
Routing Paradigms

CS 552

Richard P. Martin

Today's lecture

- Overview of Routing Paradigms
- Original Internet Paper
- Switching (SS7)
- Geometric Routing (TBF)
- Publish/Subscribe (Diffusion)

3 Addressing Strategies

- Where to send data?
 - To a specific computation in the network?
 - To a physical place or along a physical path?
 - To any process “matching” the data?

3 Addressing Paradigms

- What does the network's address space describe?
- Computations in the computer network
 - E.g: process at 128.6.4.4, port 80
- 2D and 3D Space
 - Geometric/position centric routing
 - Line segment, 45° W, 180°N, North, 3Km
- Data
 - Publish/subscribe and diffusion based routing
 - E.g., all nodes wanting data matching /[^]CS552.*/

Addressing & Routing

- Routing layer not necessarily connected to higher-layer's addressing scheme
- Geometric routing used for node-centric addressing.
 - Geographic routing, Integrated geographic forwarding (IGF)
- Publish/subscribe and tuple-spaces run over node-centric routing.
 - Linda, T-spaces.

Cerf & Khan paper

- Describes original thinking behind IP
 - Not called that.
- Goals:
 - Resource sharing across all packet-switched networks
 - Crossing network boundaries
- Means:
 - New protocols:
 - Network protocol
 - Host protocol

Concepts

- Internetwork
 - Network of networks
 - Drives many design decisions
- Gateway
 - Bridges networks
 - Must Understands IP
- Process level communication

Design Choices

- Internetwork limits functionality
 - No fancy flow-control schemes
 - End-to-end flow control, re-transmission, and re-assembly.
- Only gateways and communicating end hosts must learn know protocols
 - Incremental deployment

Concerns

- Different packet sizes
 - Gateways fragment, end hosts assemble
- Transmission failures
- Sequencing
- Flow control
 - End hosts handle
- Process to port mappings
 - End hosts rendezvous using listen and ports

Retrospect

- Fragmentation was not as critical as first thought
- TCP/process communication would have to wait for BSD socket interface
 - 1983
 - Invigorated both IP and Unix communities.
- Hugely successful
 - What made this a success?
 - Does paper follow the “New Jersey” design philosophy?
<http://www.jwz.org/doc/worse-is-better.html>
- What happened to billing and security?

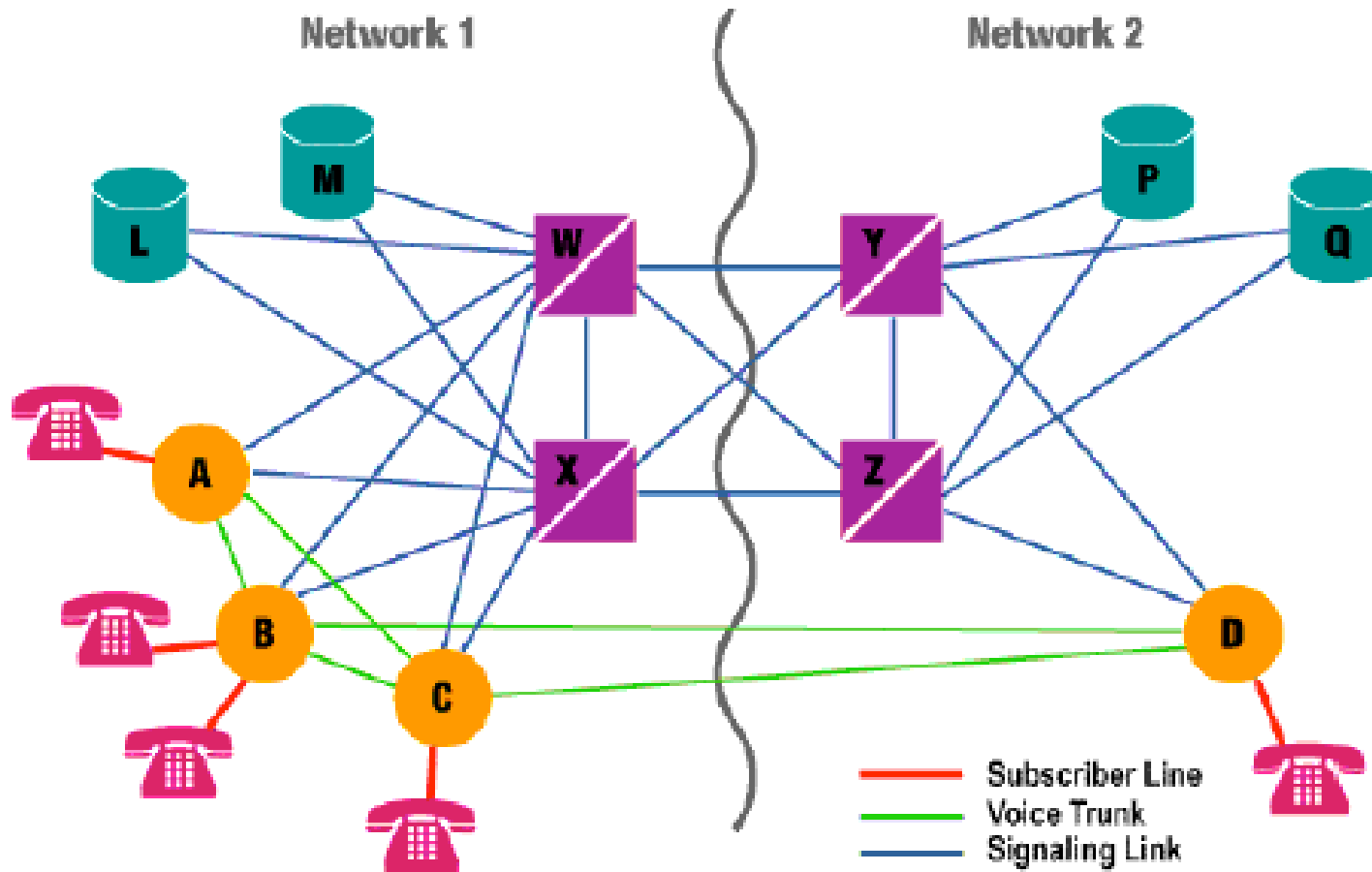
Switching

- Observe totally different way to perform routing (circuit switching) from basic packet switching
- SS7 is the classic PSTN network
 - “Alphabet soup” of networking elements
 - Complex interconnects
 - Devices and links have particular functions

Elements of an SS7 Network

- Nodes:
 - Signaling Switching Point (STP)
 - Signaling transfer point (STP)
 - Signaling control Point (SCP)
- Message types
 - Message signal units (MSU)
 - Link status signal units (LSSU)
 - Fill-in signal units (FISUs)

SS7 Network



Netheads vs. Bellheads

- Terminology: Wired article
 - <http://www.wired.com/wired/archive/4.10/atm.html>
- Different goals
 - Unified network vs. internetwork
- Separate node types
 - Vs. only gateways and hosts
- Separate link types
 - Switching, trunk,
 - Vs. All links “uniform”
- Pairwise reliability of elements and links
 - Vs. reliability only via redundant paths
- Databases provided for lookups as part of network
 - Vs. no DB needed, all DBs external to network

Retrospect

- Hard to have everything in one network
 - Billing, security, reliability: need DBs!
 - Simple data transport, flat network elements
- Reality is that IP runs on top of telecom networks
 - Network of networks - wasn't this how it was supposed to work?

Problems with traditional routing

- Properties of embedded sensor networks
 - Wireless -> mobile nodes, lots of updates
 - Dense -> High volumes
 - Battery power -> can't tolerate a lot of traffic
 - Low duty cycle -> missed updates
- Under these assumptions, TBF is an elegant way to handle many of these issues.

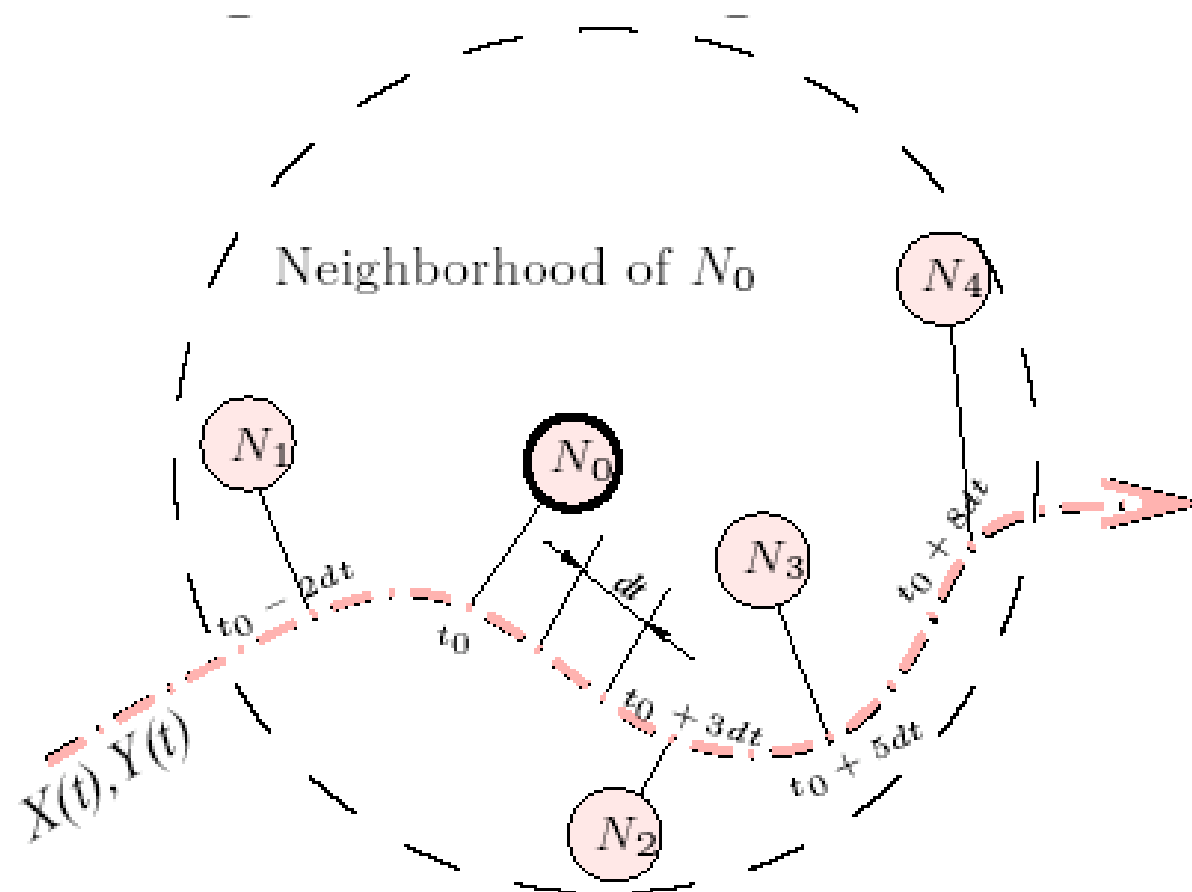
Geometric addressing and routing

- Why send data to a specific node (machine, unit, process).
- Instead, describe data flow in physical space.
 - Nodes along the space will get the data
 - Generalization allow many ways to describe data-flow:
 - Lines, circles, honeycomb
- Advantages:
 - Source based, no routing tables
 - Robust to mobility, node failure,
 - Easy to specify multi-path constructs.

Encoding

- Use parametric encoding:
 - $x=X(t)$, $y=Y(t)$
- Variable t describes packet “progress”
 - Time, hop count, distance
- How to describe in packet:
- Type of object + parameters
 - Line, circle, hexagon.
- Reverse polish notation equation of $X(t)$, $Y(t)$ and t in packet itself.

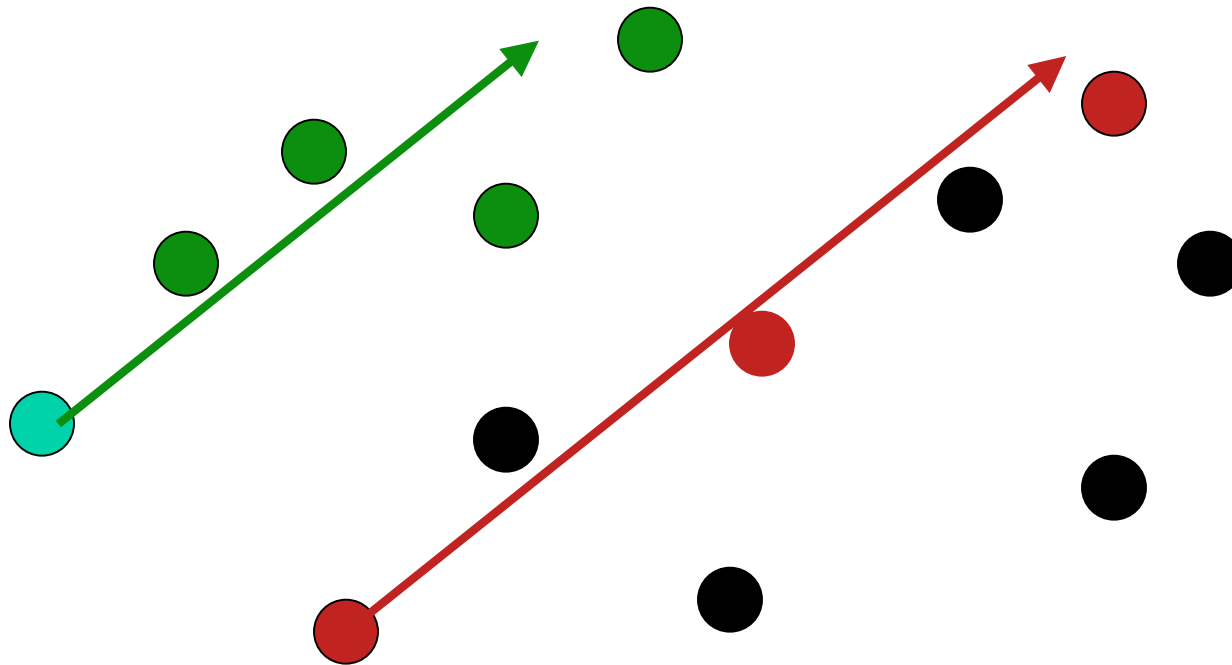
TBF



Linear Example

Trajectory 1: $x(t) = t$, $y(t) = 2t + 1$

Trajectory 2: $x(t) = 2t$, $y(t) = 4t + 1$



Boomerang (circle)

- Circle with radius 2, clockwise:

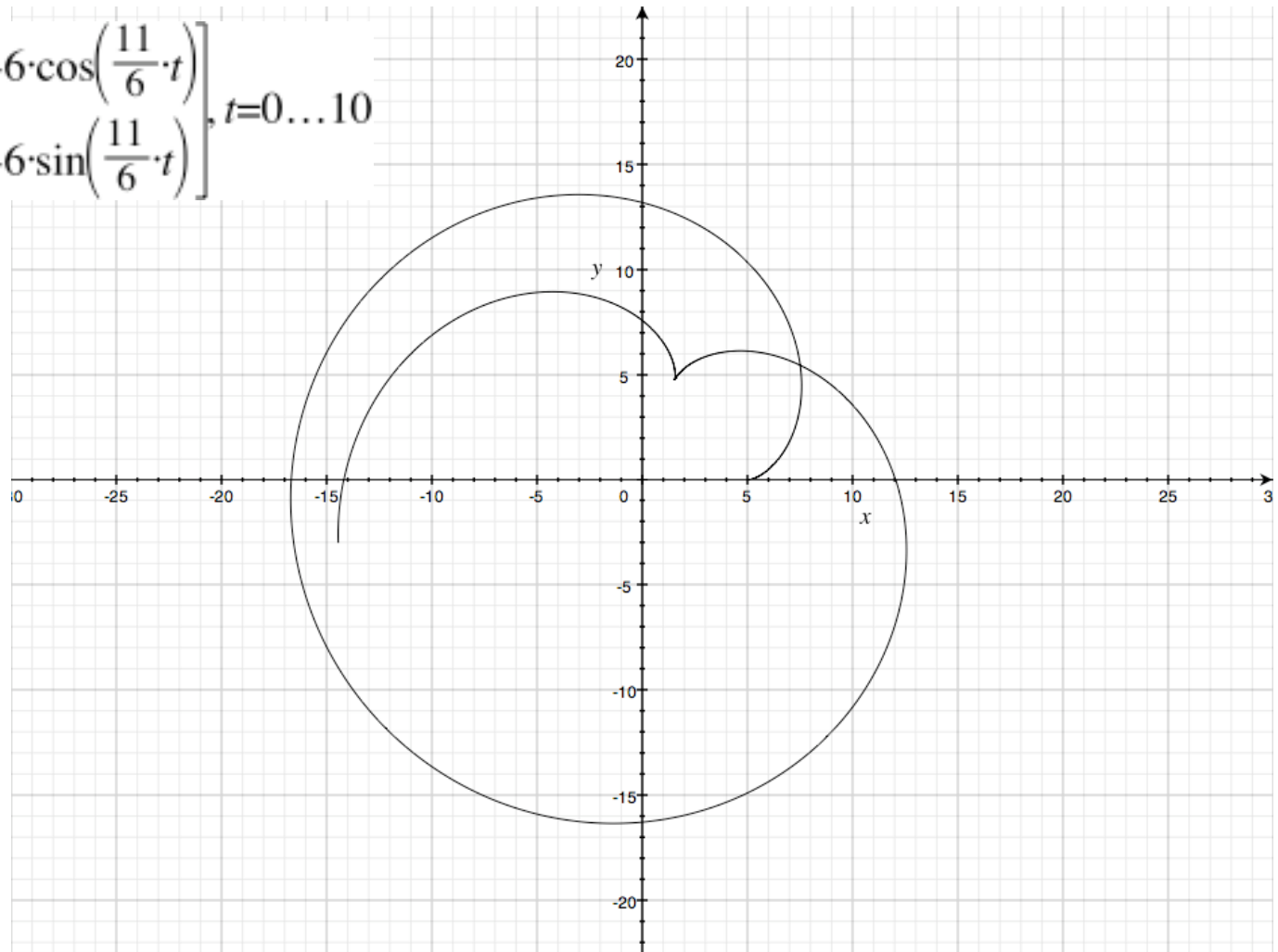
$$x(t) = 2\cos(t), \quad y = 2\sin(t)$$

- Counterclockwise:

$$x(t) = 2\cos(-1*t), \quad y = 2\sin(-1*t)$$

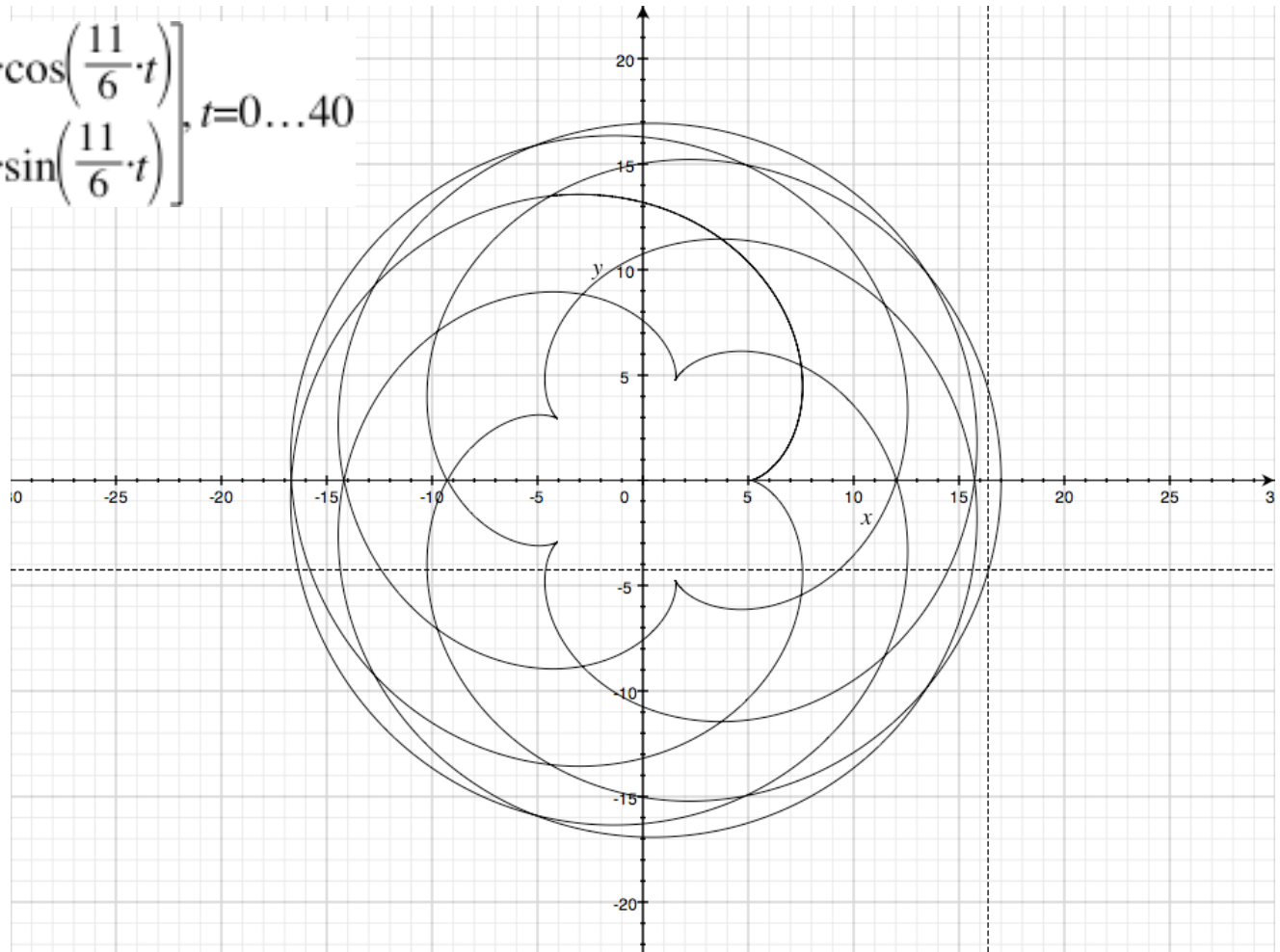
Planar covering example (10 hops)

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 11 \cdot \cos(t) - 6 \cdot \cos\left(\frac{11}{6} \cdot t\right) \\ 11 \cdot \sin(t) - 6 \cdot \sin\left(\frac{11}{6} \cdot t\right) \end{bmatrix}, t=0 \dots 10$$



Planar covering example (40 hops)

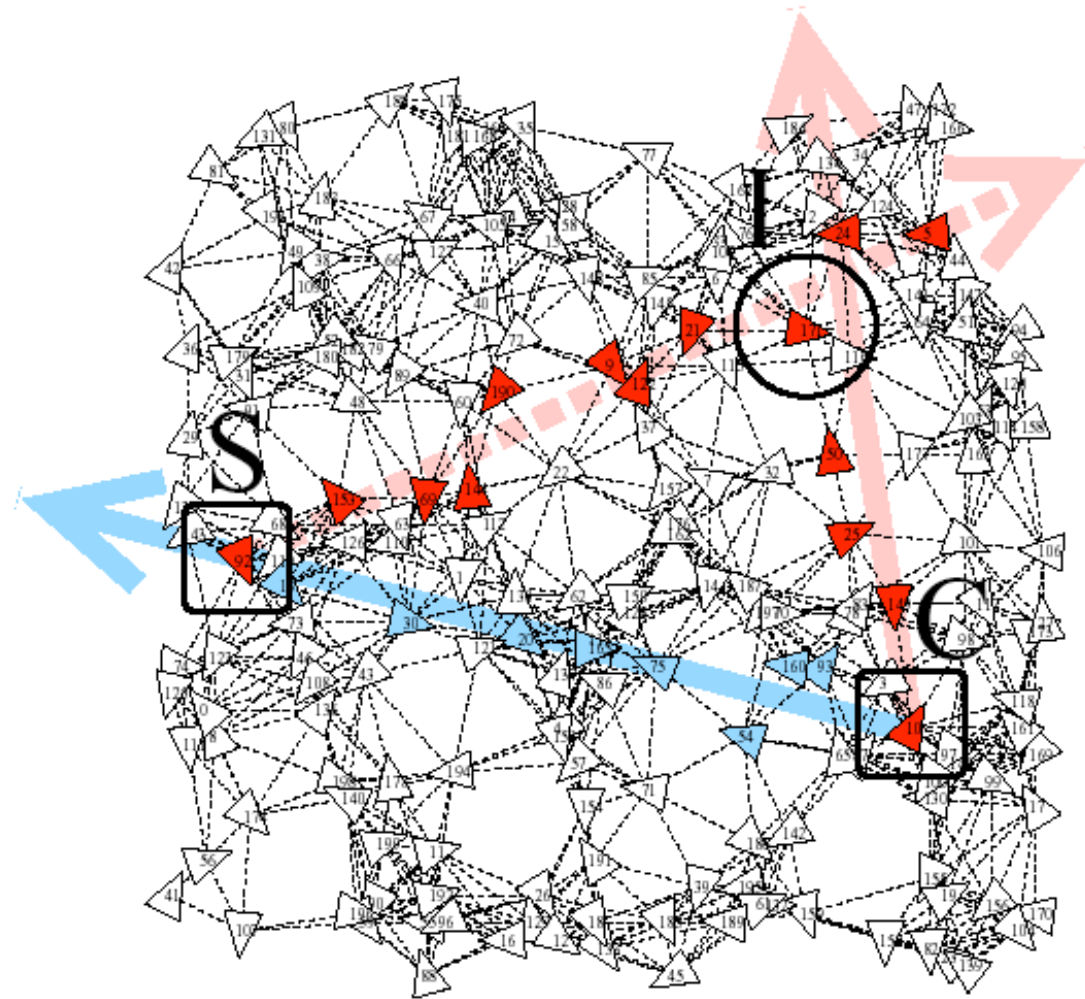
$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 11 \cdot \cos(t) - 6 \cdot \cos\left(\frac{11}{6} \cdot t\right) \\ 11 \cdot \sin(t) - 6 \cdot \sin\left(\frac{11}{6} \cdot t\right) \end{bmatrix}, t=0 \dots 40$$



Uses

- Discovery (maps to intersection)
- Flooding (maps to covering)
- Multipath routing
- Ad-Hoc routing

Discovery Example



Limitations

- Requires physically dense networks
- Nodes need positioning information
 - Global
 - Local
- How to unify with node-based addressing?
 - What's the best way to perform both?

Data-Centric Routing

- Addresses same problems as TBF
- More directed for sensor networks
 - More like a programming model for sensor networks?

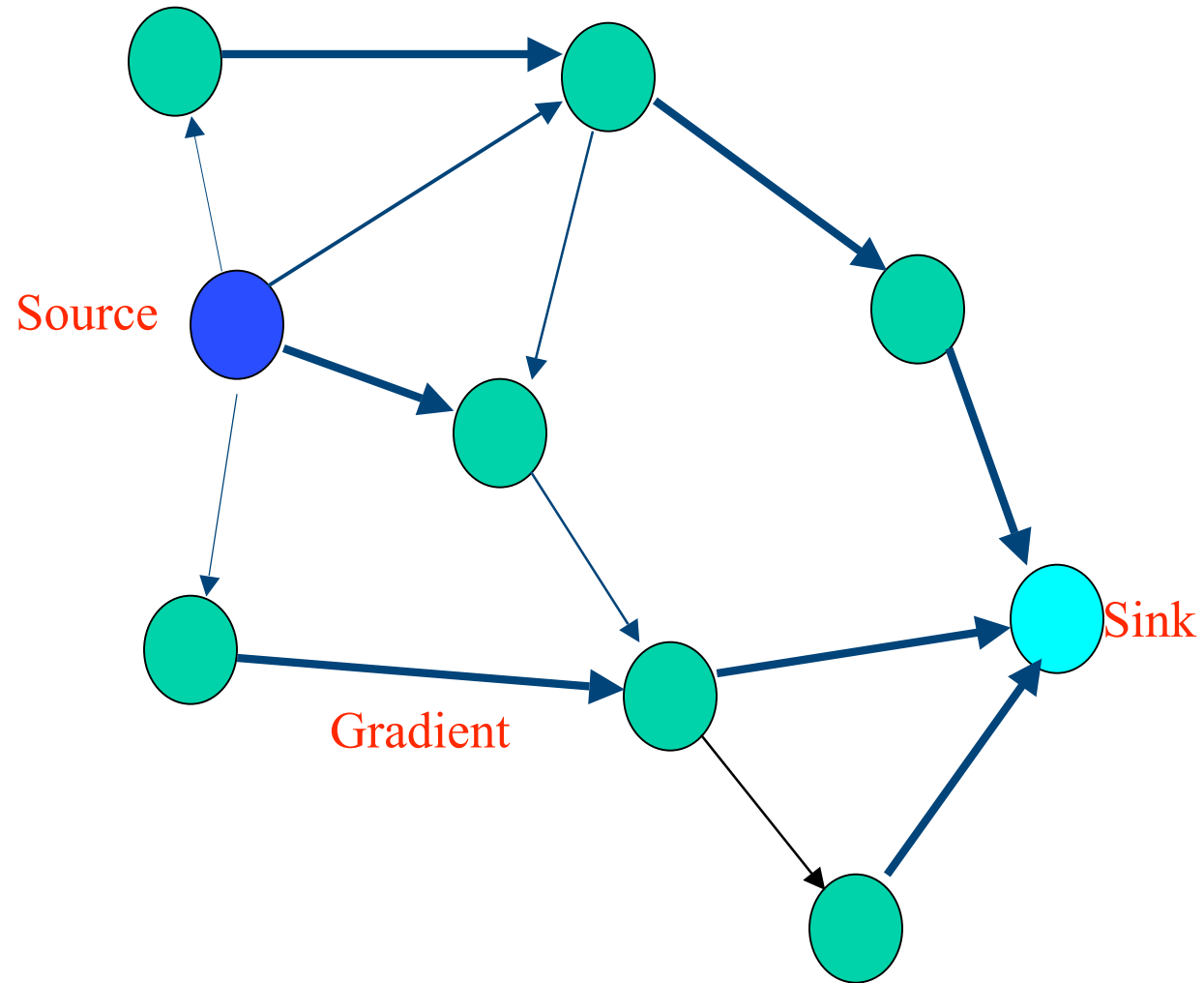
Directed Diffusion

- Sensor node **names** data with attributes
 - This is like a “**publish**”
- Other nodes express **interests** based on these attributes
 - This is like a “**subscribe**”
- Network nodes propagate interests
 - interests establish **gradients** that direct diffusion of data
 - A gradient is a **route** between a publisher and subscriber
- As it propagates, data may be **locally transformed** (e.g. **aggregated**) or **cached** at nodes

Building gradients(routing)

- What are the local rules for propagating interests?
 - flood interest
 - More sophisticated techniques possible: directional interest propagation, based on cached aggregate information
- What are the rules for establishing gradients?
 - In example, highest gradient towards neighbor who first sends interest
 - Others possible e.g., towards neighbor with highest remaining energy

Example



Implicit assumptions

- Not much unicast traffic
 - Valid for sensor networks?
- Gradients/routes are soft state
 - Require continuous reinforcement to maintain
- Gradients/routes can vary
 - E.g. Multipath
- Traffic can be reduced with aggregation

Implementation issues

- Simple implementations possible
 - Flood interest
 - Use backward learning to build gradients
 - Use timers to discard gradients if not refreshed.
- Straightforward to build broadcast, multicast,
- Simple in this case is not efficient.

Limitations

- Efficient naming and interest matching
 - Flooding?. similar problems in any pub/sub network (Tivoli, Linda, T-Spaces)
- If placed in routing layer, how to get efficient node-centric routing?
 - Simple way if first bullet is solved though
- Right layer?
 - Networking vs. application.

Summary

- Switching/Node centric dominate
- Geometric and pub/sub
 - Elegant, but will be widely used?