not in the UDG, that is, nodes that cannot hear each other are “neighbors” on DT



# Breakthrough idea: route on faces



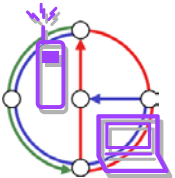
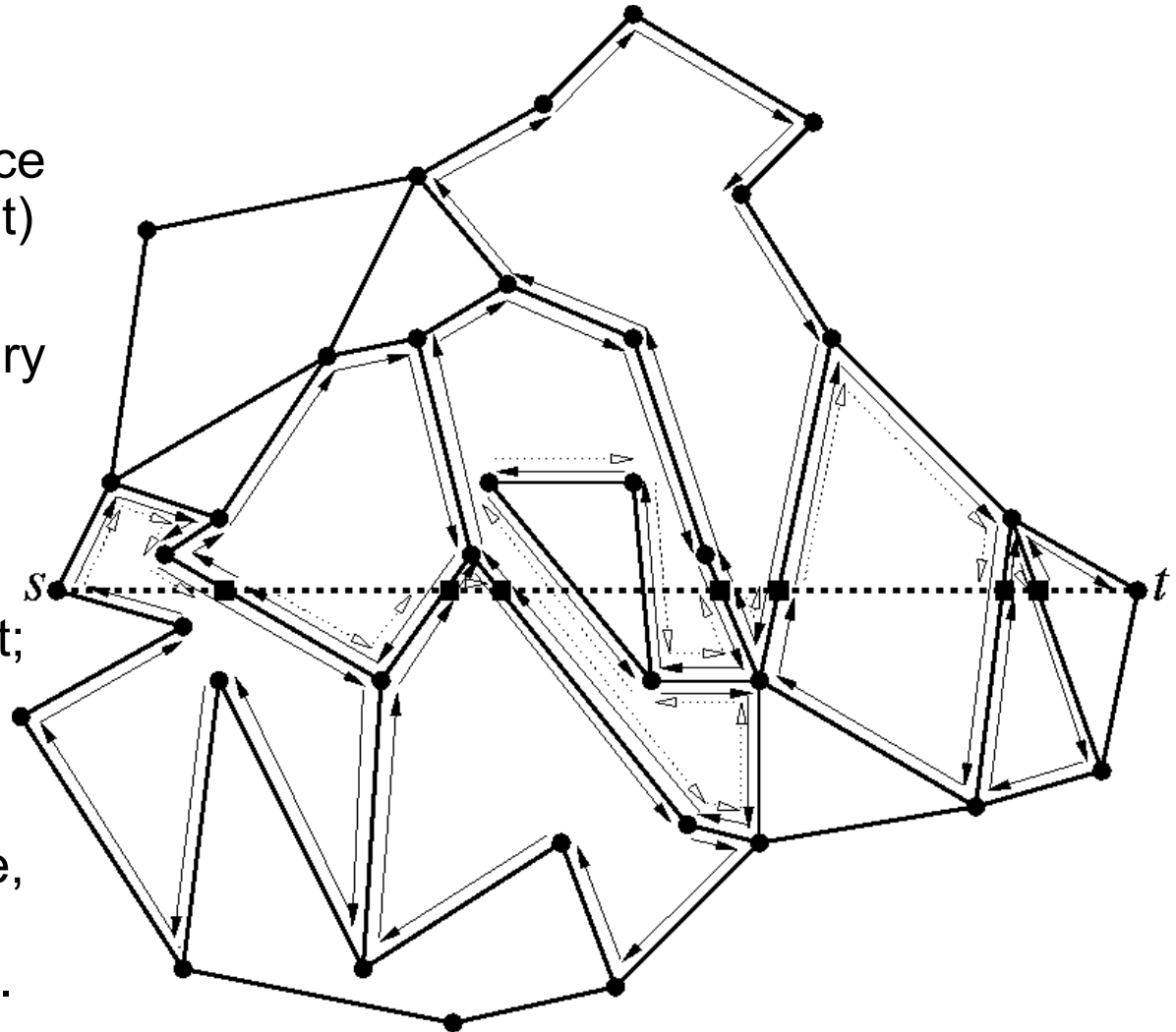
- Remember the faces...
- Idea:  
Route along the boundaries of the faces that lie on the source–destination line



# Face Routing

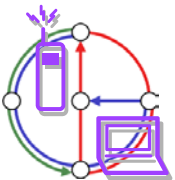
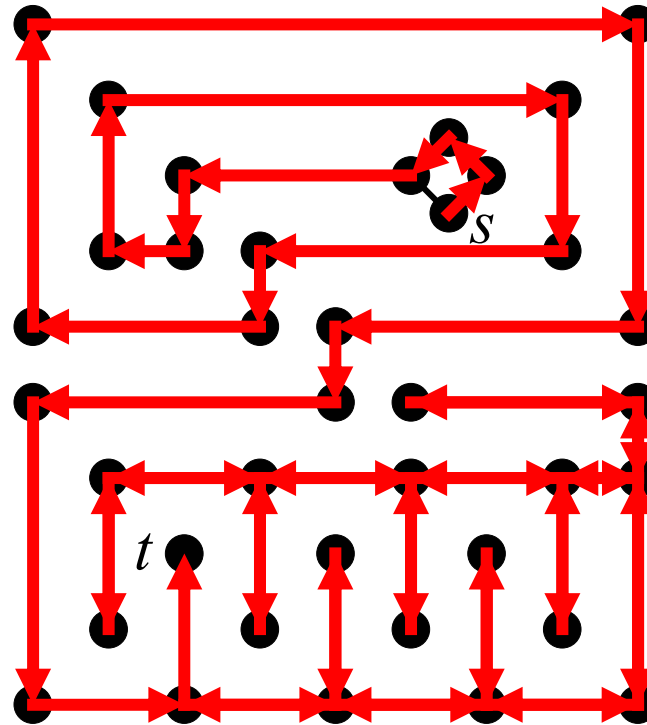


0. Let  $f$  be the face incident to the source  $s$ , intersected by  $(s,t)$
1. Explore the boundary of  $f$ ; remember the point  $p$  where the boundary intersects with  $(s,t)$  which is nearest to  $t$ ; after traversing the whole boundary, go back to  $p$ , switch the face, and repeat 1 until you hit destination  $t$ .





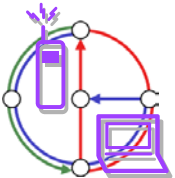
# Face Routing Works on *Any* Graph



# Face routing is correct



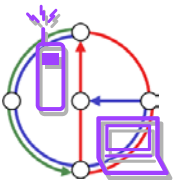
- Theorem: Face routing terminates on any simple planar graph in  $O(n)$  steps, where  $n$  is the number of nodes in the network
- Proof: A simple planar graph has at most  $3n-6$  edges. You leave each face at the point that is closest to the destination, that is, you never visit a face twice, because you can order the faces that intersect the source—destination line on the exit point. Each edge is in at most 2 faces. Therefore each edge is visited at most 4 times. The algorithm terminates in  $O(n)$  steps.



# Is there something better than Face Routing?



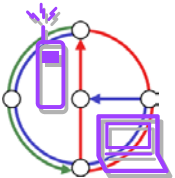
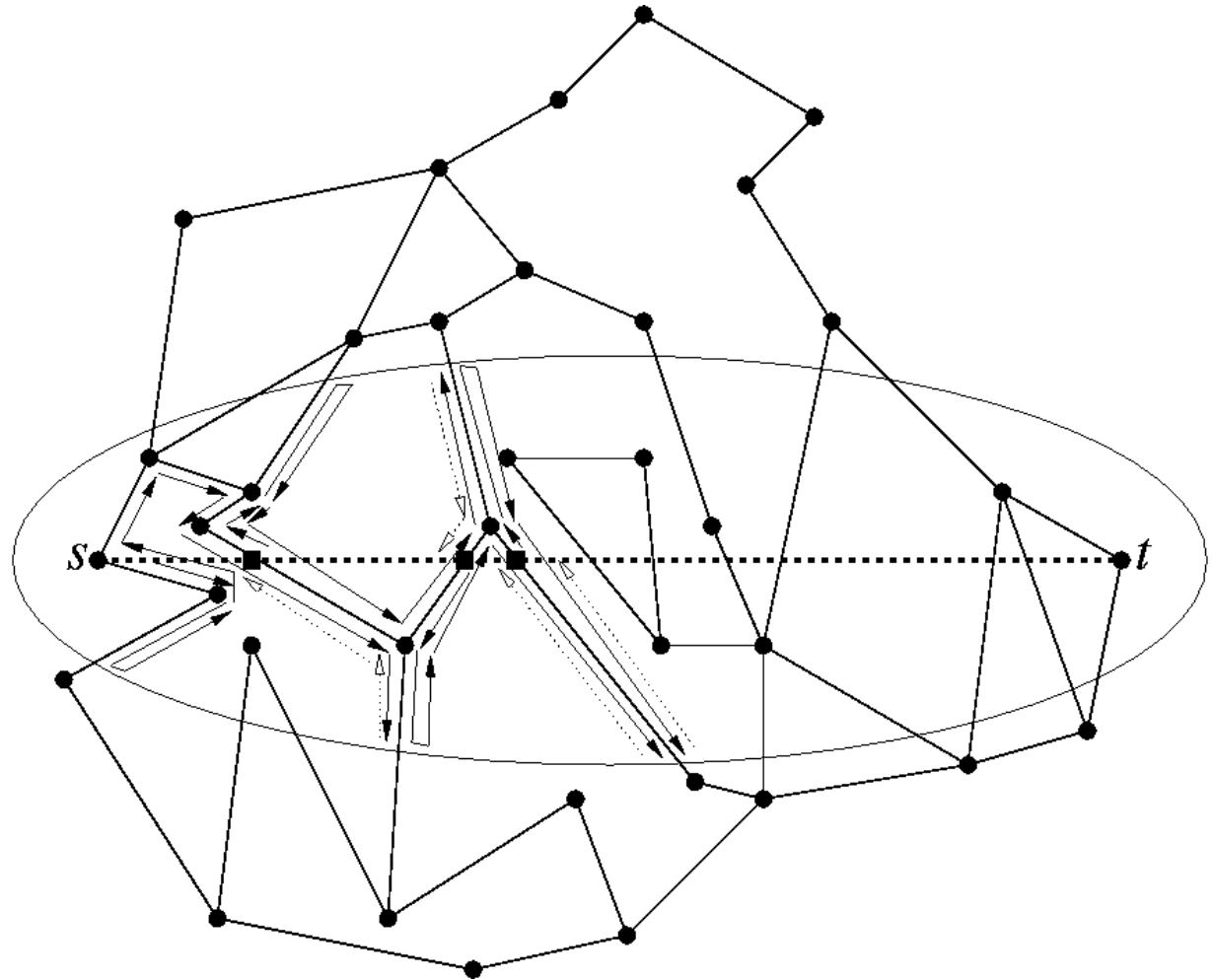
- How to improve face routing? Face Routing 2 😊
- Idea: Don't search a whole face for the best exit point, but take the first (better) exit point you find. Then you don't have to traverse huge faces that point away from the destination.
- Efficiency: Seems to be practically more efficient than face routing. But the theoretical worst case is worse –  $O(n^2)$ .
- Problem: if source and destination are very close, we don't want to route through all nodes of the network. Instead we want a routing algorithm where the cost is a function of the cost of the best route in the unit disk graph (and independent of the number of nodes).



# Adaptive Face Routing (AFR)



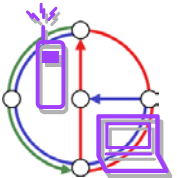
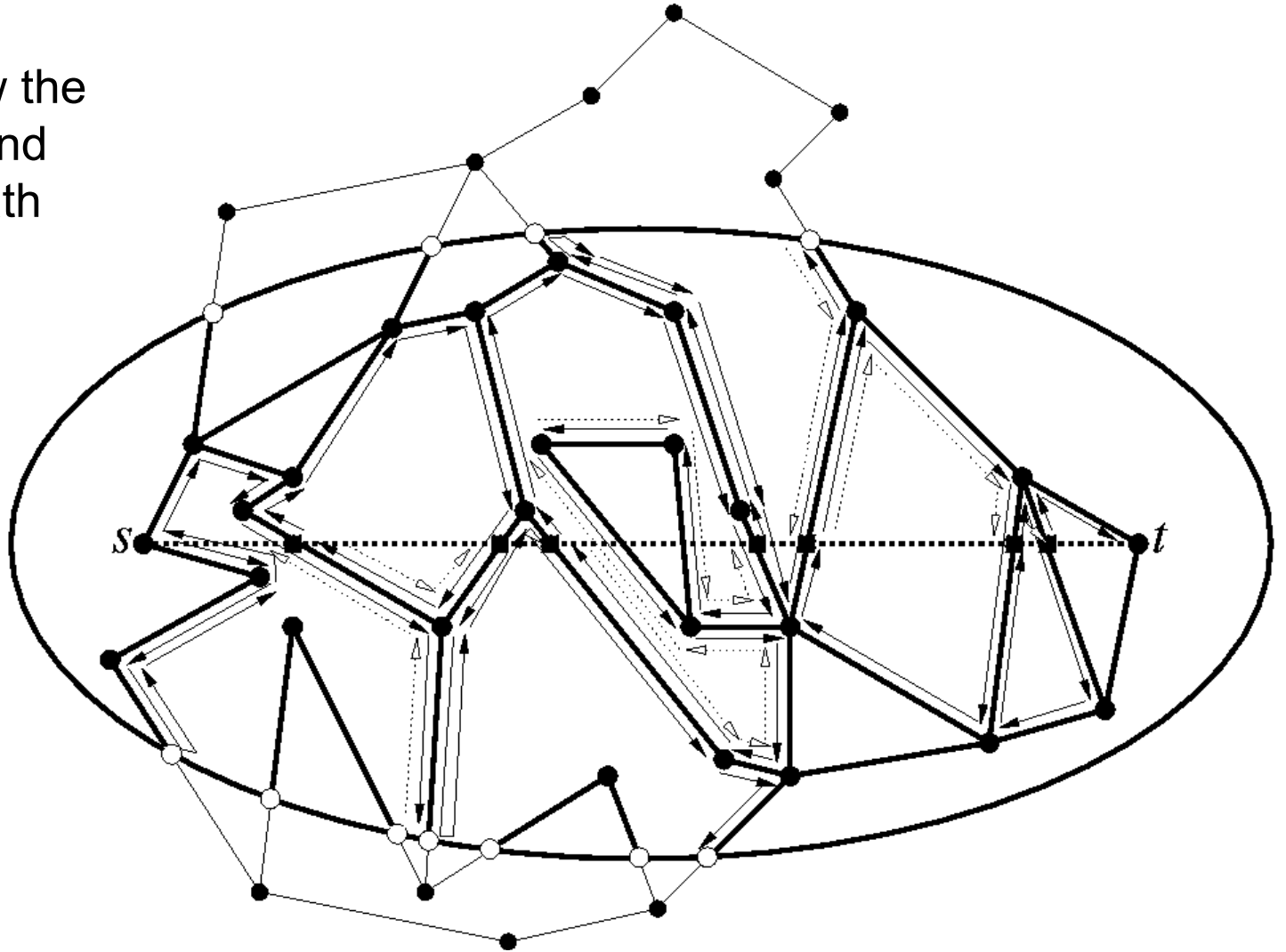
- Idea: Use face routing together with ad-hoc routing trick 1!!
- That is, don't route beyond some radius  $r$  by branching the planar graph within an ellipse of exponentially growing size.



# AFR Example Continued



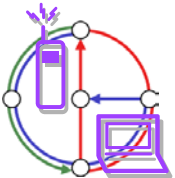
- We grow the ellipse and find a path



# AFR Pseudo-Code



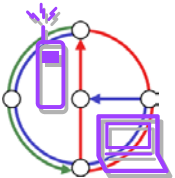
0. Calculate  $G = GG(V) \cap UDG(V)$   
Set  $c$  to be twice the Euclidean source—destination distance.
  1. Nodes  $w \in W$  are nodes where the path  $s-w-t$  is larger than  $c$ . Do face routing on the graph  $G$ , but without visiting nodes in  $W$ . (This is like pruning the graph  $G$  with an ellipse.) You either reach the destination, or you are stuck at a face (that is, you do not find a better exit point.)
  2. If step 1 did not succeed, double  $c$  and go back to step 1.
- Note: All the steps can be done completely locally, and the nodes need no local storage.



# The $\Omega(1)$ Model



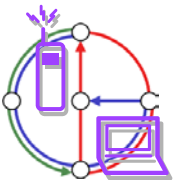
- We simplify the model by assuming that nodes are sufficiently far apart; that is, there is a constant  $d_0$  such that all pairs of nodes have at least distance  $d_0$ . We call this the  $\Omega(1)$  model.
- This simplification is natural because nodes with transmission range 1 (the unit disk graph) will usually not “sit right on top of each other”.
- Lemma: In the  $\Omega(1)$  model, all natural cost models (such as the Euclidean distance, the energy metric, the link distance, or hybrids of these) are equal up to a constant factor.
- Remark: The properties we use from the  $\Omega(1)$  model can also be established with a backbone graph construction.



# Analysis of AFR in the $\Omega(1)$ model



- Lemma 1: In an ellipse of size  $c$  there are at most  $O(c^2)$  nodes.
- Lemma 2: In an ellipse of size  $c$ , face routing terminates in  $O(c^2)$  steps, either by finding the destination, or by not finding a new face.
- Lemma 3: Let the optimal source—destination route in the UDG have cost  $c^*$ . Then this route  $c^*$  must be in any ellipse of size  $c^*$  or larger.
- Theorem: AFR terminates with cost  $O(c^{*2})$ .
- Proof: Summing up all the costs until we have the right ellipse size is bounded by the size of the cost of the right ellipse size.

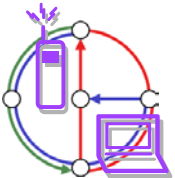
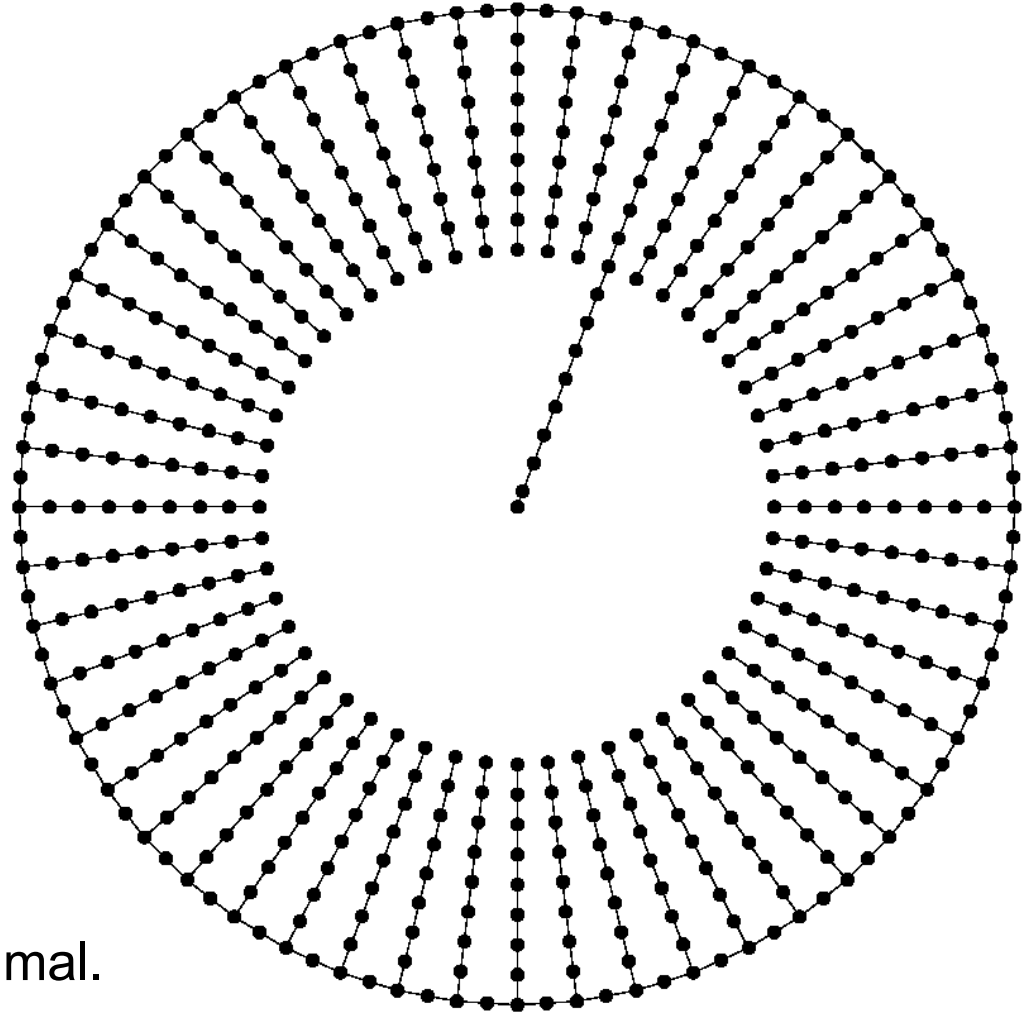




# Lower Bound



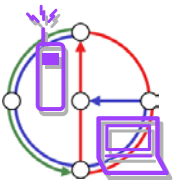
- The network on the right constructs a lower bound.
- The destination is the center of the circle, the source any node on the ring.
- Finding the right chain costs  $\Omega(c^2)$ , even for randomized algorithms
- Theorem: AFR is asymptotically optimal.



# Non-geometric routing algorithms



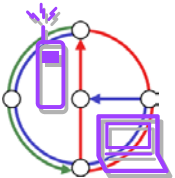
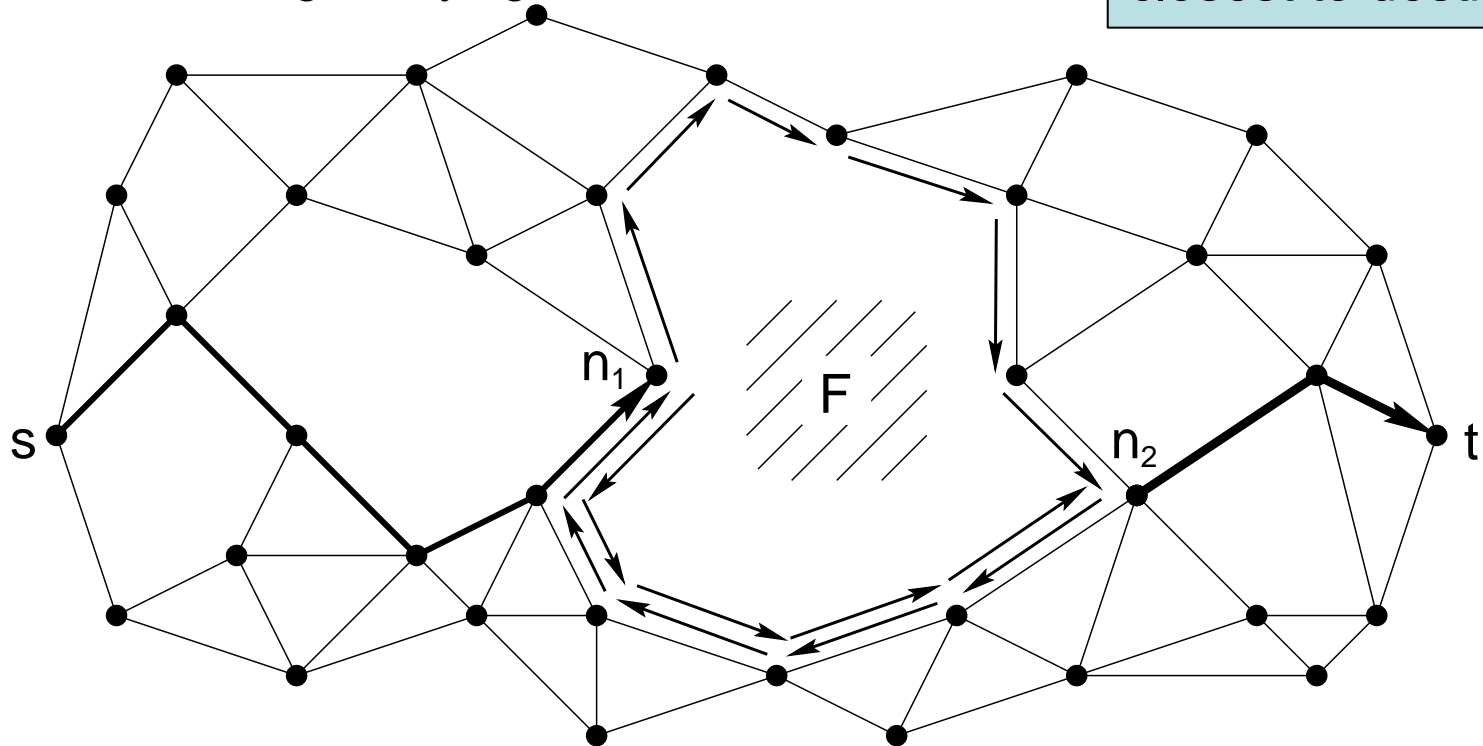
- In the  $\Omega(1)$  model, a standard flooding algorithm enhanced with trick 1 will (for the same reasons) also cost  $O(c^2)$ .
- However, such a flooding algorithm needs  $O(1)$  extra storage at each node (a node needs to know whether it has already forwarded a message).
- Therefore, there is a 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 **G**reedy and (**O**ther **A**daptive) **F**ace **R**outing
  - Route greedily as long as possible
  - Circumvent “dead ends” by use of face routing
  - Then route greedily again

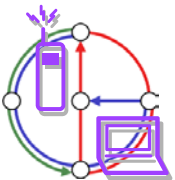
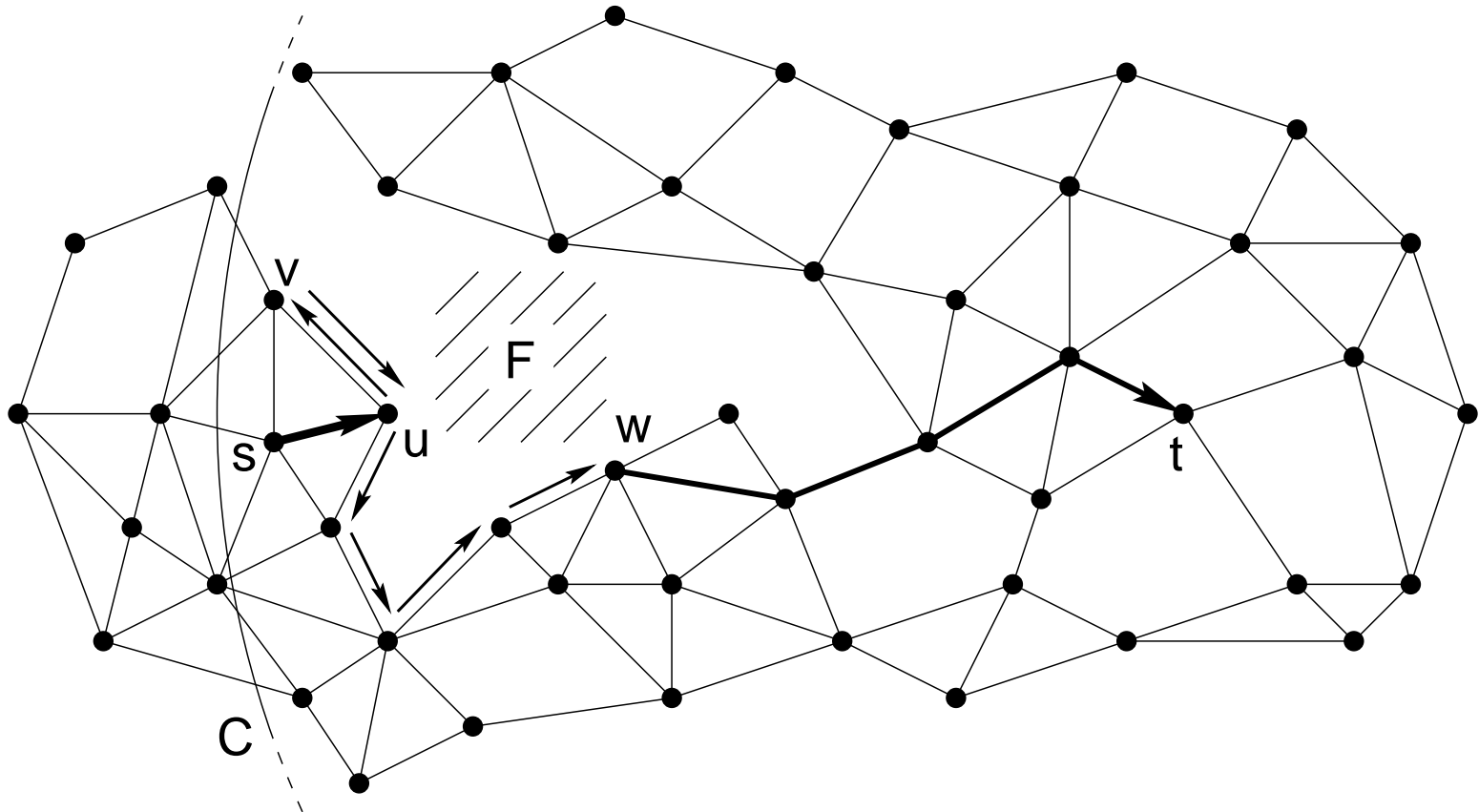
Other AFR: In each face proceed to node closest to destination



# GOAFR+



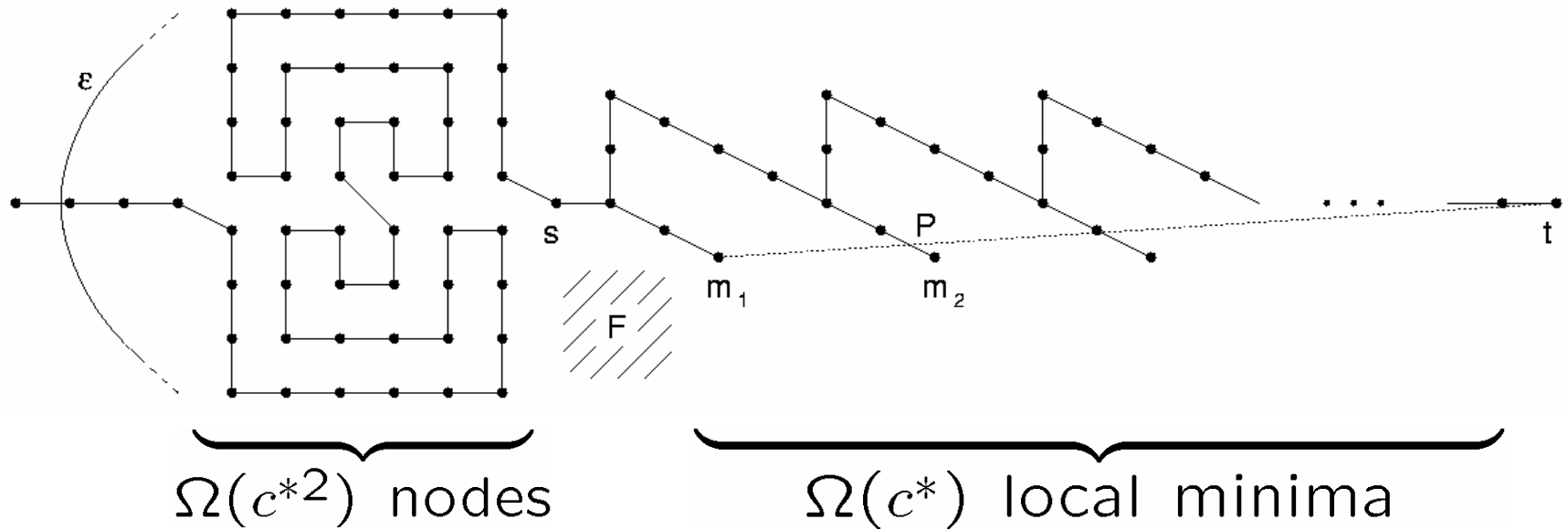
- GOAFR+ improvements:
  - Early fallback to greedy routing
  - (Circle centered at destination instead of ellipse)



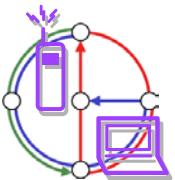
# GOAFR+ — Early Fallback



- 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  $\rightarrow \Omega(c^{*3})$  steps



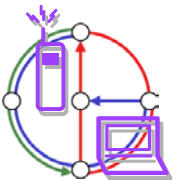
# GOAFR – Greedy Other Adaptive Face Routing



- 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.

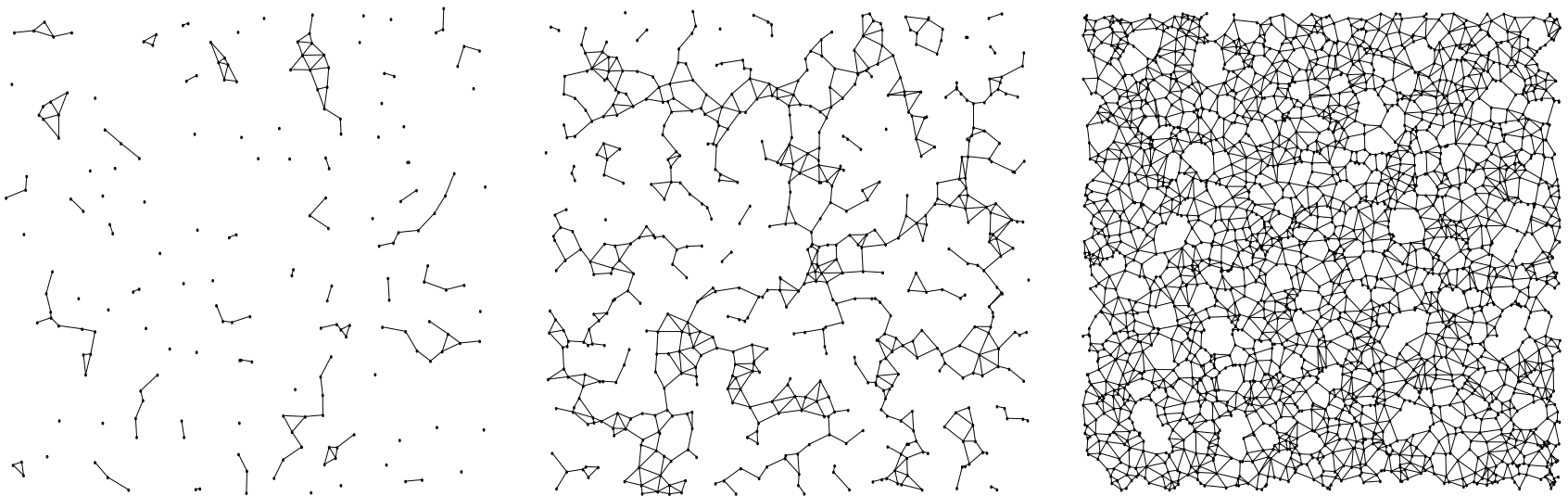
- What does “practice” mean?
  - Usually nodes placed uniformly at random



# Average Case



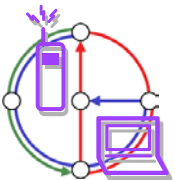
- Not interesting when graph not dense enough
- Not interesting when graph is too dense
- **Critical density range** (“percolation”)
  - Shortest path is significantly longer than Euclidean distance



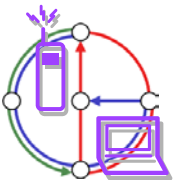
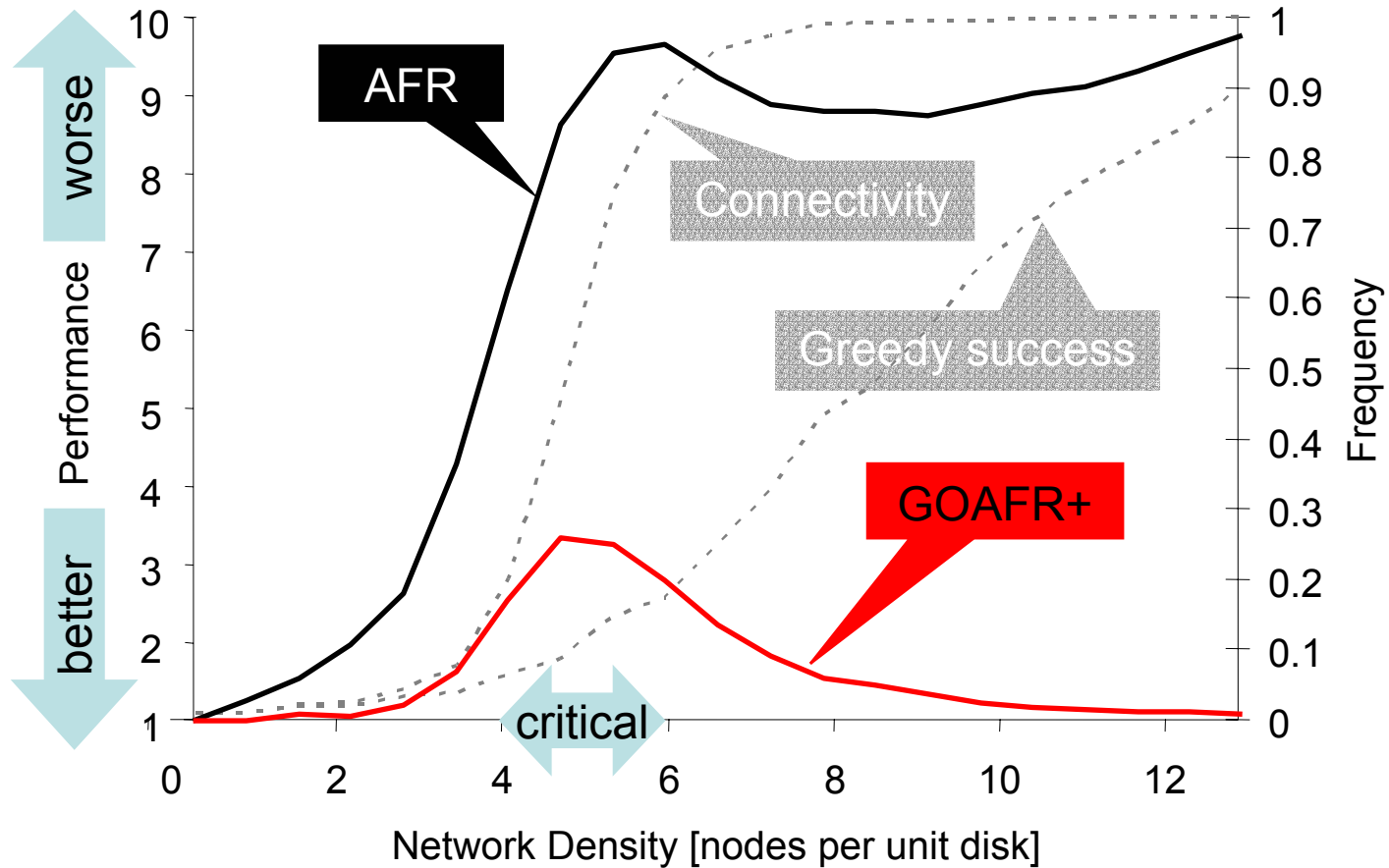
← too sparse

critical density

→ too dense



# Simulation on Randomly Generated Graphs

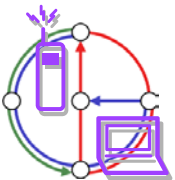




# A Word on Performance



- 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  $3.3 \cdot c$  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

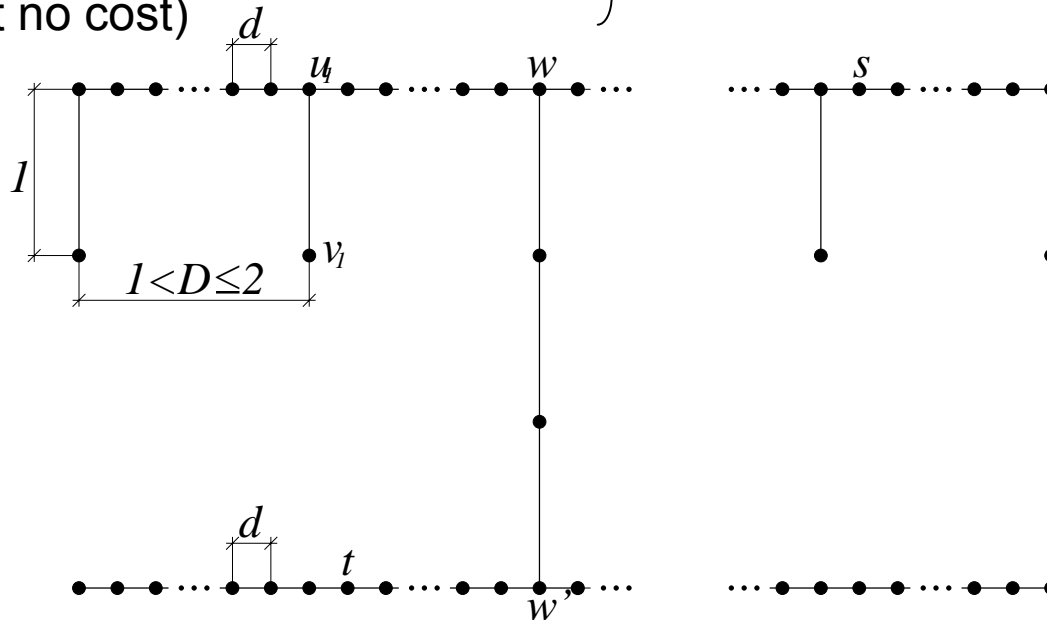


# Energy Metric Lower Bound

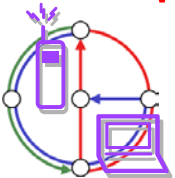
Example graph:  $k$  “stalks”, of which only one leads to  $t$

- 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)

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



→ With energy metric there is no competitive geometric routing algorithm



# Milestones in Geometric Routing



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

