Location Awareness in Wireless Computing

By
Evangelos Kranakis
School of Computer Science
Carleton University

Lots of Wireless Systems

Wireless networks ranging from

- Piconets, Bluetooth Networks (Home/Office Networks)
- Sensor Networks
- (Packet) Radio Networks
- Cellular phone Networks
- Satellite Networks

are becoming all too pervasive in our everyday lives.

Important Question:

How do you navigate efficiently in such a wireless system?

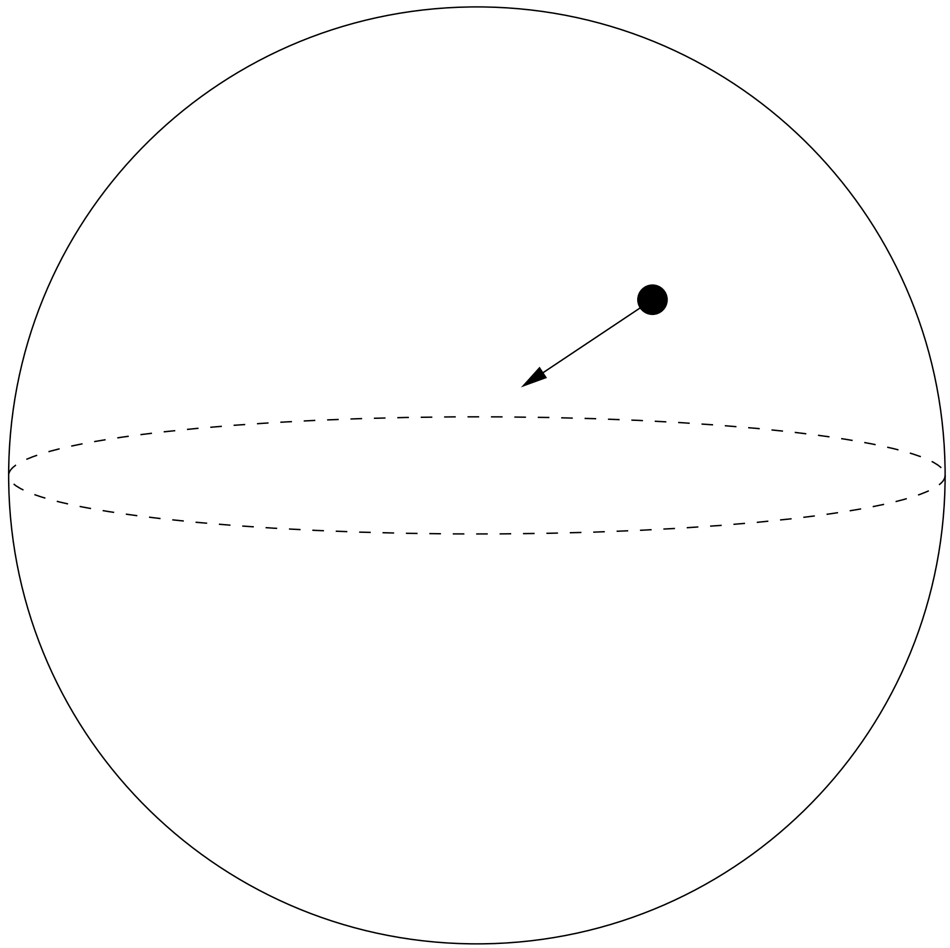
Location Awareness Paradigm

An ideal wireless system should be infrastructureless!

- Evolving wireless networks (adhoc, sensor) take advantage of location awareness of the hosts.
- Wireless networking is made possible by tracking hosts' every movement.

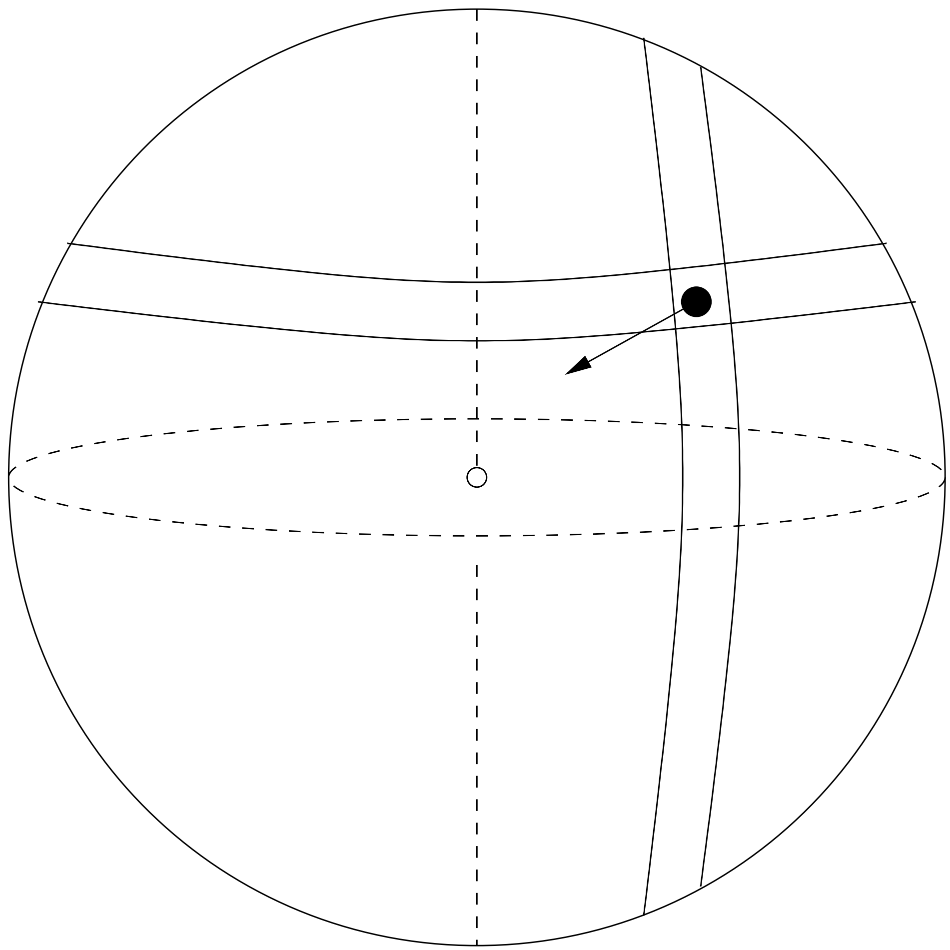
Lots of questions raised: from efficiency of wireless computing to privacy!

How do you find your location when traveling?



Important problem in ship navigation!

Use longitude and lattitude



Easy to find the lattitude! But the longitude?

Lets step back! The longitude problem's "who is who"

- Werner, 1514 (Moon travels its diameter every hour)
- Gallileo, 1610 (Suggestion to use Jupiter's moons)
- Huygens, 1658 (Published the "Horologium")
- Hooke, ≈1660 (Accuses Huygens he stole "spring" concept)
- Digby, 1687 (Wounded dog theory and powder of sympathy)
- Cassini, 1668 (Used Jupiter's moons)
- Roemer, 1676 (Observations depended on speed of light)
- St Pierre, 1678 (Use earth's moon plus select stars)
- Fyler, 1699 (Identify 24 discrete rows of stars in sky)
- Ditton+Whinston, 1713 (System of lightning and sound blasts)

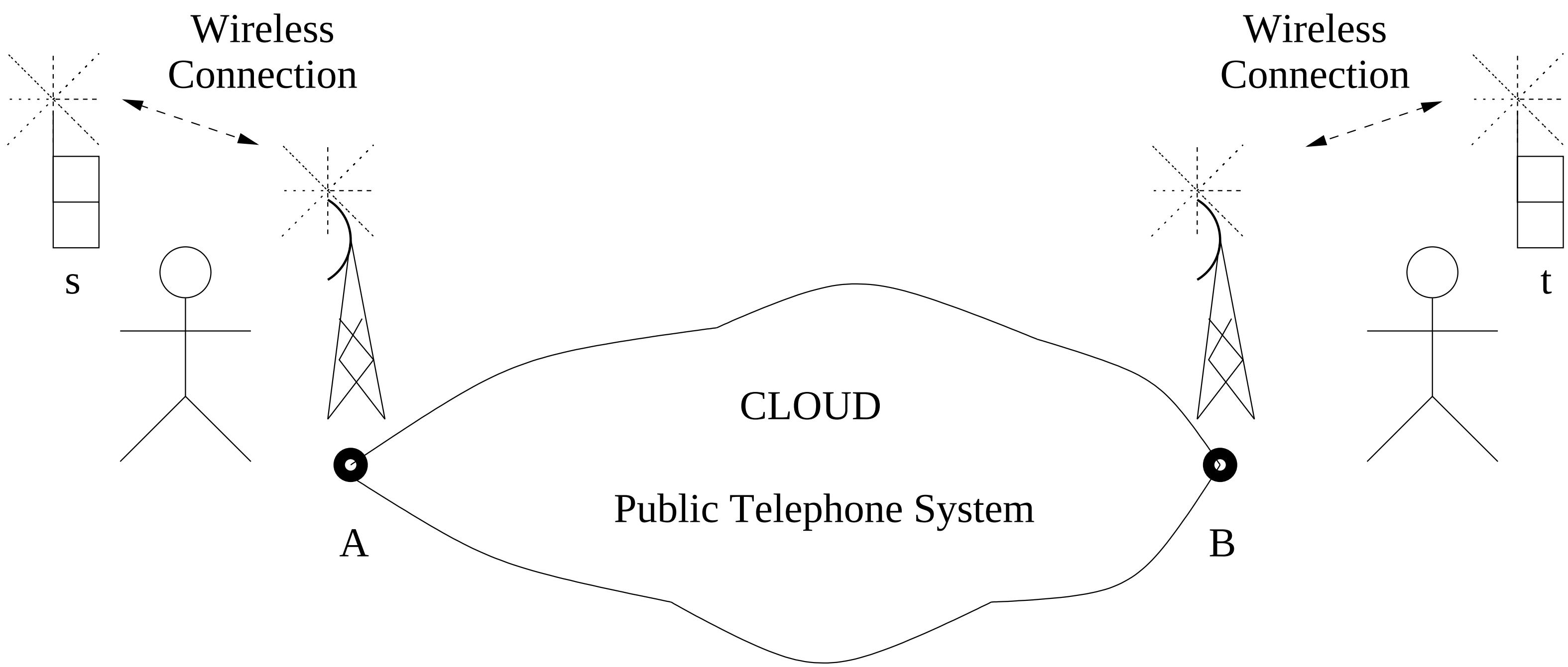
The longitude problem's "who is who"

July 15, 1714, Longitude Act: 20,000 pound prize for method to determine longitude to an accuracy of half a degree of a great circle.

- Thacker, 1714 (Chronometer adjusted for temperature)
- John "Longitude" Harrison (self-taught)
  - 1713 (Pendulum clock)
  - H-1 (Made with specially crafted "self-lubricating" wooden gears)
- Flamsteed, 1725 (Published catalogue of stars)

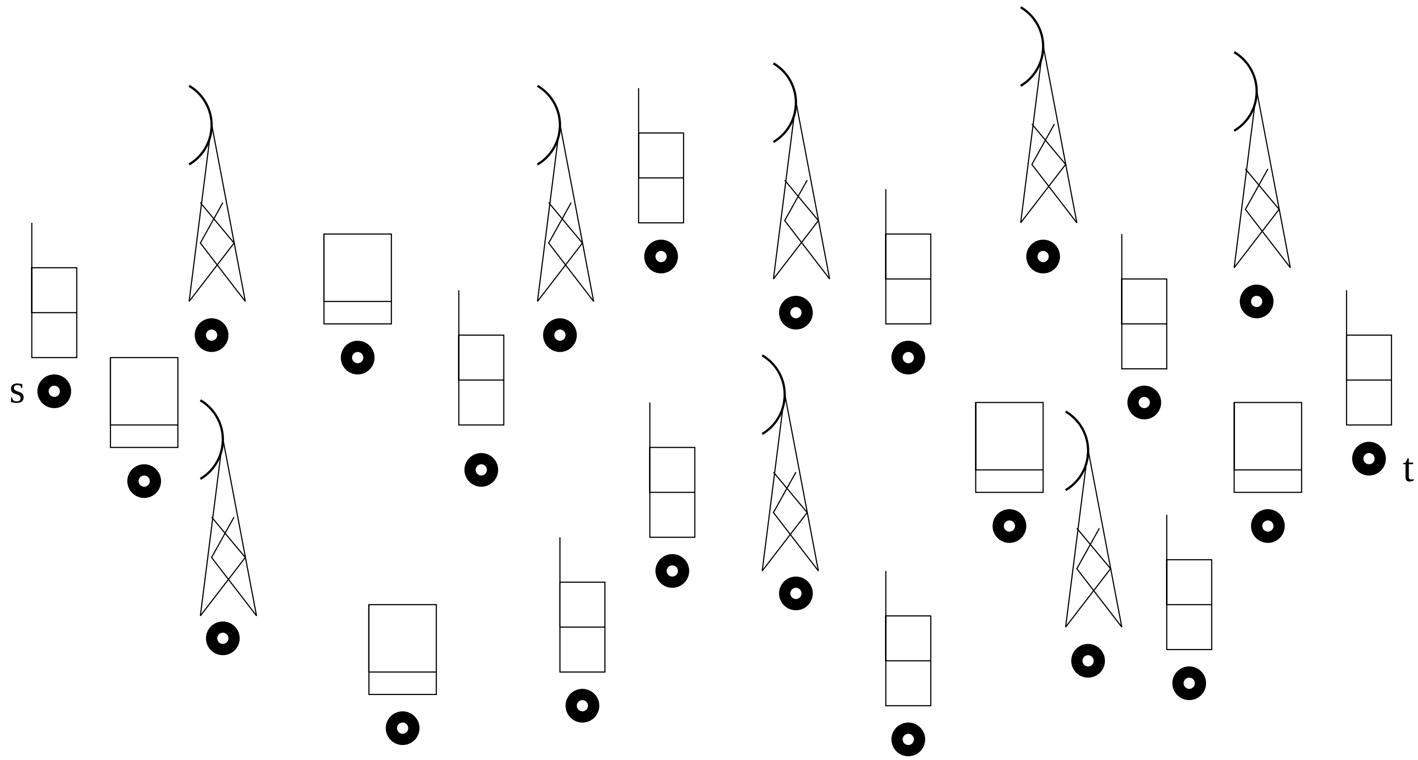
The Way it is today

How do you discover a route from a source s to a destination t?



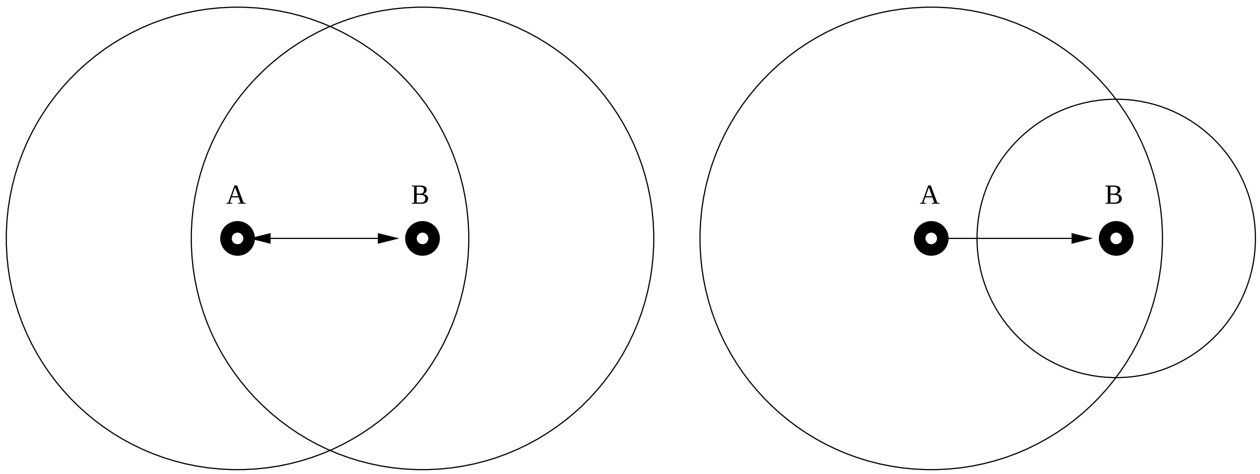
1. s sends message to base station A.
2. The public phone system transmits it to base station B.
3. Base station B transmits it to t.

Routing Data from s to t in a Wireless Ad-Hoc System



How do you discover a route from a source s to a destination t through a maze of mobile devices (radios, mobile phones, PDAs, etc) and antenas?

Power of the Signal and Connectivity

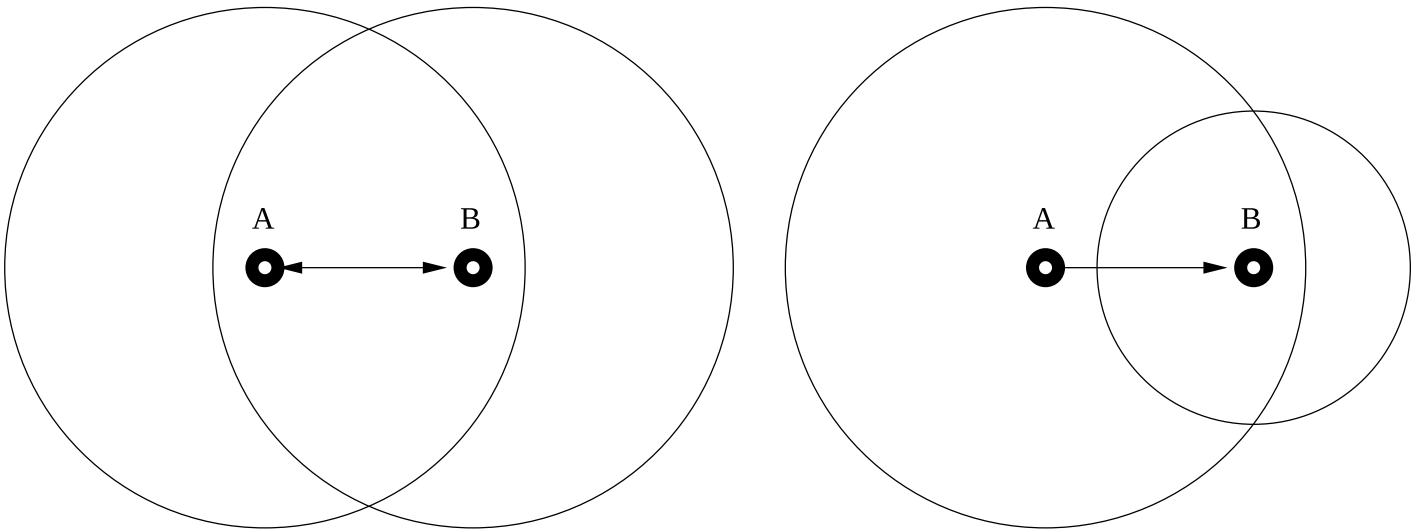


The power of received signal is inversely proportional to the distance d between the receivers raised to some power a, i.e.,

$$P(d) = \frac{P_0}{d^a},$$

where P_0 is the power received at distance 1 from the transmitter.

Equal Power Assumption!

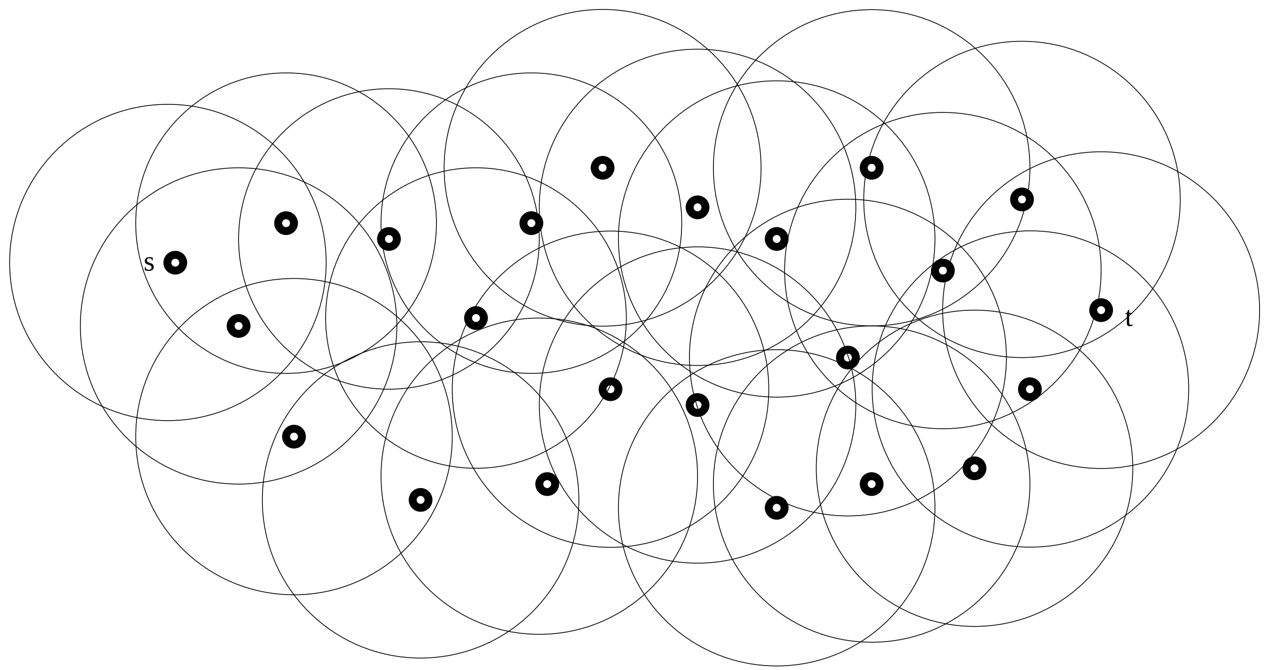


To simplify things stations are assumed to have equal power!

In the left picture A can reach B and B can reach A. In the right picture A can reach B but B cannot reach A.

Later we will remove the equal power assumption!

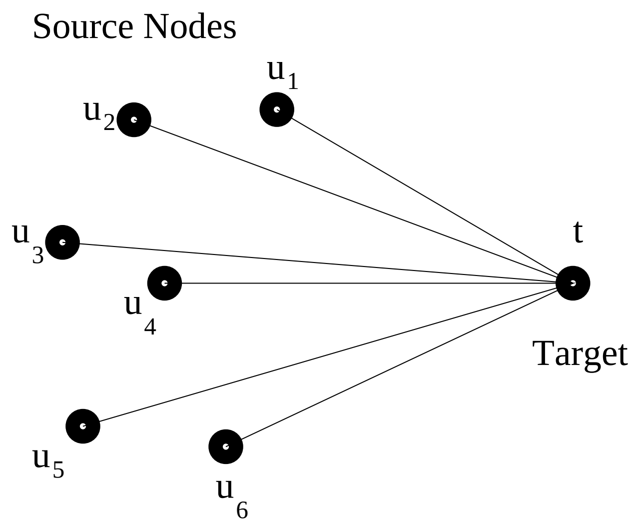
From Mobile Devices to Circles



Looking at the power of received signal a group of circles is formed that determines network connectivity, i.e. who can reach whom!

There are Other Factors Complicating Connectivity!

More than one node may be transmiting to a given node t.



Node i succeeds only if

$$\frac{\frac{P(u_i,t)}{d(u_i,t)^a}}{N + \sum_{j \neq i} \frac{P(u_j,t)}{d(u_j,t)^a}} \geq \alpha$$

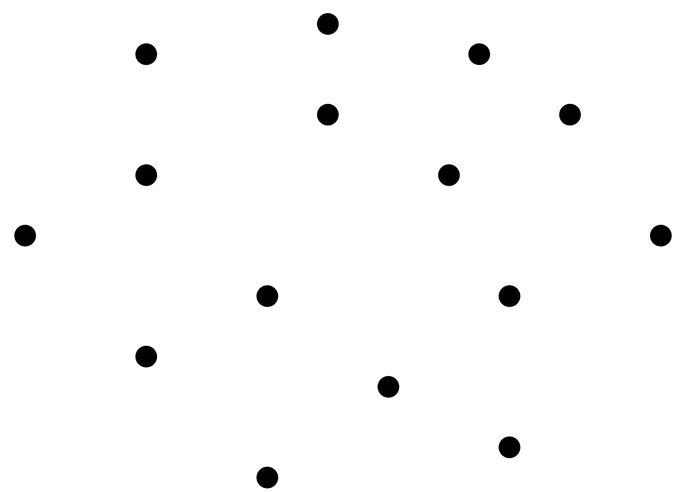
where α is a threshold value, and N is the noise level

We ignore these factors!

Defining Location Awareness: The Neighborhood Graph

The neighborhood graph $G_{\mathcal{S},\mathcal{P}}$ of a planar pointset P is determined by

P



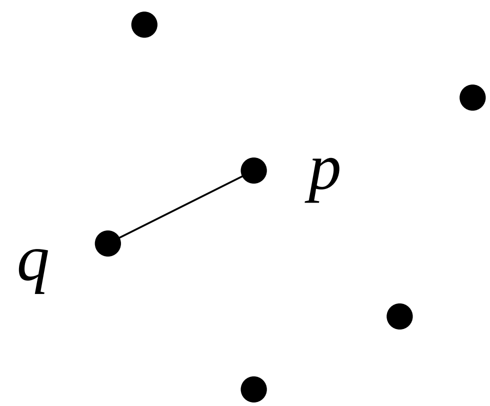
1. $(p, q) \to S_{p,q} \subseteq R^2$, for $p, q \in P$.
2. $\mathcal{P}$ is a property on $\mathcal{S} := \{S_{p,q} : p, q \in P\}$.

Graph $G_{\mathcal{S},\mathcal{P}} = (P, E)$:
$(p, q) \in E \Leftrightarrow S_{p,q}$ has property $\mathcal{P}$.

Nearest Neighbor Graphs

Nearest Neighbor Graph (NNG):

$(p, q) \in E \Leftrightarrow p$ is nearest neighbor of q.

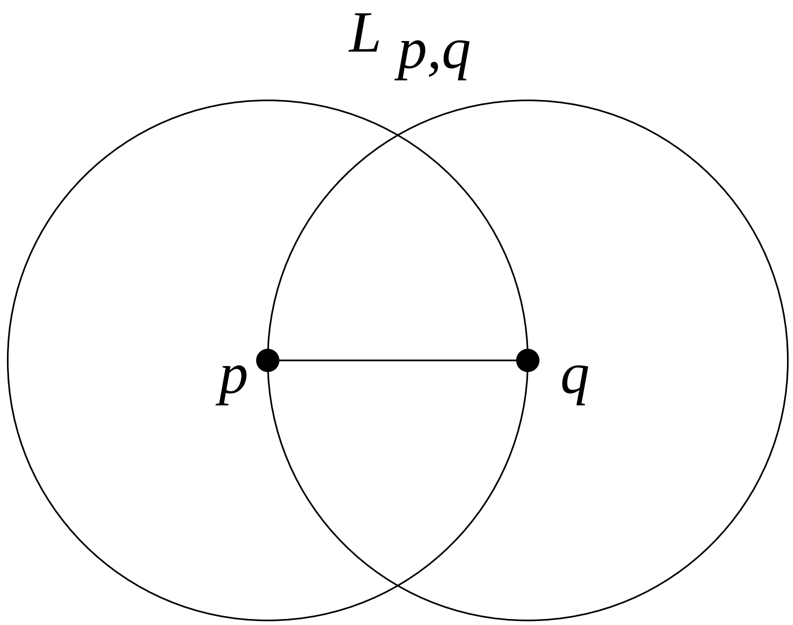


k-Nearest Neighbor Graph (k-NNG):

$(p, q) \in E \Leftrightarrow p$ is k-th nearest neighbor of q or q is k-th nearest neigbhor of p.

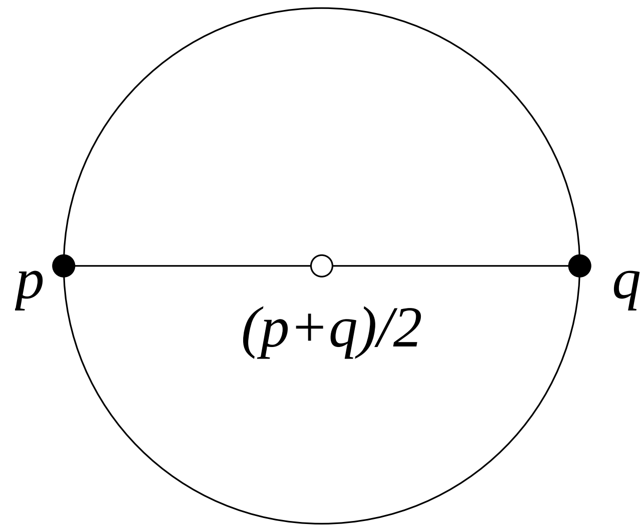
Relative Neighbor Graph (RNG)

The **lune** $L_{p,q}$ of p and q is the interscetion of the open discs with radius $d(p,q)$ and centered at p and q, respectively.



$(p,q) \in E \Leftrightarrow$ the lune $L_{p,q}$ does not contain any point in the pointset P.

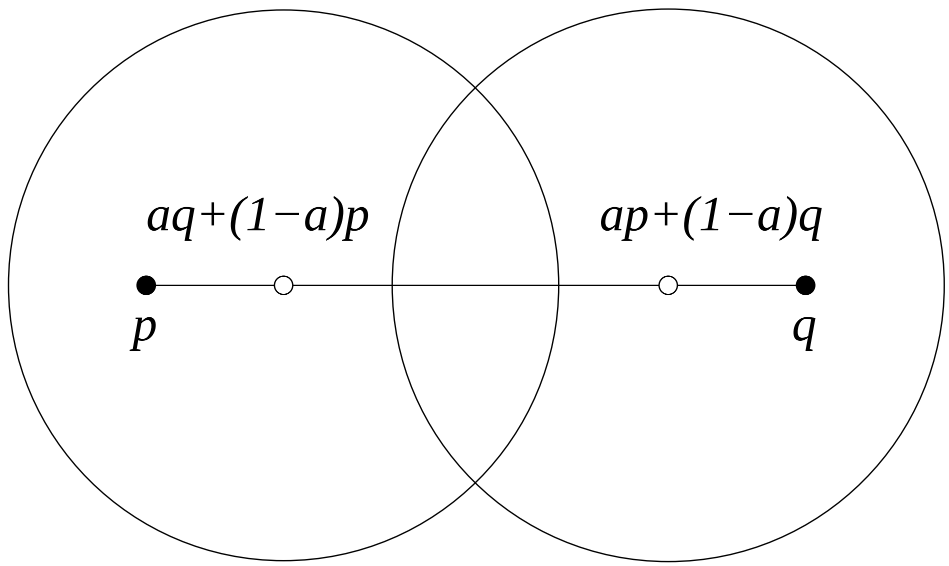
Gabriel Graph (GG)



$(p,q) \in E \Leftrightarrow$ the disc centered at $\frac{p+q}{2}$ and radius $\frac{d(p.q)}{2}$ does not contain any point in the pointset P.

α-Gabriel Graph (α-GG)

Assume $1/2 \leq a \leq 1$.



$(p, q) \in E \Leftrightarrow$ the intersection of the discs $D(p(1-a)+aq, ad(p,q))$ and $D(q(1-a)+ap, ad(p,q))$ does not contain any point in the pointset P.

The Important Questions

Something from Pattern Recognition

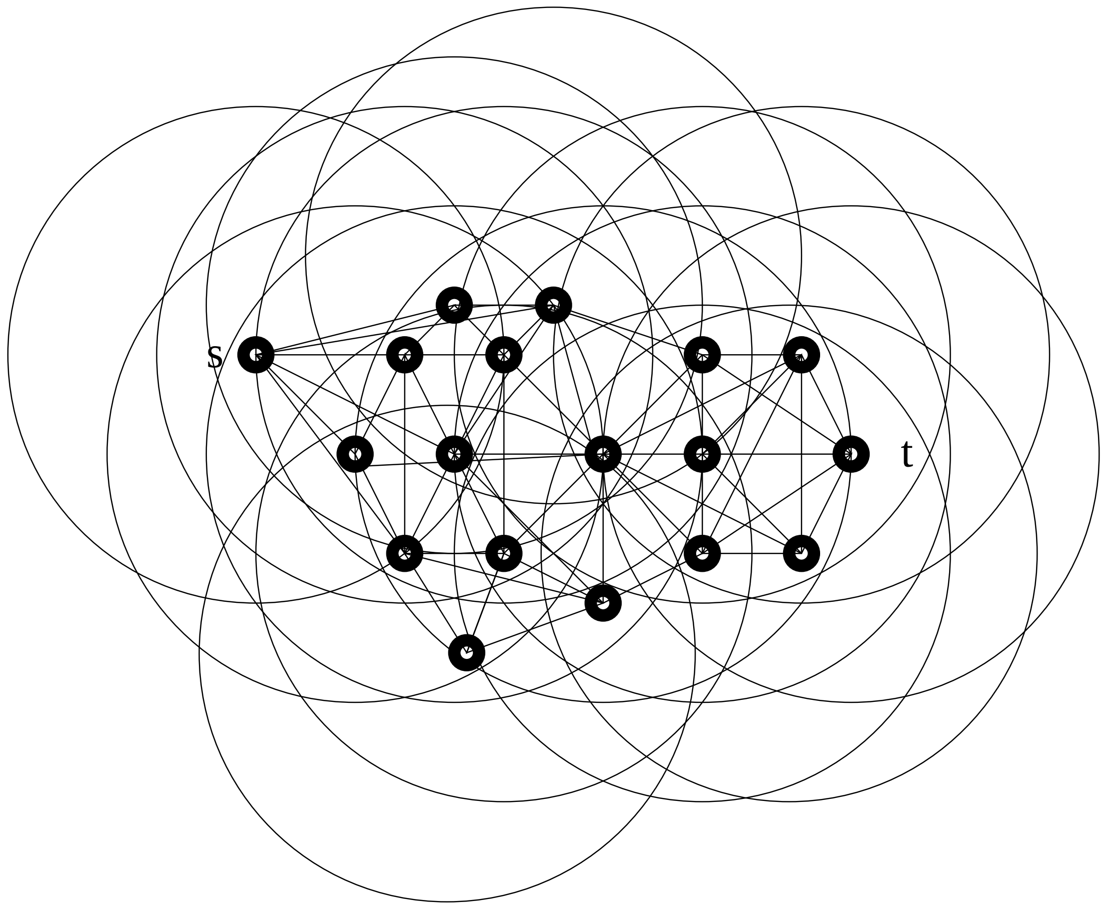
Theorem: (Toussaint, 1980) For a given planar pointset P,

$$NNG \subseteq MST \subseteq RNG \subseteq \alpha - GG \subseteq GG \subseteq DT$$

Of Relevance to Wireless Computing!

1. *How do you construct these graphs?*
2. *Can you base your constructions only on local information?*
3. *Can you use these graphs in order to navigate in a wireless system?*

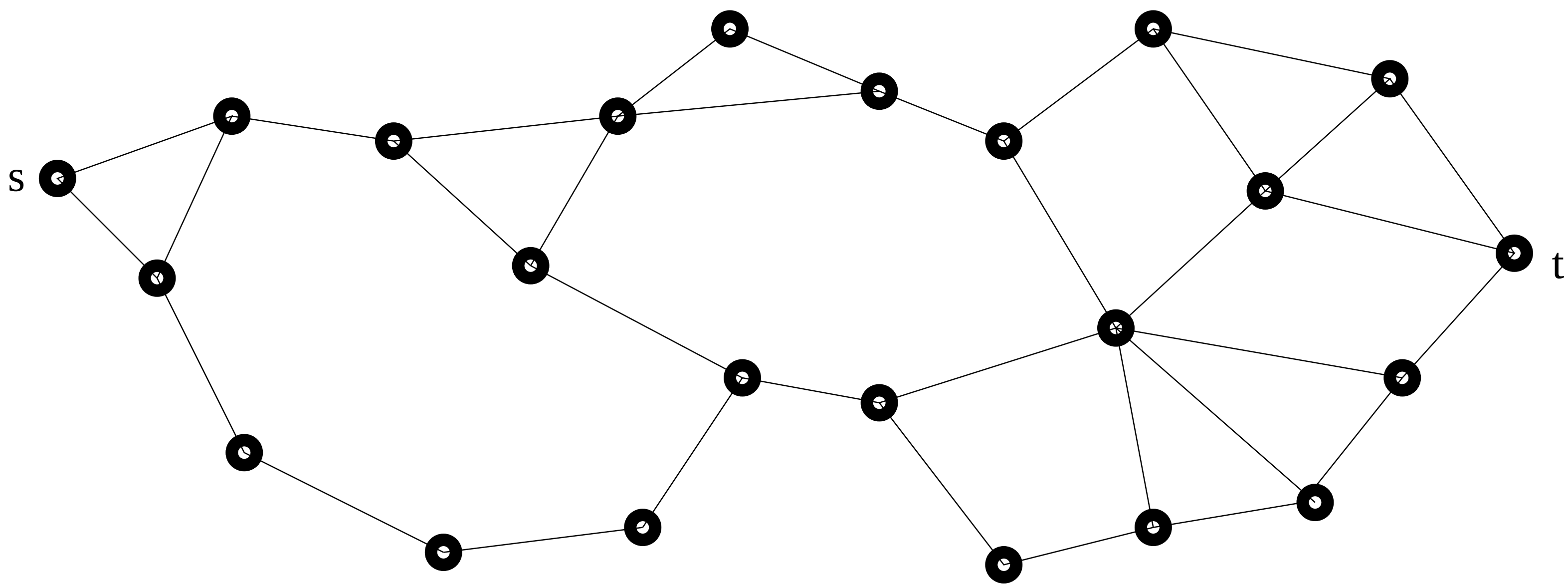
Constructing Geometric Graph not Always Simple!



Graph is even more complicated when transmitters are too close to each other!

Spanners: From Circles to Edges

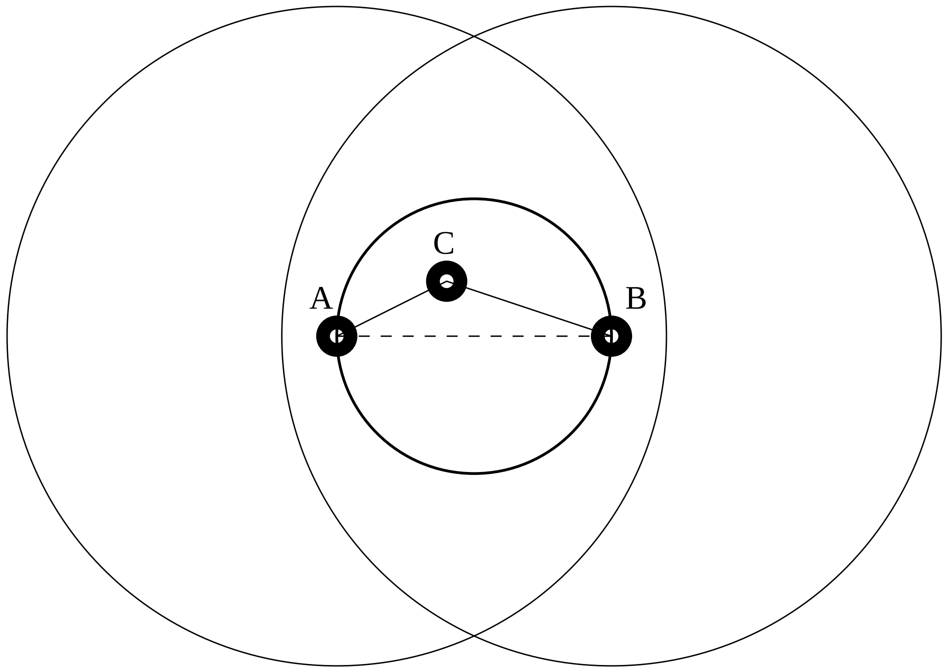
There is always an underlying geometric graph with *faces* and *edges* spanning the whole wireless network. It is called a spanner.



How do you construct this geometric graph from the original wireless system?

You must remove non-essential edges!

Gabriel Test



Assume points A and B can reach each other.

Draw circle with diameter AB. If there is another point, say C inside this circle then the link connecting A to B is not needed!

So, forget the direct link from A to B!

How do you forget something?

You maintain a *Routing Table*.

A kind of data base that when you are at A you ask:
How do I reach B?

It gives you the answer: Go to C.

And when you reach C you ask again:
How do I reach B?

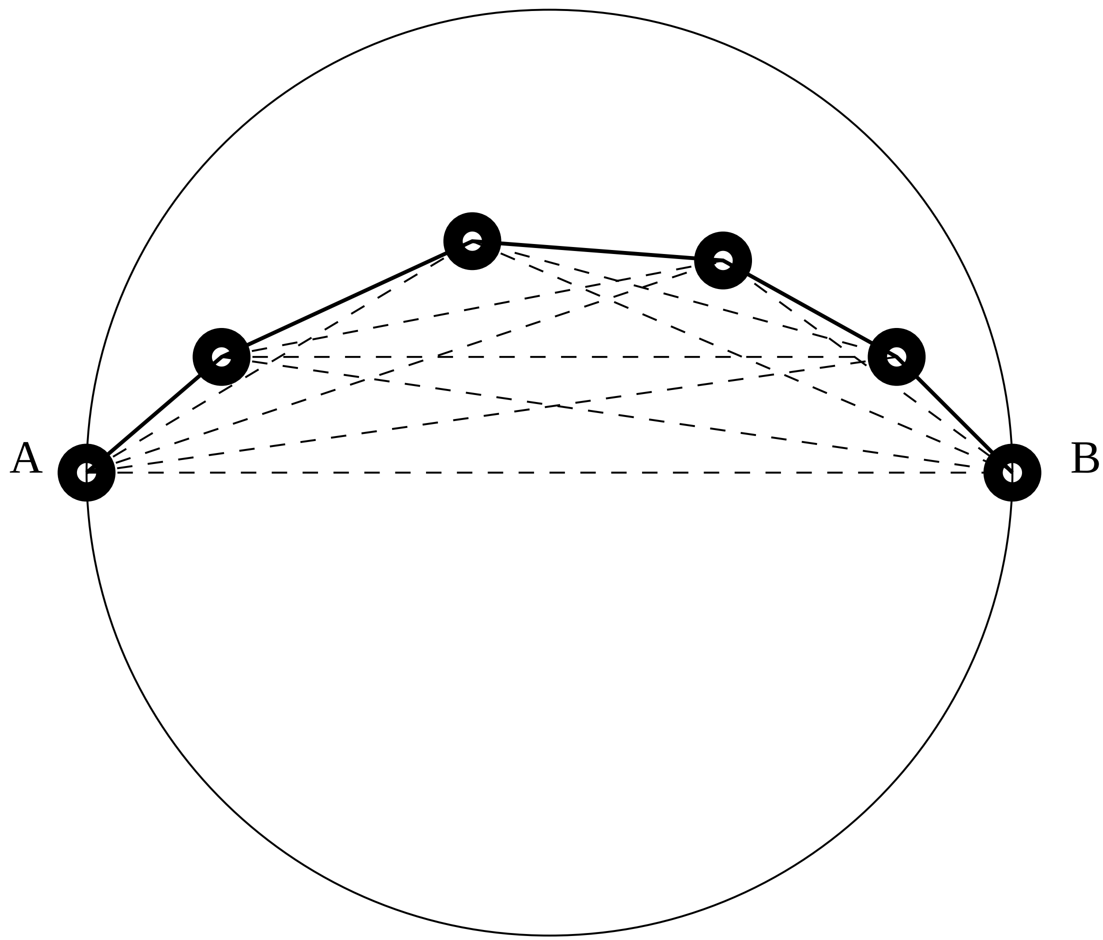
It gives you the answer: Go to B.

Standard routing table contains an entry for each possible destination with the out-going link to use for destination

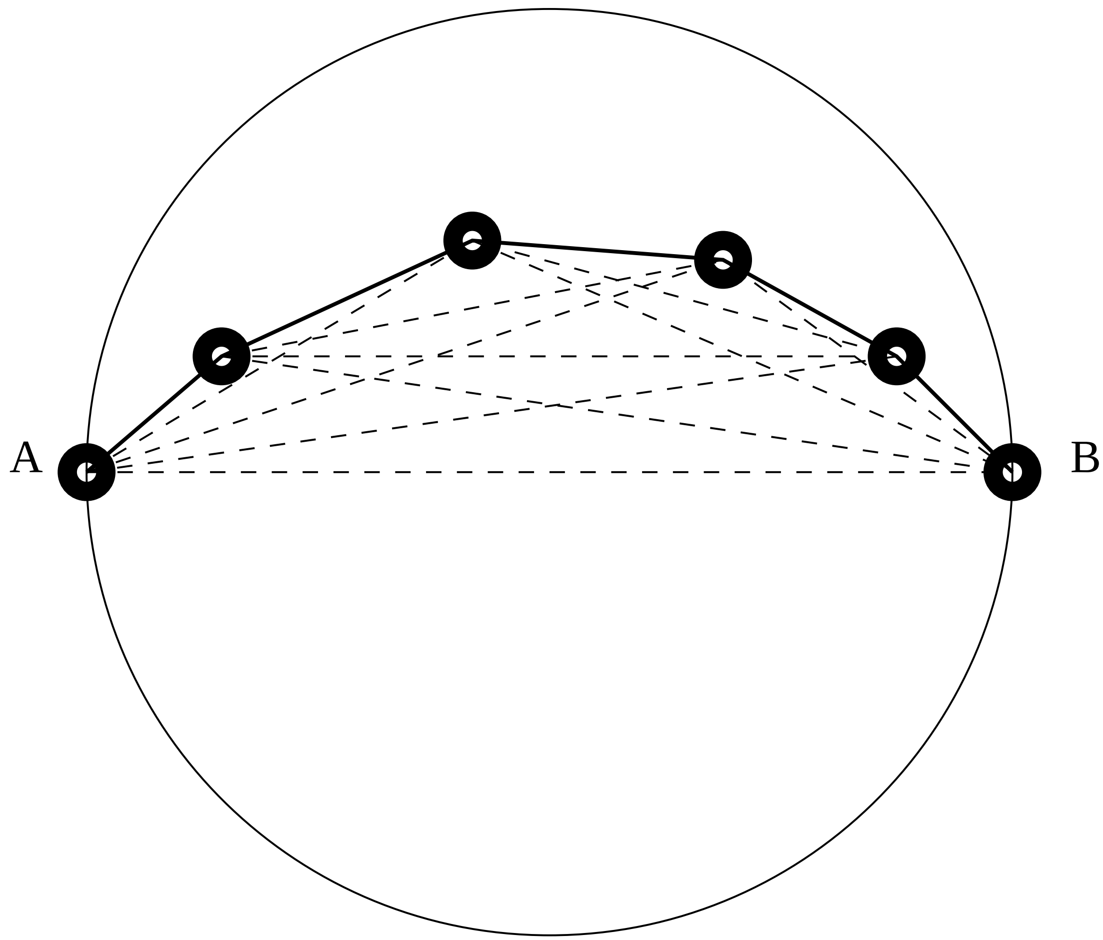
Message delivery proceeds in the obvious manner one link at a time, looking up the next link in the table.

Simple Example: Applying the Gabriel Test

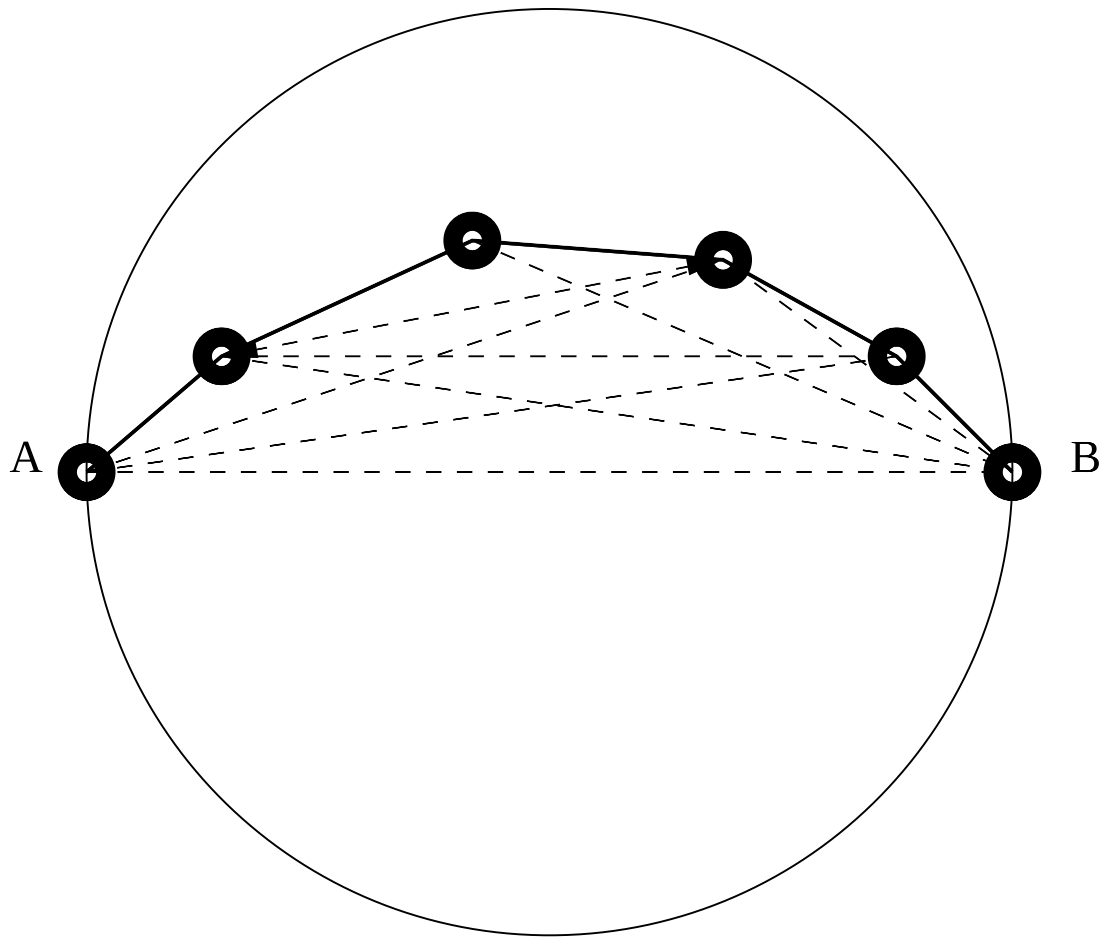
The Gabriel test is a distributed algorithm.



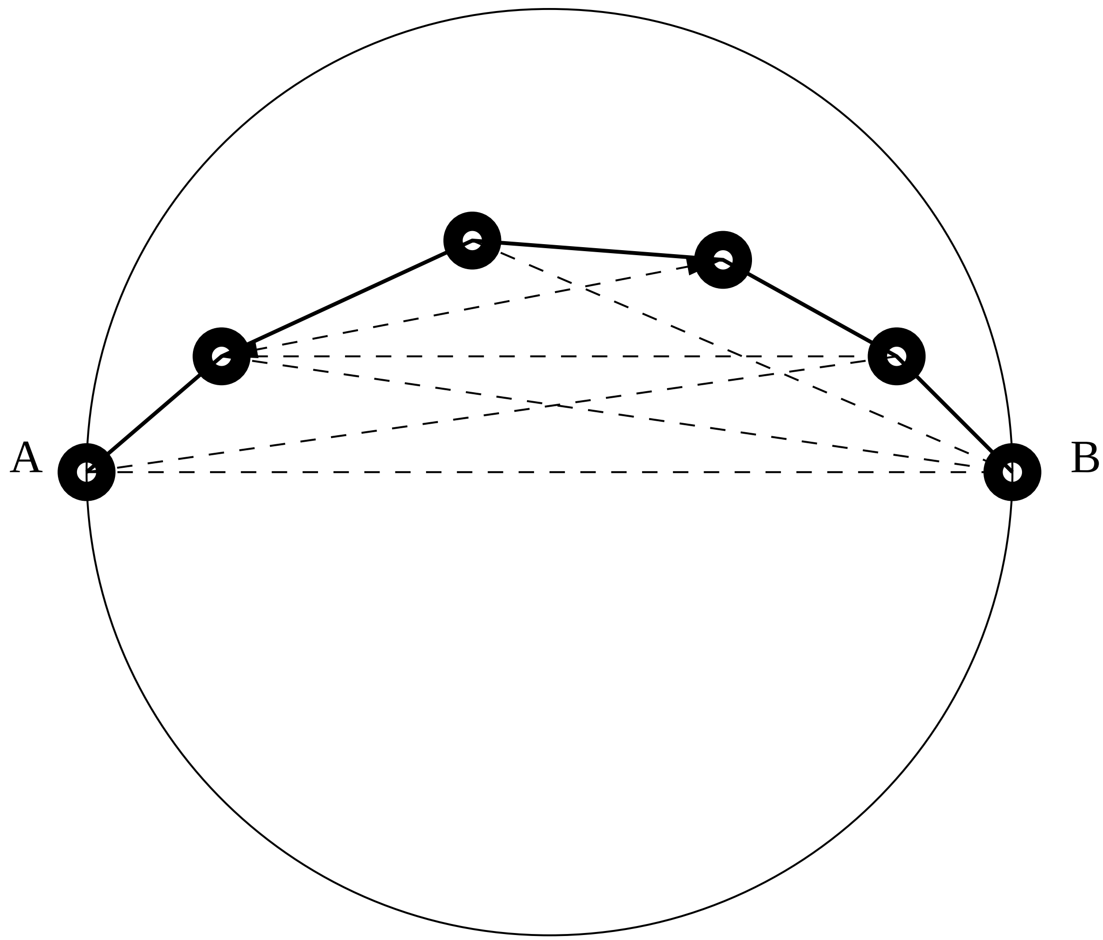
Applying the Gabriel Test



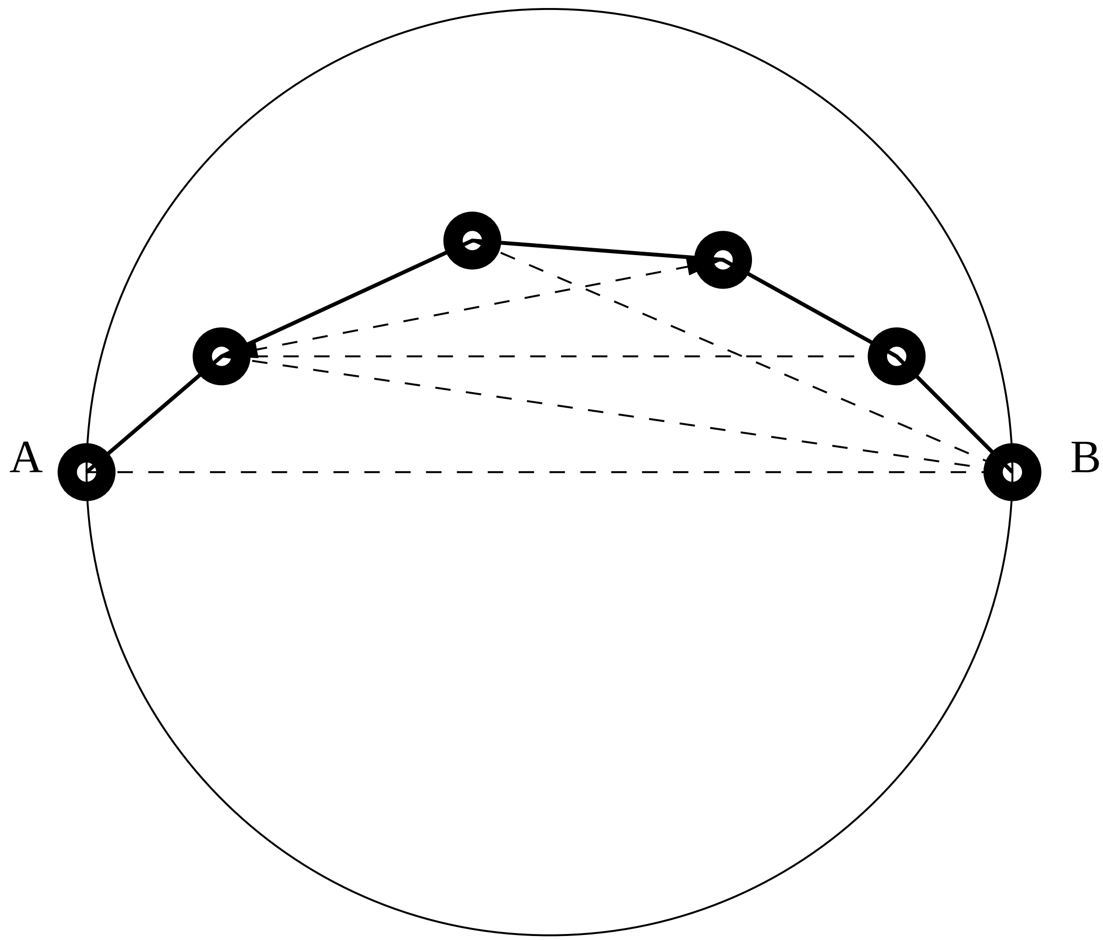
Applying the Gabriel Test



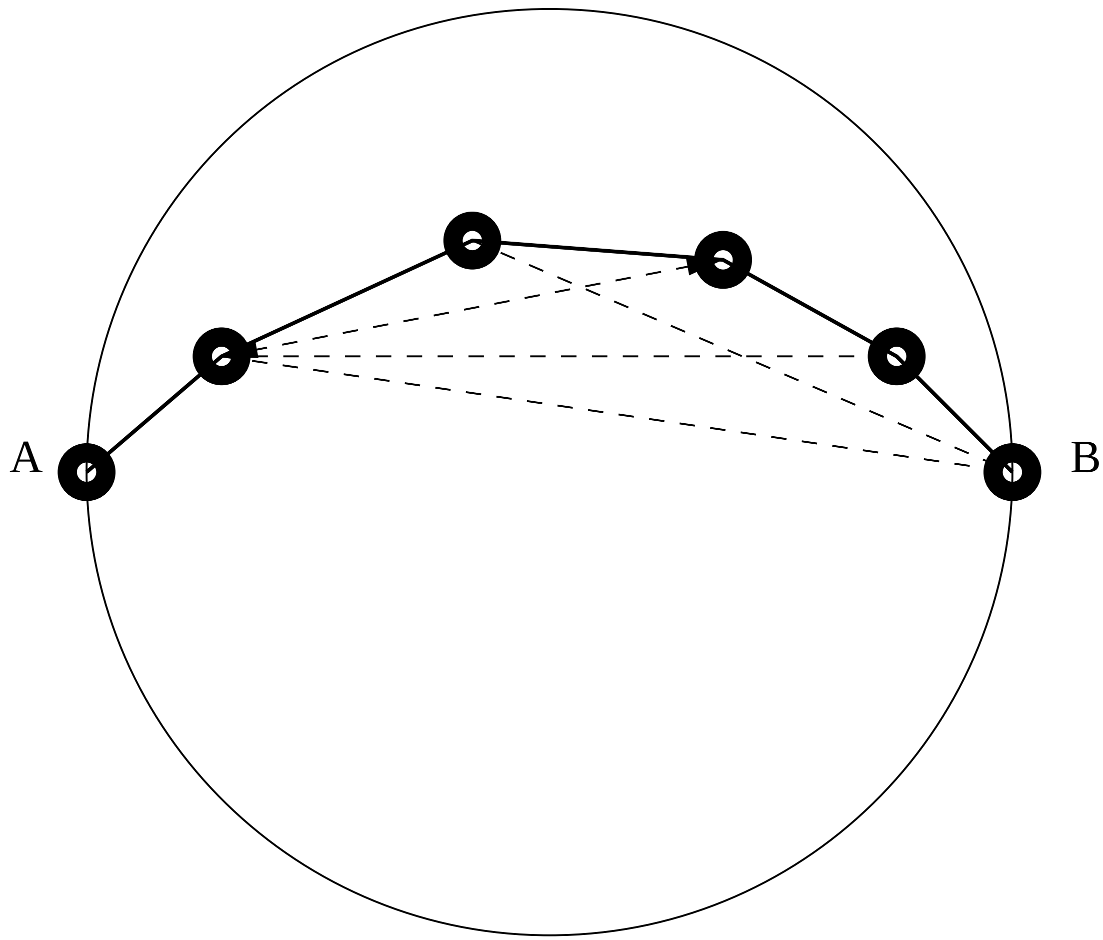
Applying the Gabriel Test



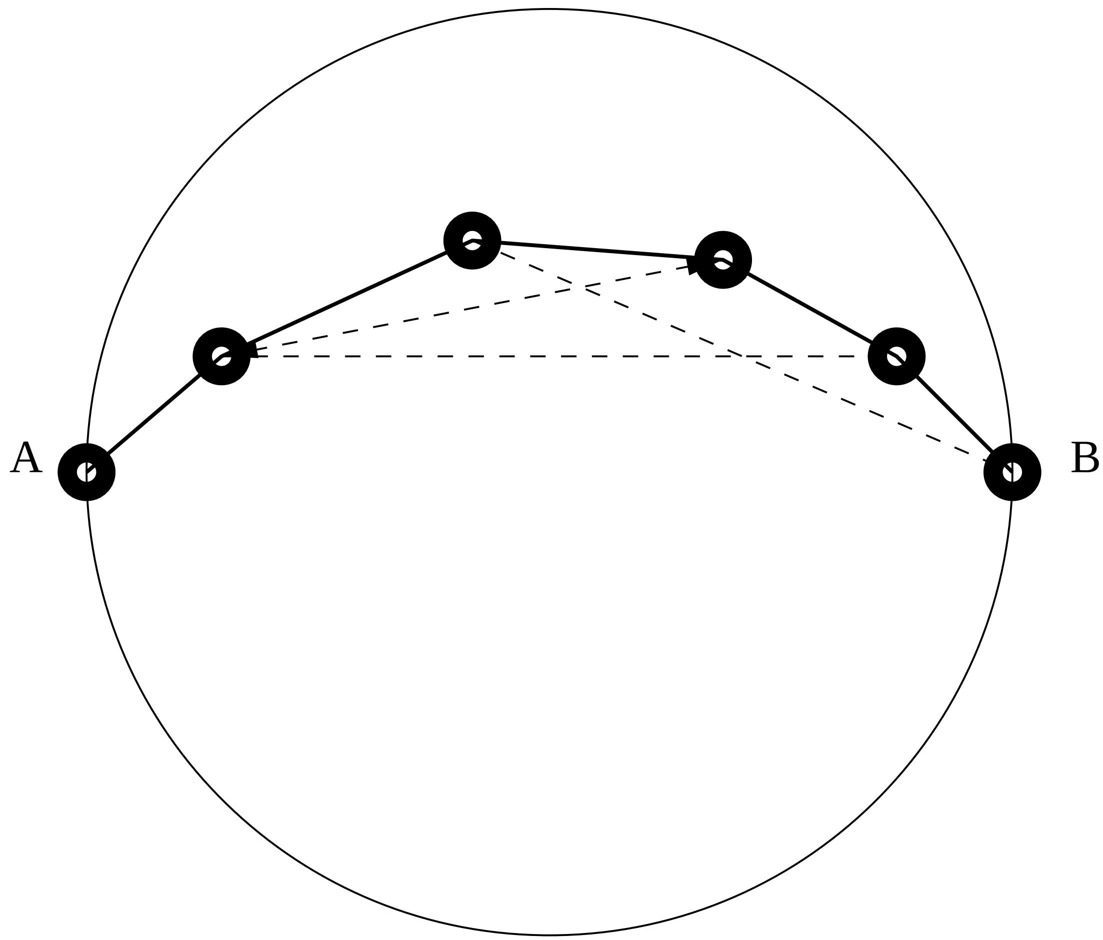
Applying the Gabriel Test



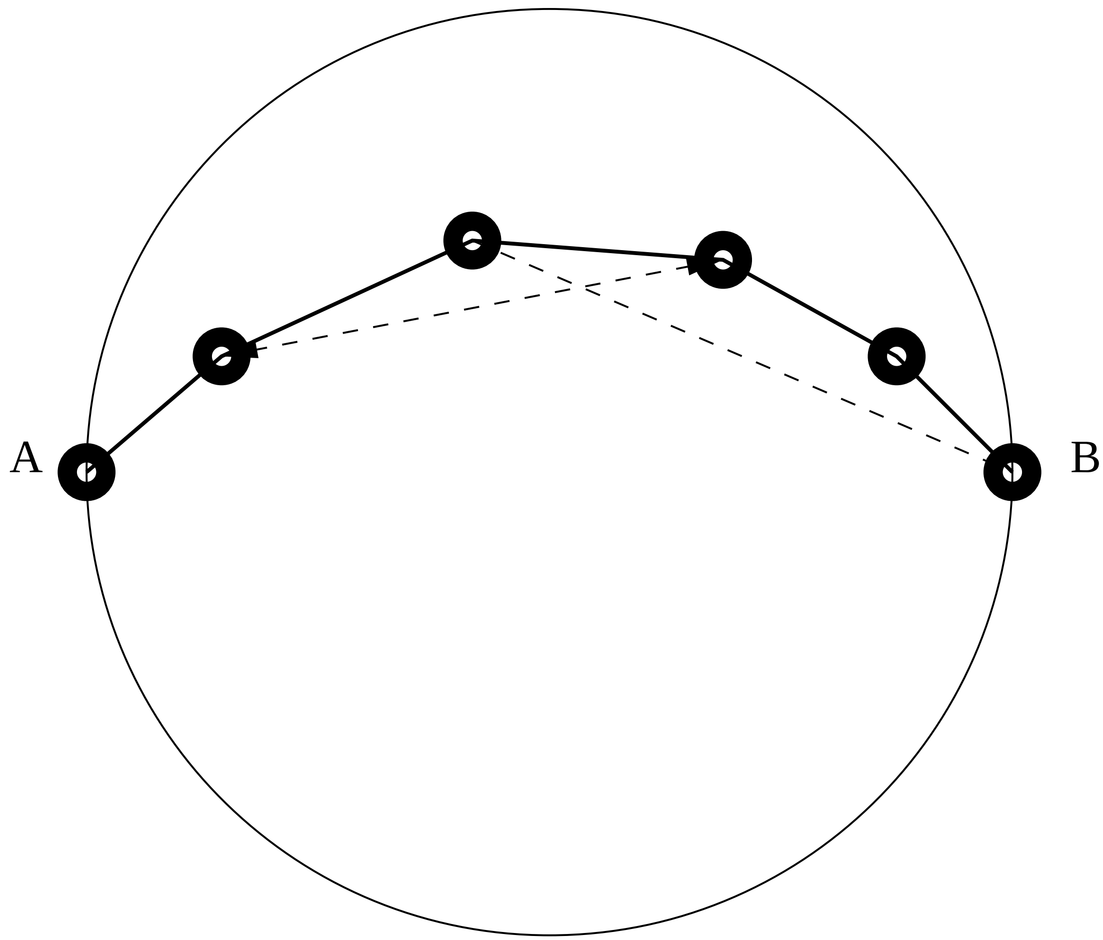
Applying the Gabriel Test



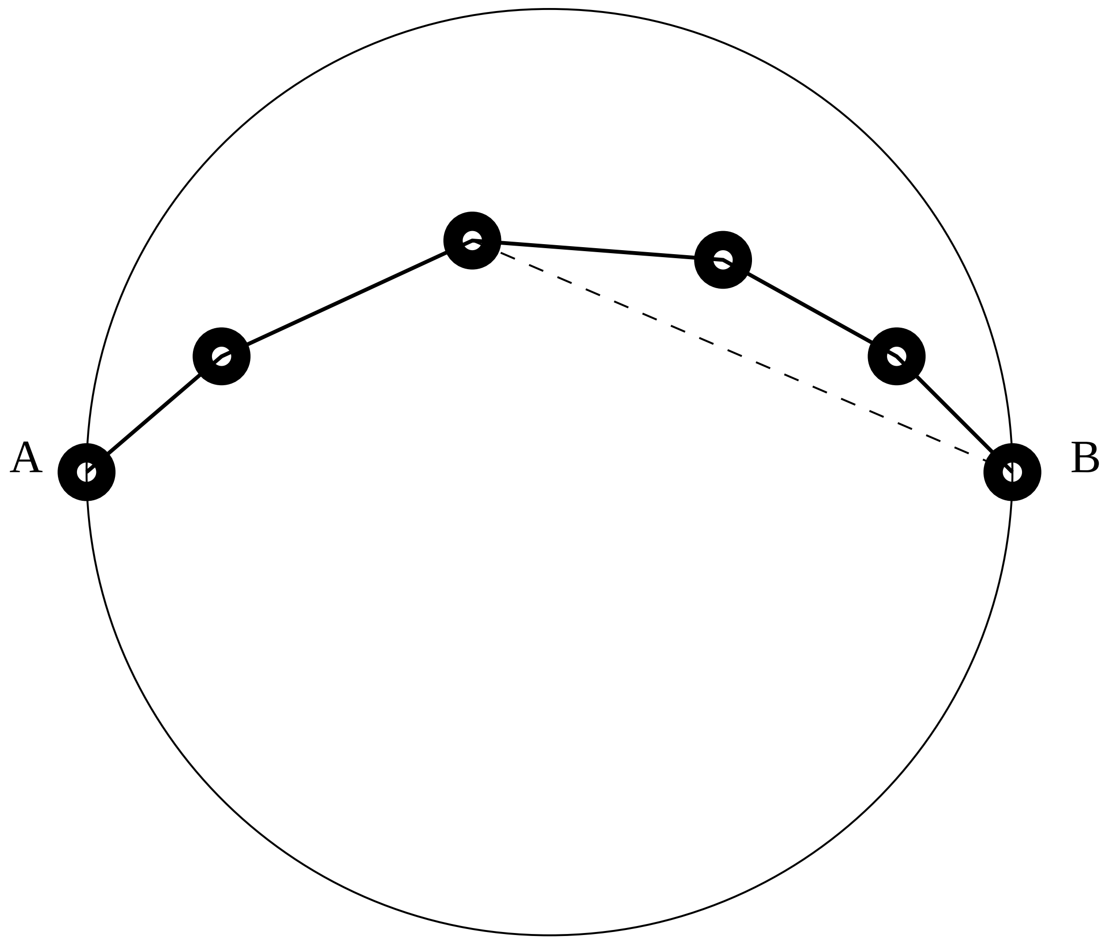
Applying the Gabriel Test



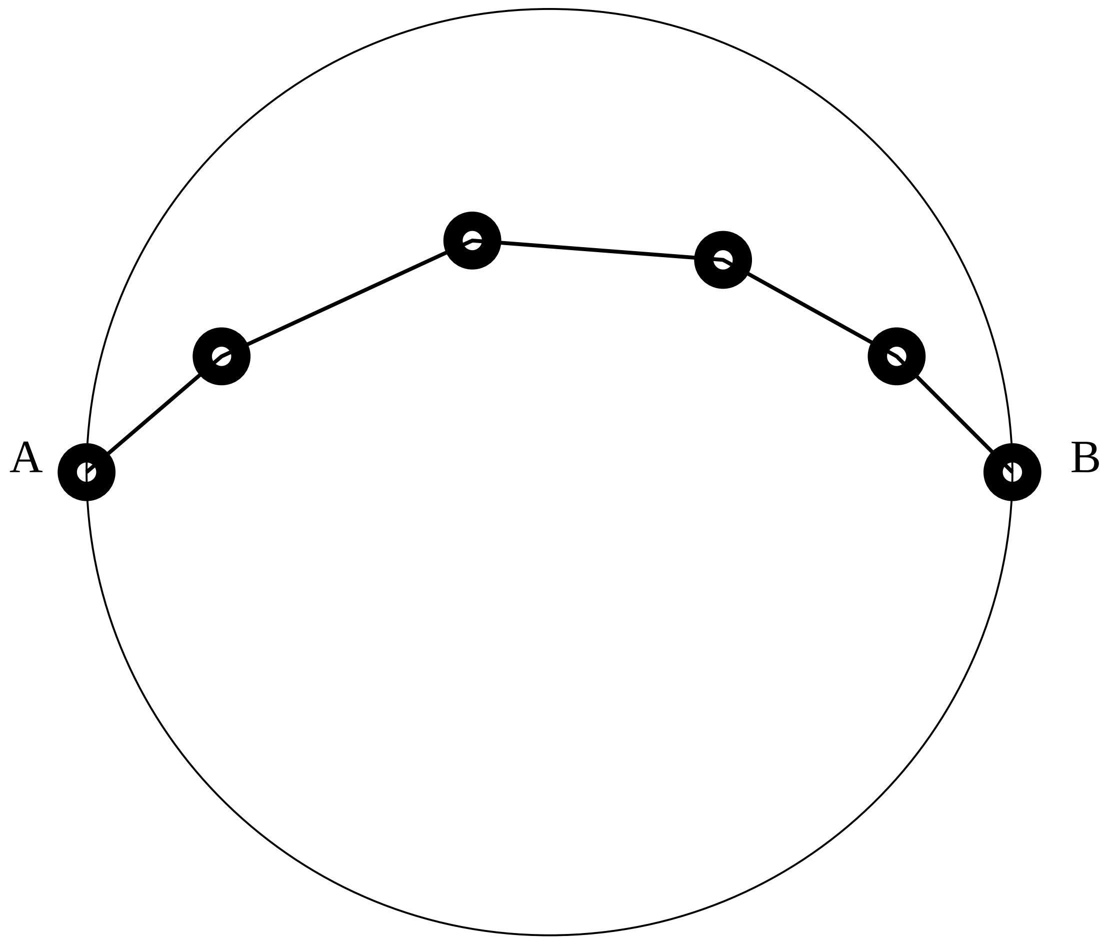
Applying the Gabriel Test



Applying the Gabriel Test



Applying the Gabriel Test



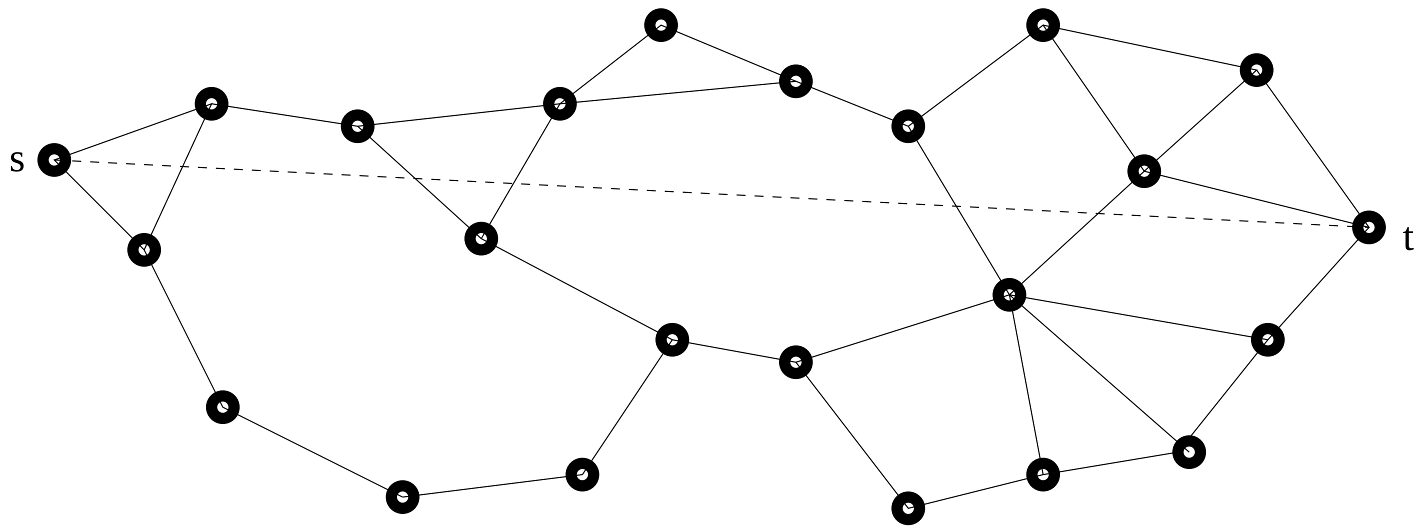
Navigating from source to desination

How does the source discover the coordinates of the destination?

1. Can use the Global Positioning System.
2. It may be given a priori!
   E.g., New cell phones are alredy GPS enabled.
3. It may be given as an IP address: in this case one must use a search algorithm to associate the geographic location to the IP address.
   E.g., "Tell me the address of the Tratoria Vitoria Italian Restaurant in Ottawa".

Back to Navigation

After applying the Gabriel test we have a planar graph.



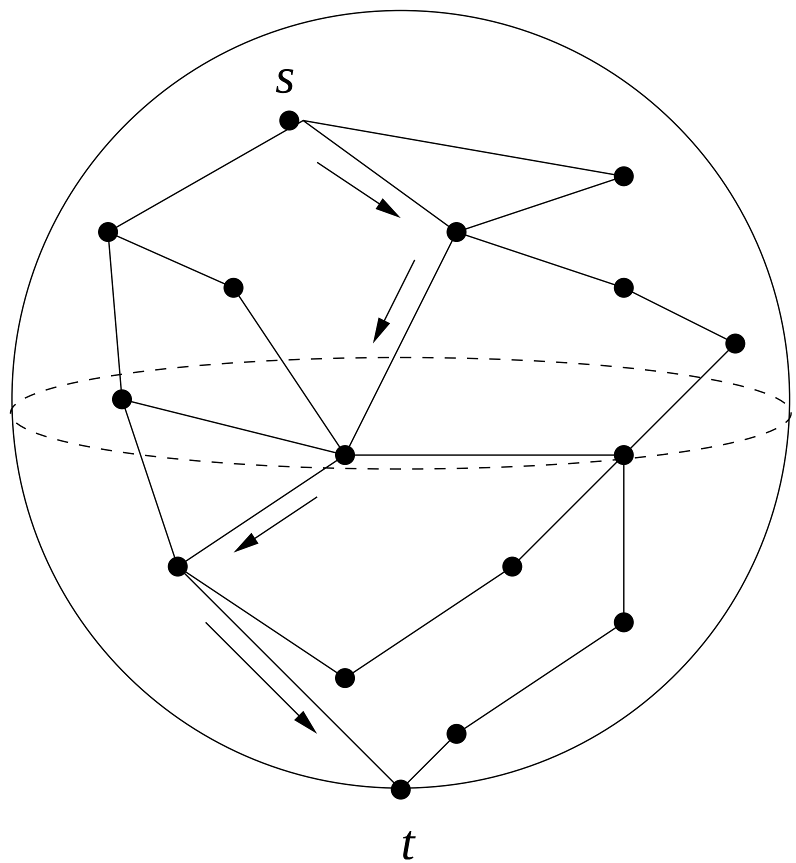
Using GPS we can find out the (x, y) coordinates of s and t.

Hence, we can compute the slope of the line $\vec{st}$.

But how do we use this information in order to discover a route?

Gravitation Routing

Project the planar graph onto a sphere and place the target node on the south pole. Route by gravitation!



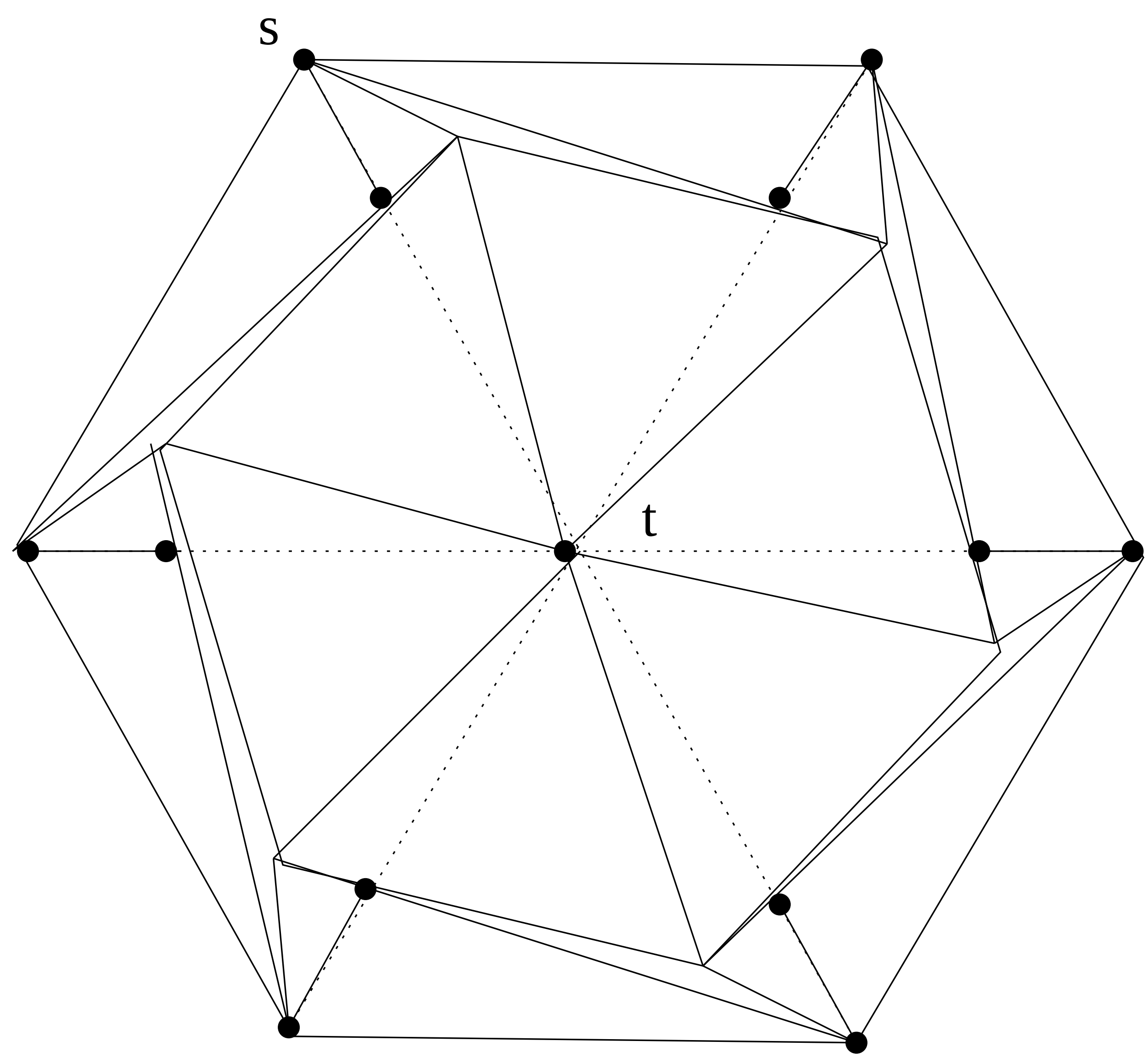
Problem: Must reprocess the whole graph!

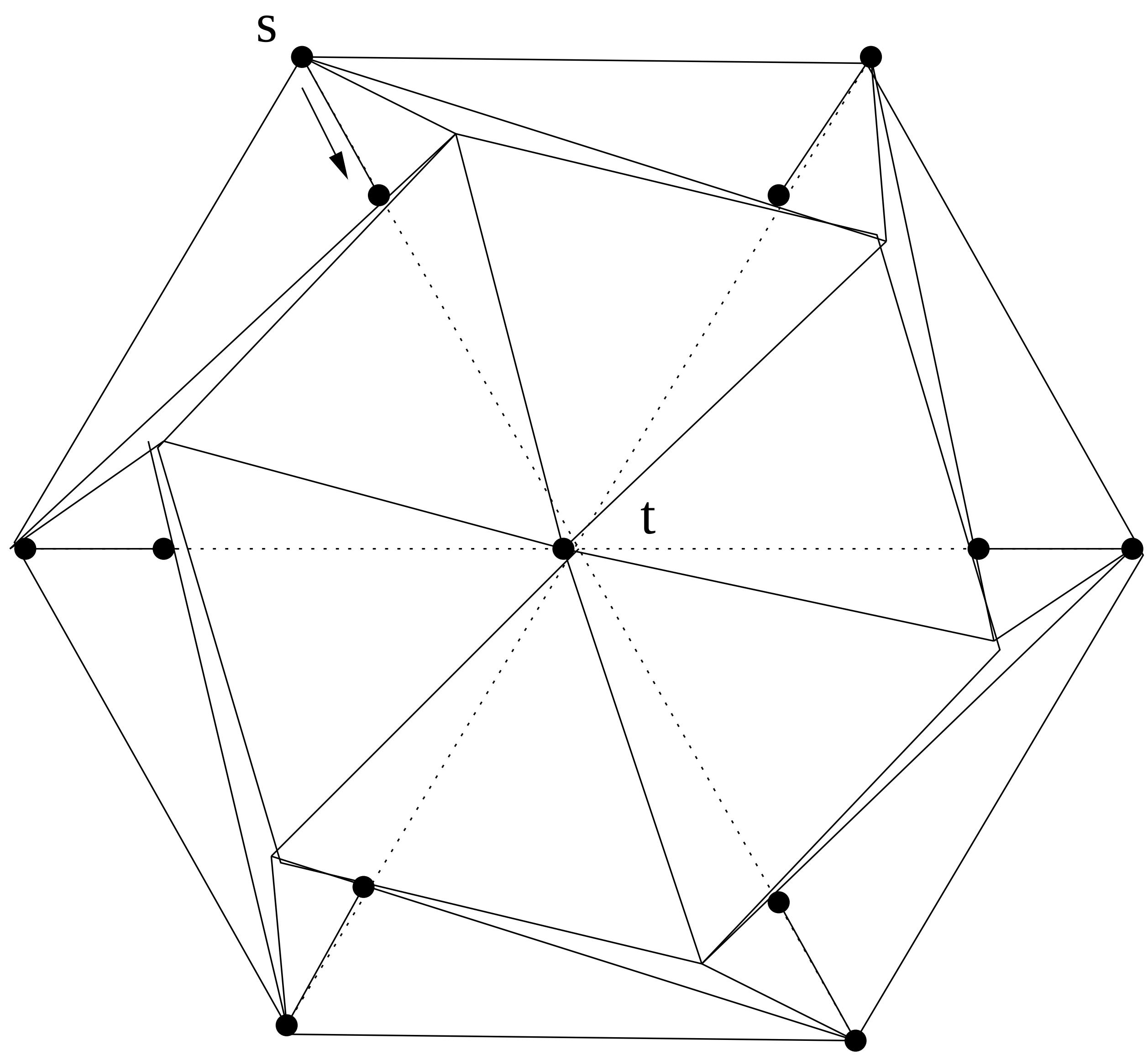
Compass Routing

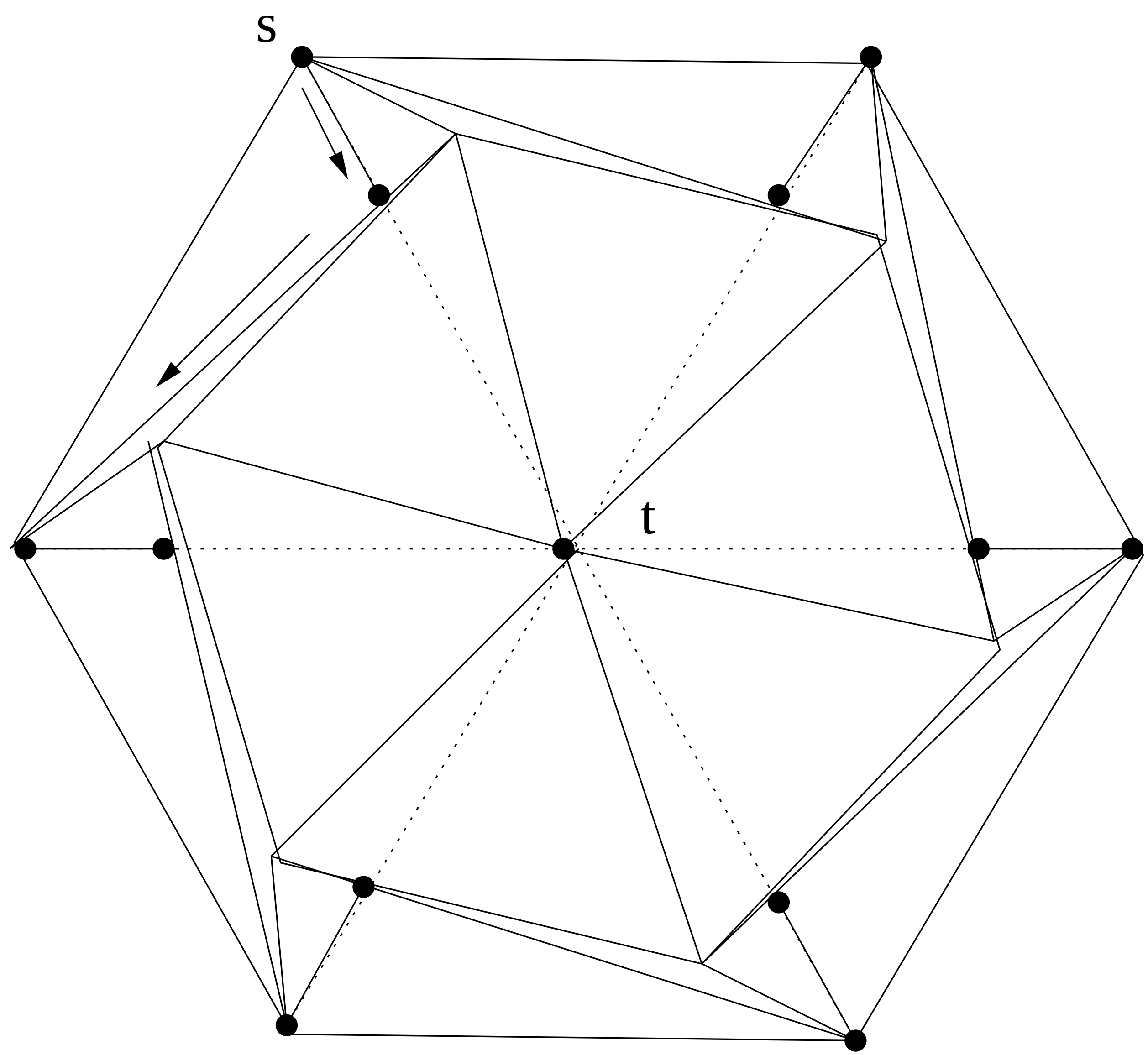
Algorithm

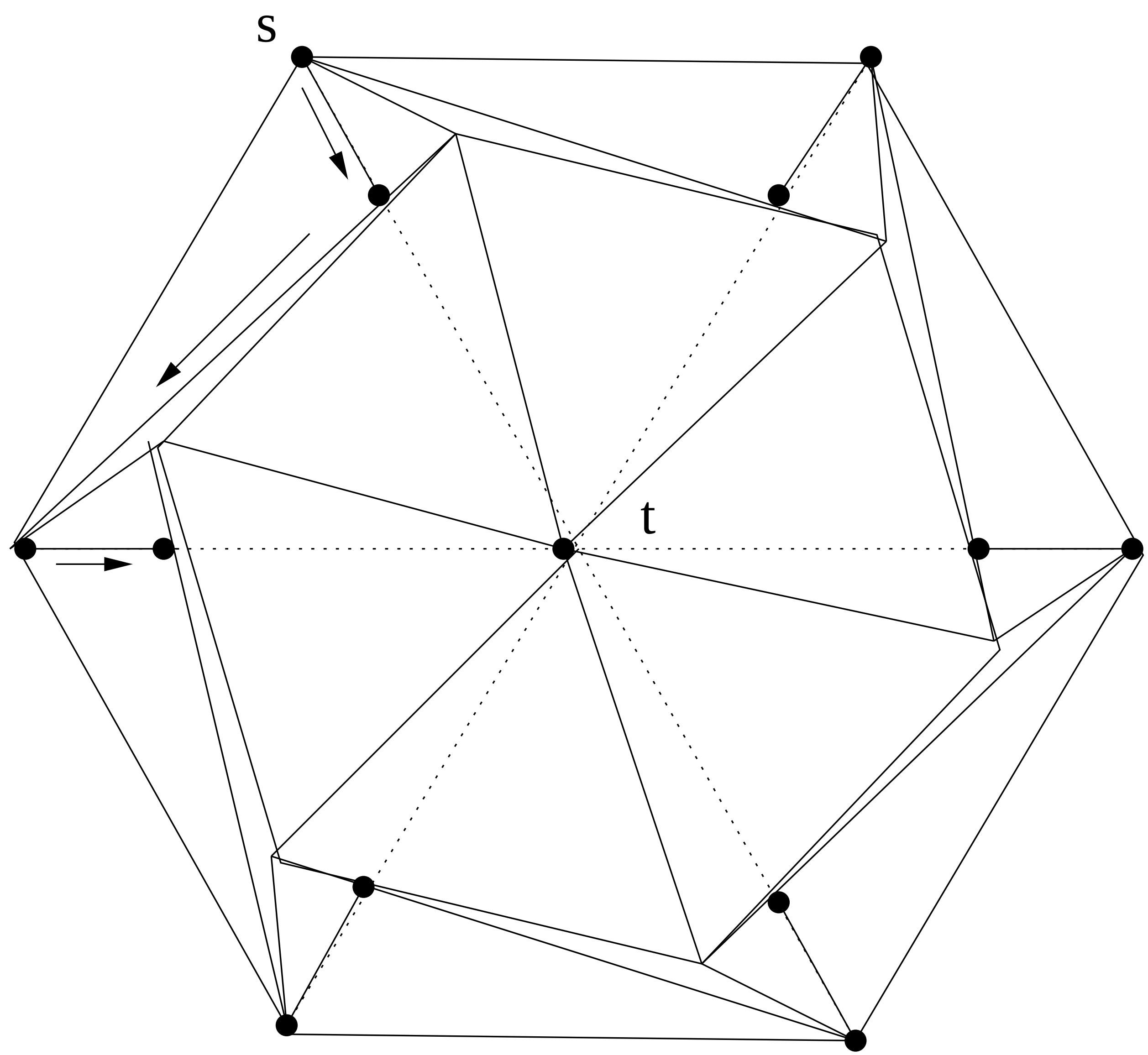
1. Start at source node $c := s$.
2. in a recursive way:
   (a) Choose edge of our geometric graph incident to our current position and with the smallest slope to that of the line $\vec{ct}$.
   (b) Traverse the chosen edge.
   (c) Go back to (a) and repeat until target t is found

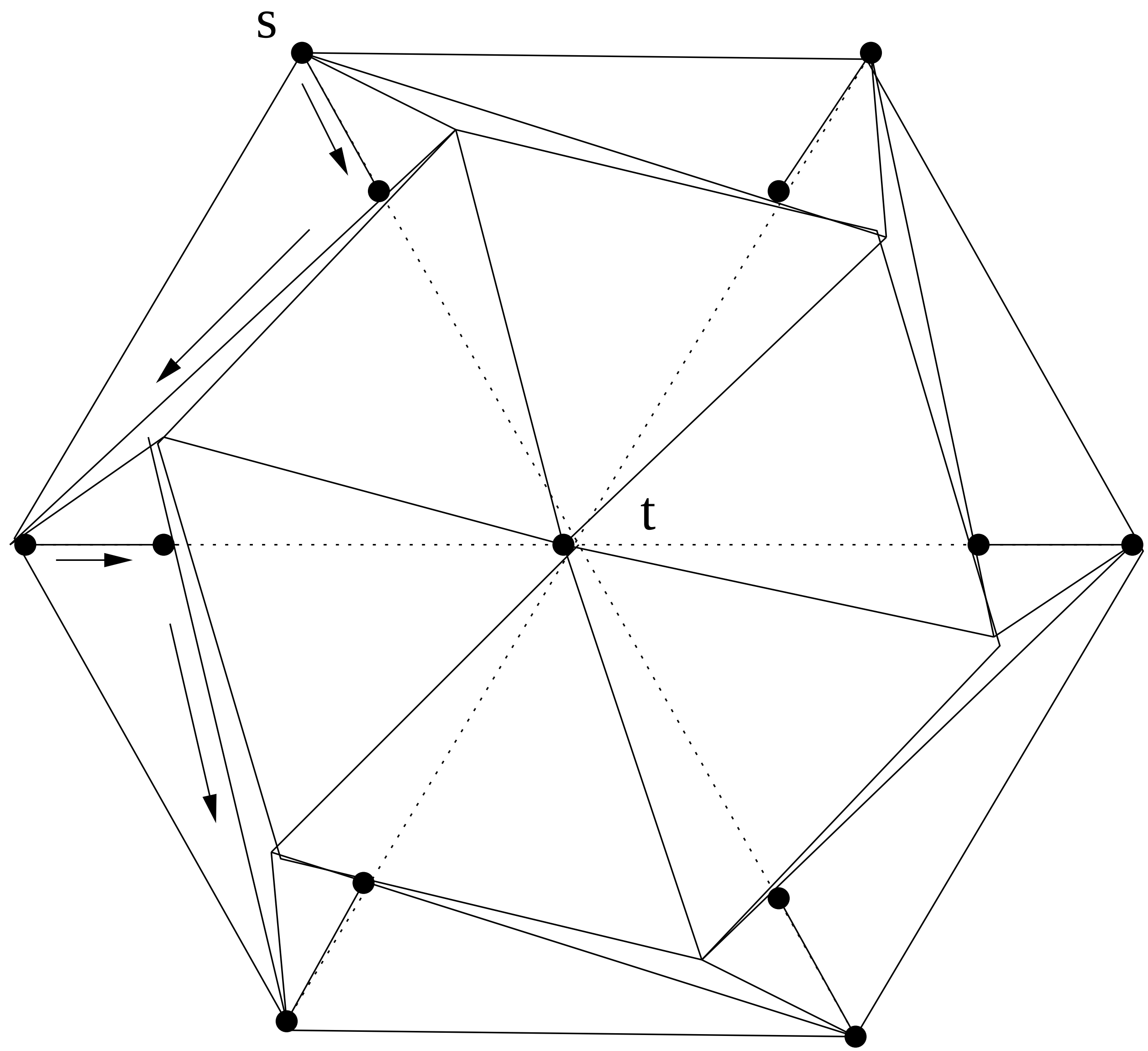
Problem: Compass routing can fail to reach destination!

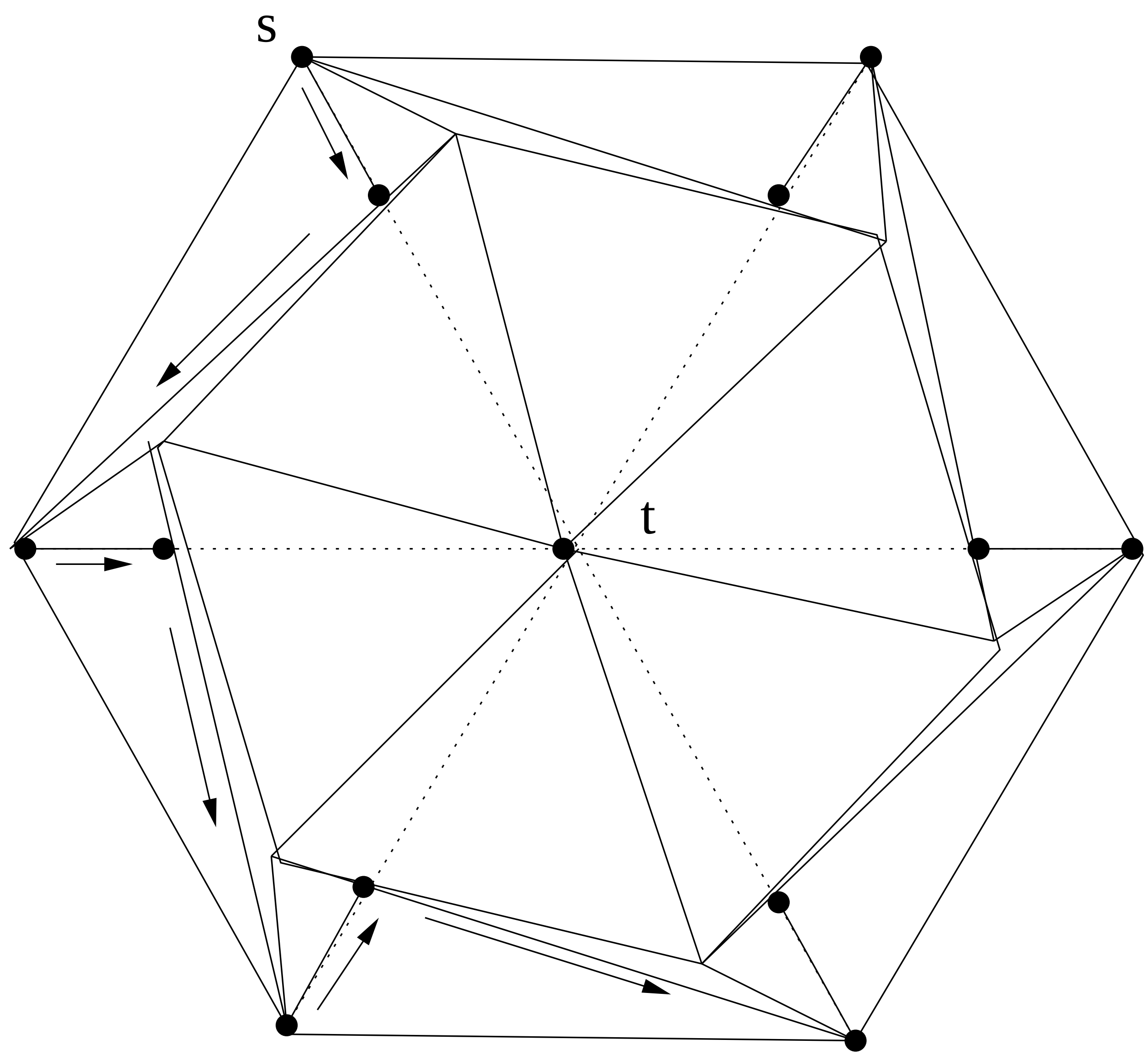

s
t

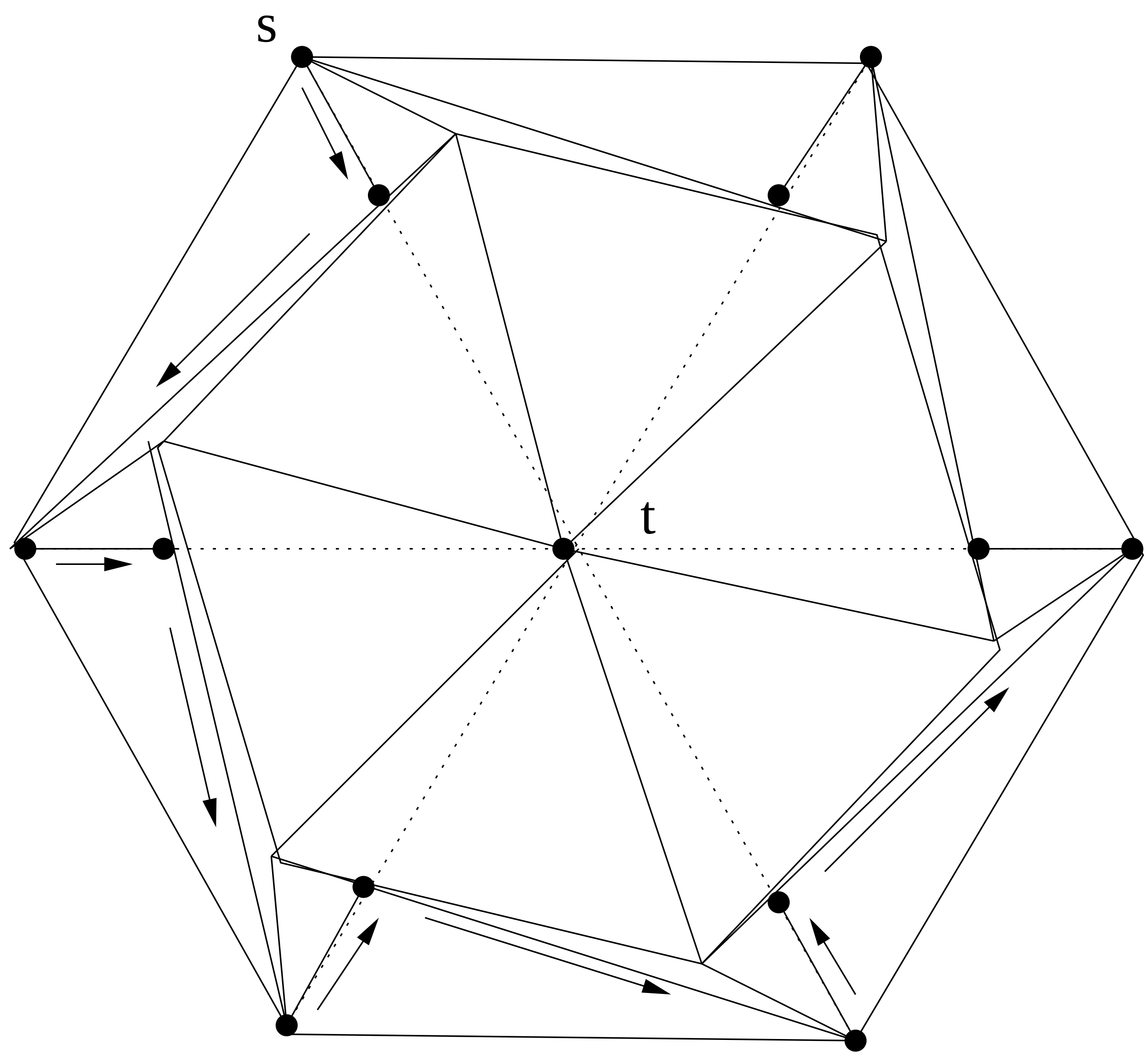

s
t

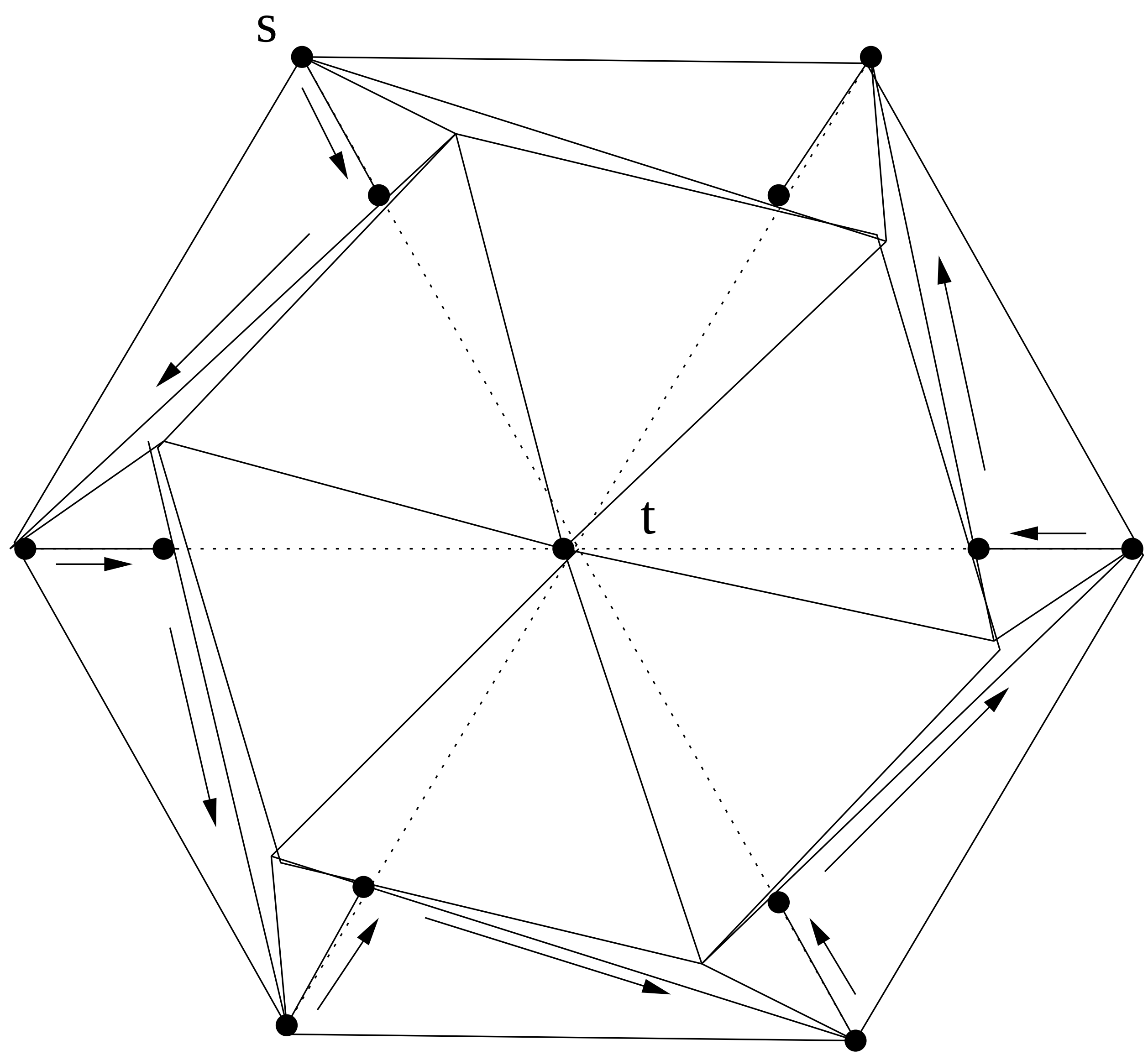

s
t

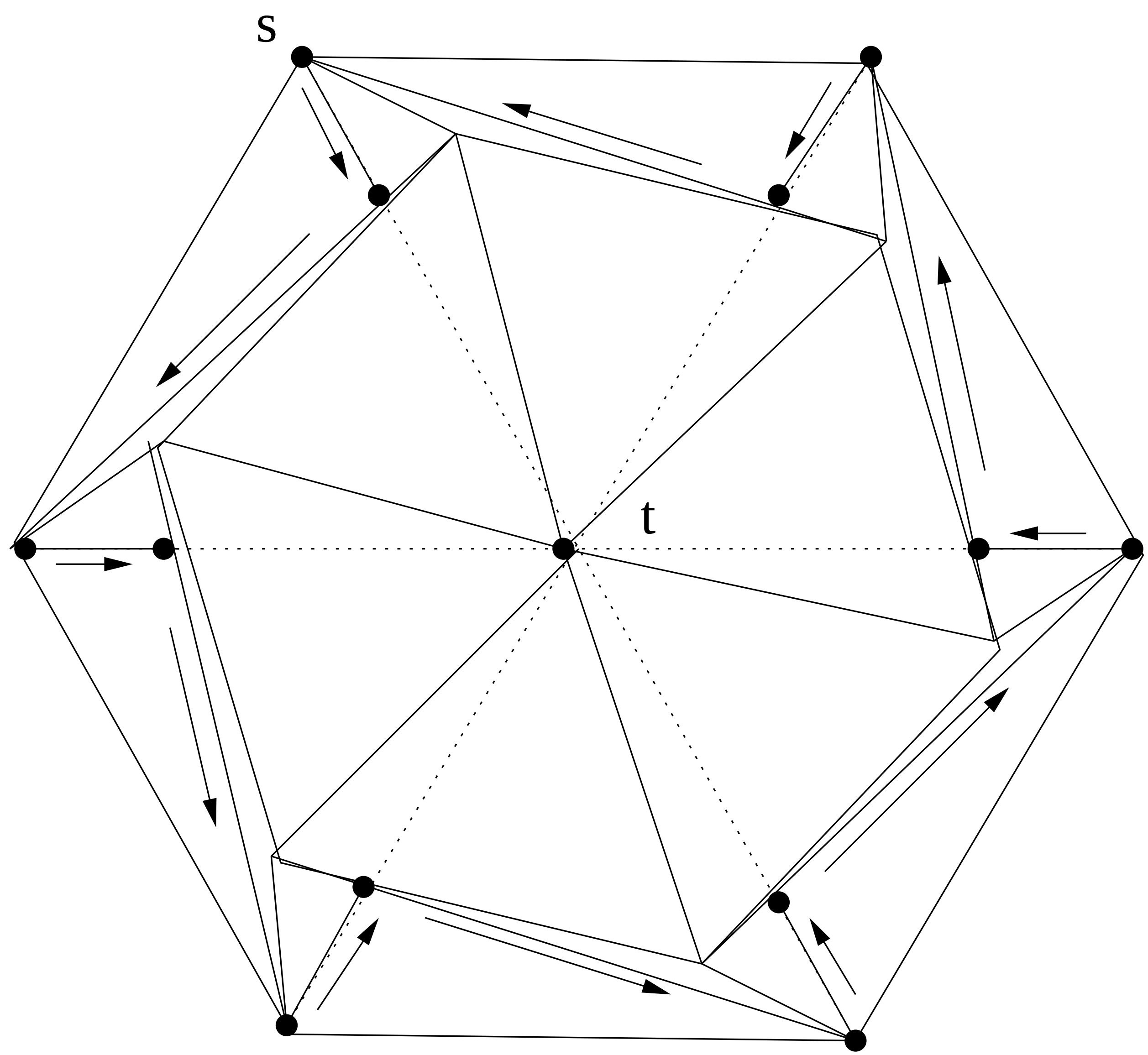

s
t


s
t


s
t


s
t


s
t

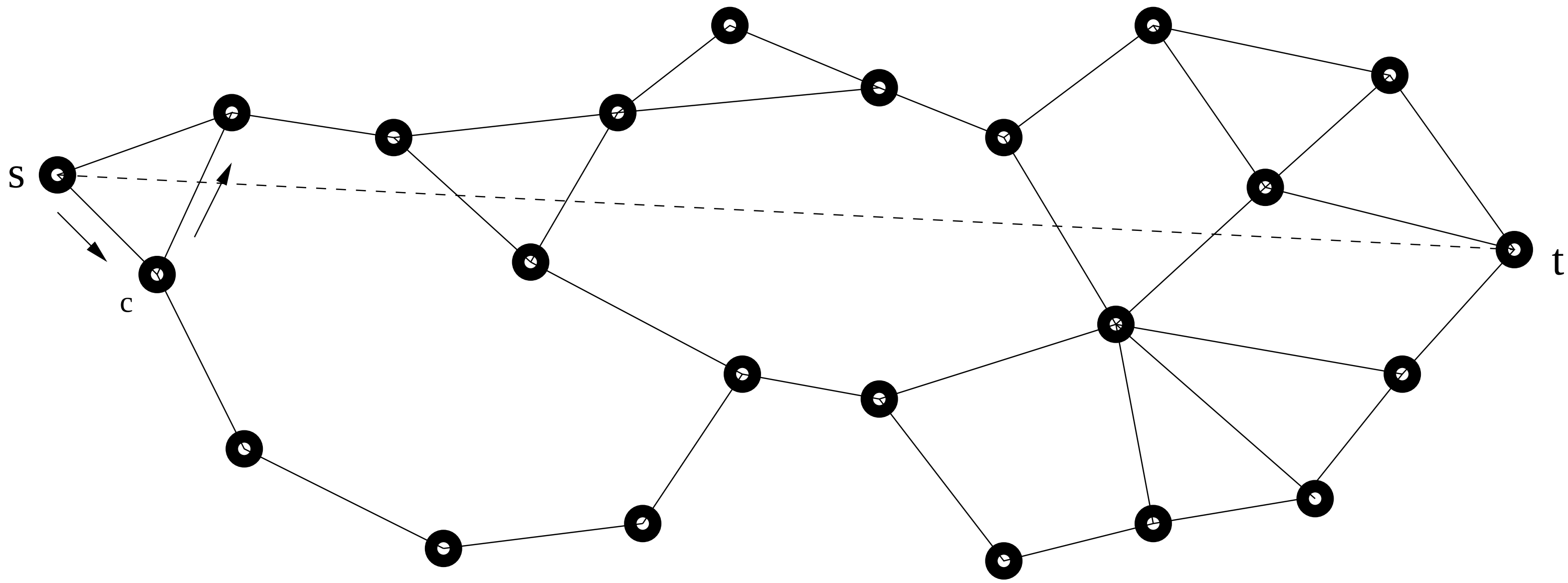

s
t

Face-Routing

Face-Routing Algorithm.

1. Starting at $c := s$ determine face $F := F_0$ incident to c intersected by the line segment $\vec{st}$.
2. Select any of the two edges of F_0 incident to c and start traversing the edges of F_0 until we find the second edge, say xy, of F_0 intersected by $\vec{st}$.
3. Update face F to the new face of the graph containing edge uv, and vertex v to either of the vertices x or y.
4. Iterate until t is found.

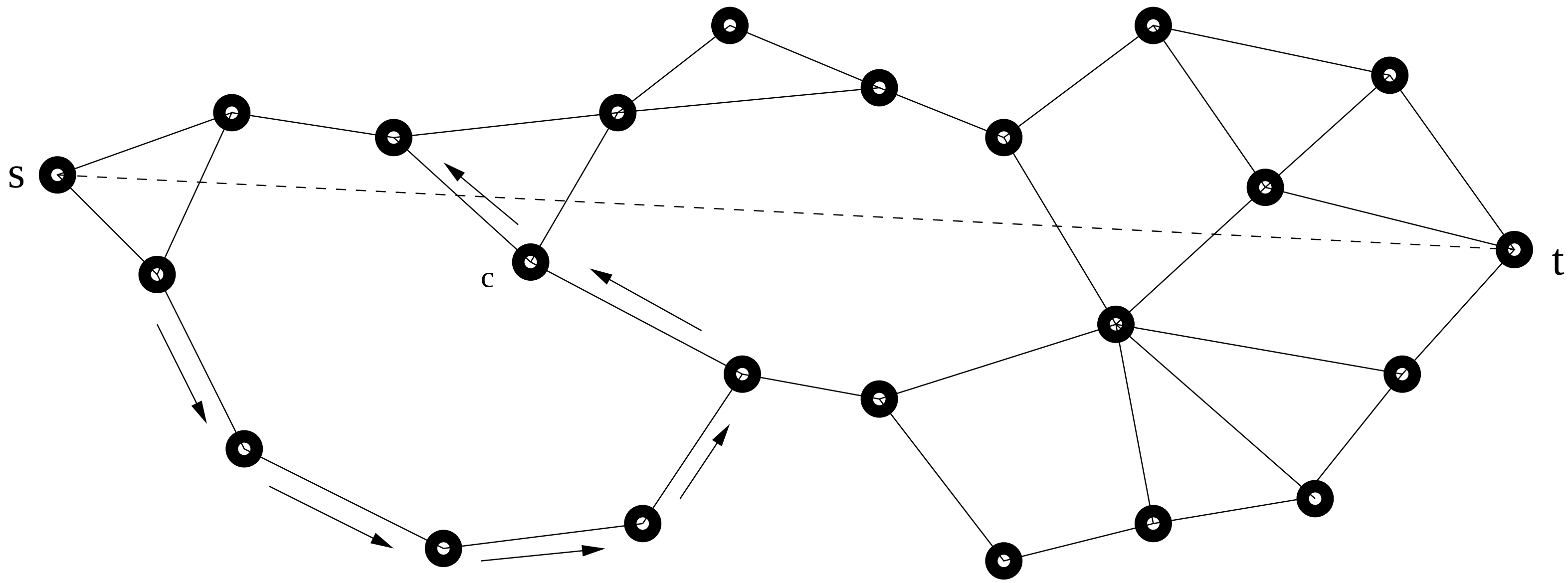
Go from s to t



Initially $c := s$.

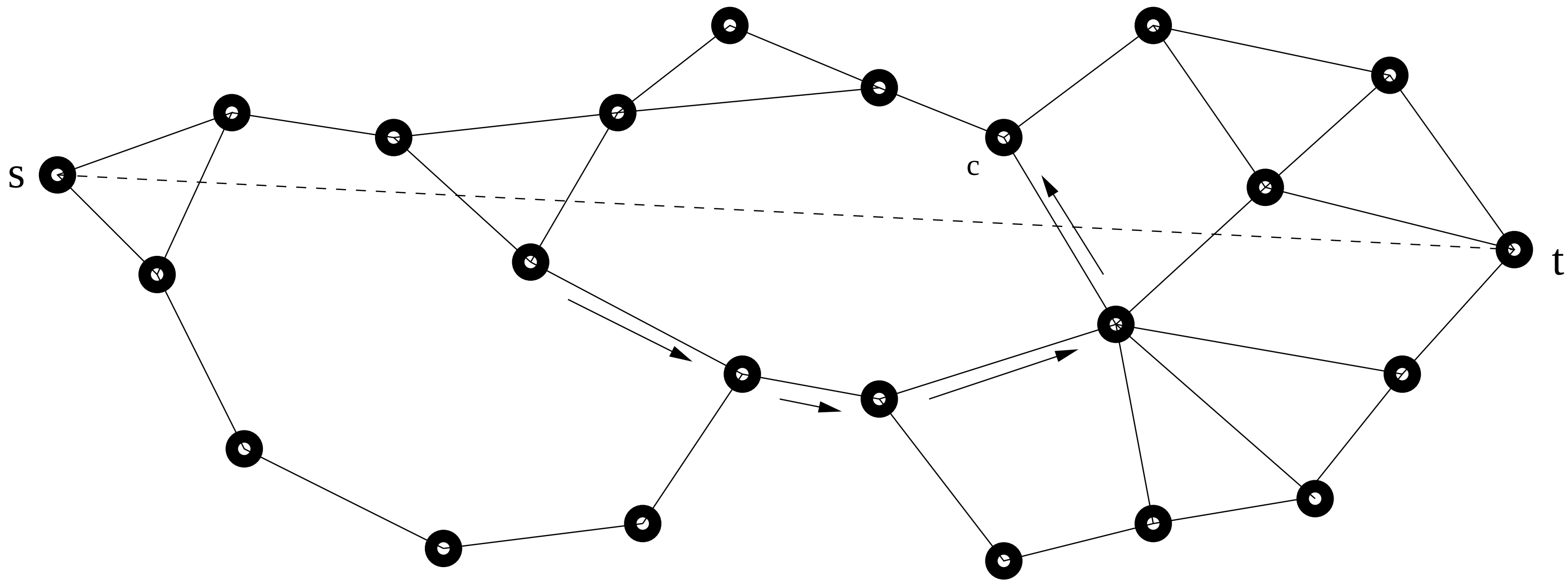
Update c and repeat.

Go from s to t



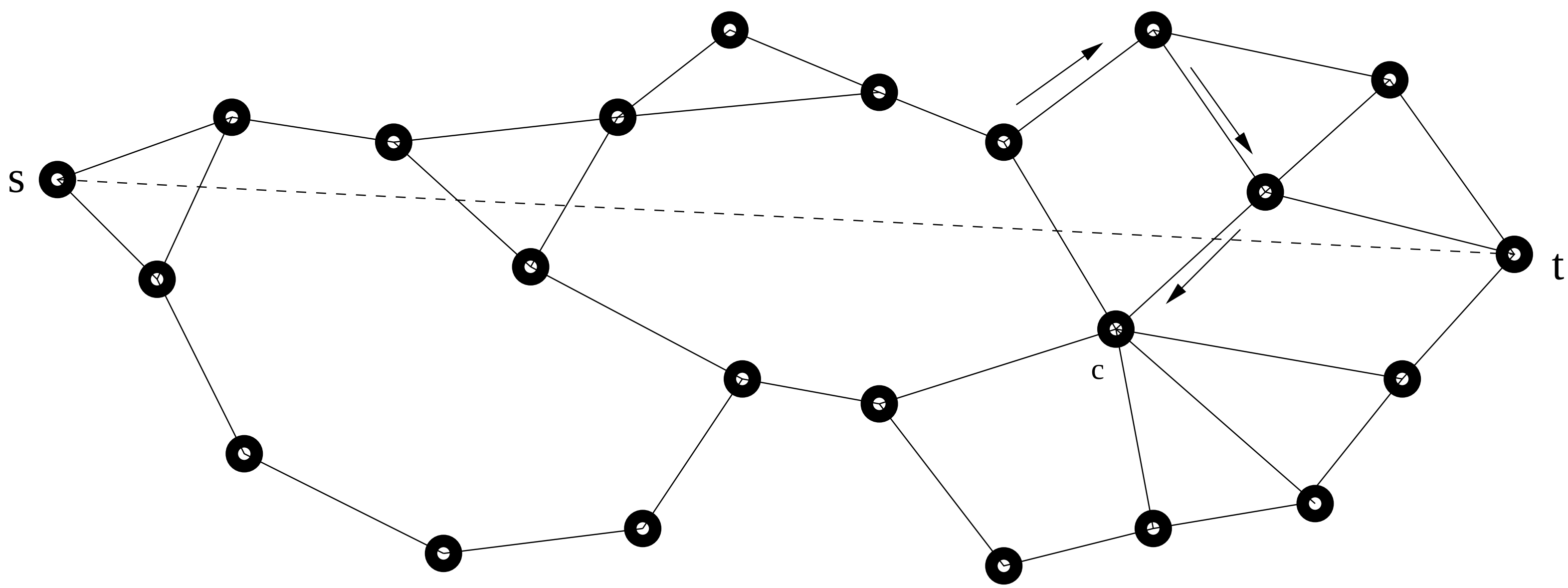
Update c and repeat.

Go from s to t



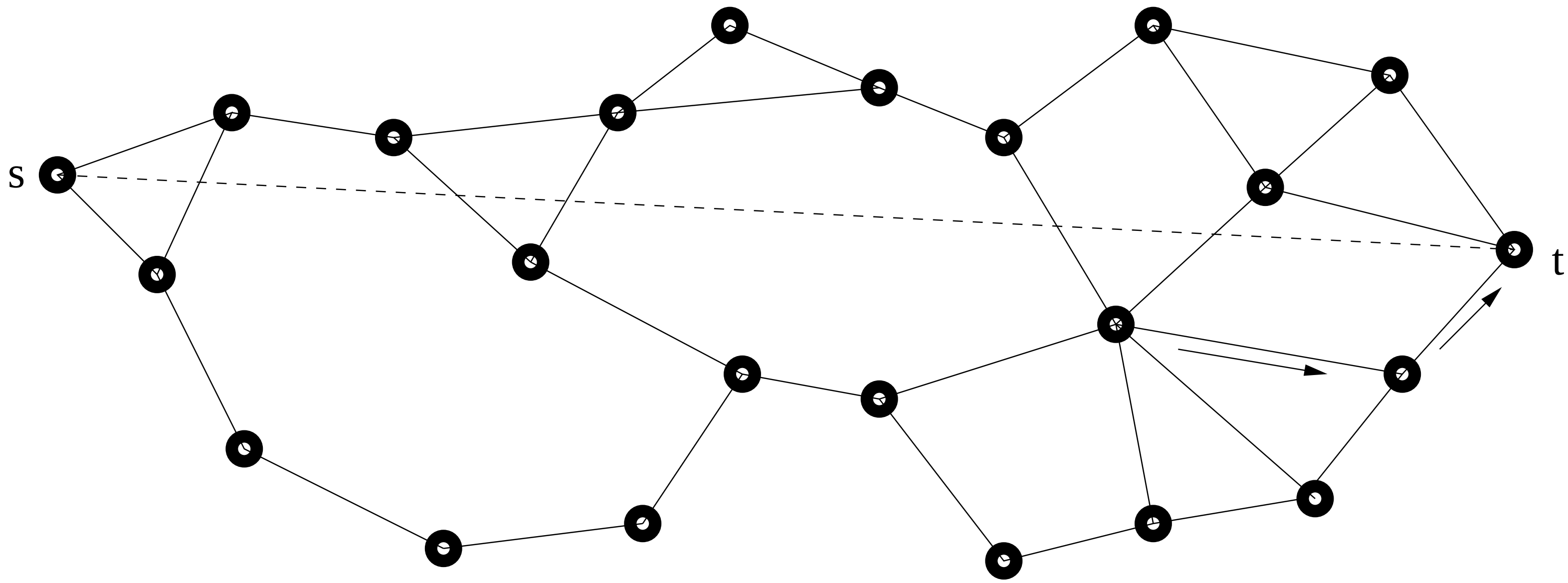
Update c : and repeat.

Go from s to t



Update c : and repeat.

Go from s to t



Now t is found.

Analysis of Face-Routing

- Face routing always advances to a new face. We never traverse the same face twice.
- The distance from the current position c to t gets smaller with each iteration.
- Each link is traversed a constant number of times. Since the graph is planar face routing traverses at most $O(n)$ edges.

Characteristics of Face-Routing

- No indication how long is the Euclidean distance traveled!
- But does it matter? All we wanted was to discover a route!

Conclusions

- You can always find a route in a planar network
  - By using GPS
  - And you need little memory
- Sometimes you can create planarity:
  - By applying "link removal" tests
  - By creating virtual links
  - By increasing power
- The Menace of Non-planarity: Planarity is not necessary but preprocessing increases!
- Can you do the same thing in 3-dimensional space?

More information and papers in my web page:

`http://www.scs.carleton.ca/~kranakis/`