Chapter 9
DATA GATHERING

Distributed
Computing
Group

Mobile Computing
Winter 2005 / 2006

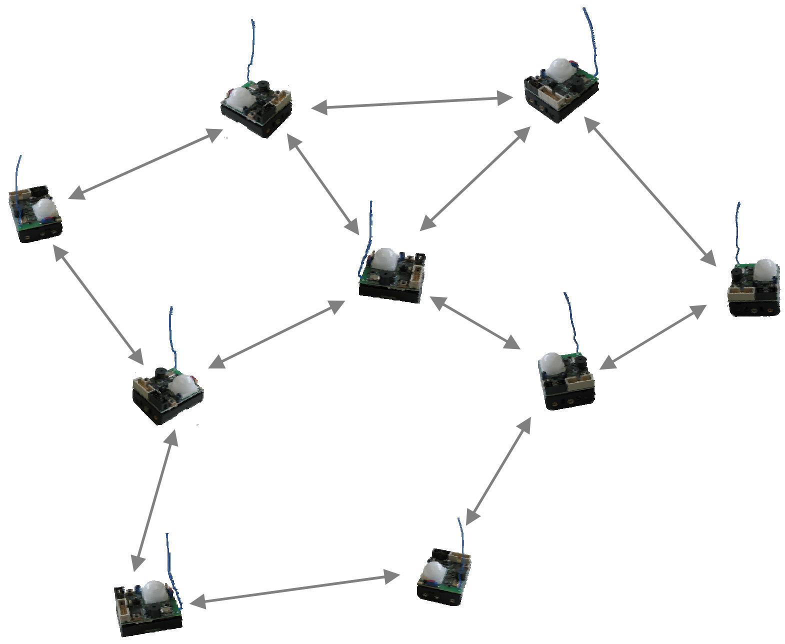
Overview

- Motivation
- Data gathering with coding
  - Self-coding
    - Excursion: Shallow Light Tree
  - Foreign coding
  - Multicoding
- Universal data gathering tree
  - Max, Min, Average, Median, Count Distinct, …
- Energy-efficient broadcasting

Sensor networks

- Sensor nodes
  - Processor & memory
  - Short-range radio
  - Battery powered

- Requirements
  - Monitoring geographic region
  - Unattended operation
  - Long lifetime

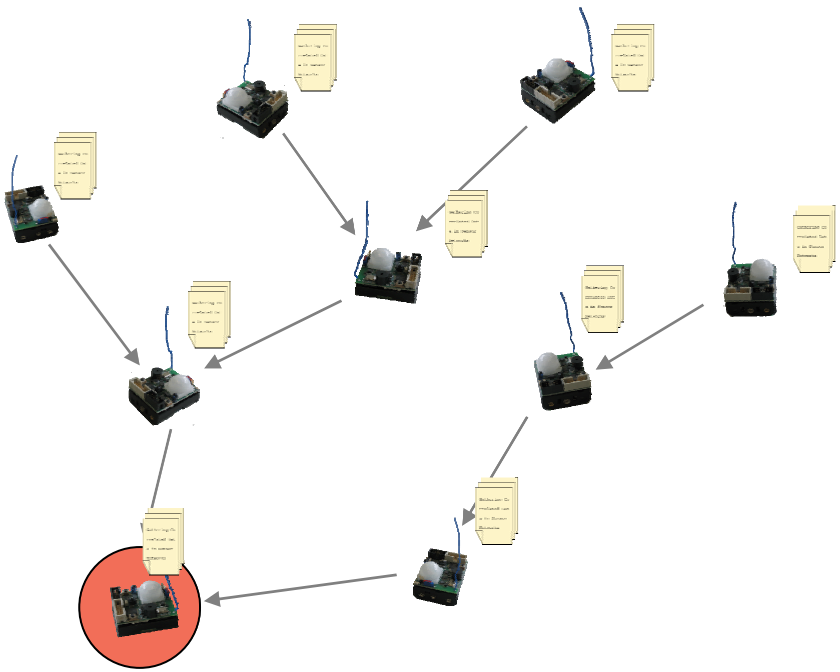


Data gathering

- All nodes produce relevant information about their vicinity periodically.
- Data is conveyed to an information sink for further processing.

➡ Routing scheme

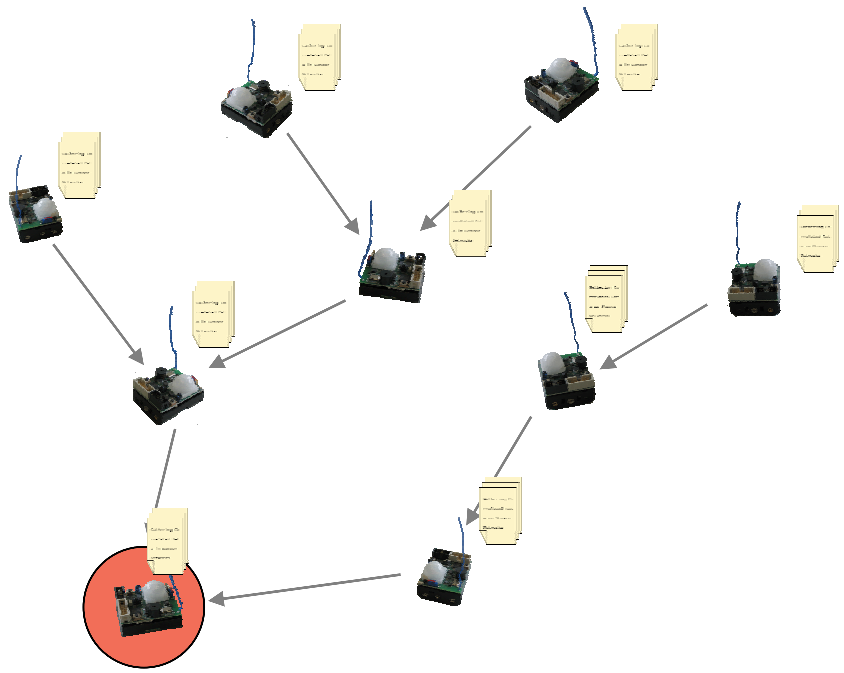
On which path is node u's data forwarded to the sink?



Time coding

- The simplest trick in the book: If the sensed data of a node changes not too often (e.g. temperature), the node only needs to send a new message when its data changes.

- Improvement: Only send change of data, not actual data (similar to video codecs)



More than one sink?

- Use the anycast approach, and send to the closest sink.
- In the simplest case, a source wants to minimize the number of hops. To make anycast work, we only need to implement the regular distance-vector routing algorithm.
- However, one can imagine more complicated schemes where e.g. sink load is balanced, or even intermediate load is balanced.

Correlated Data

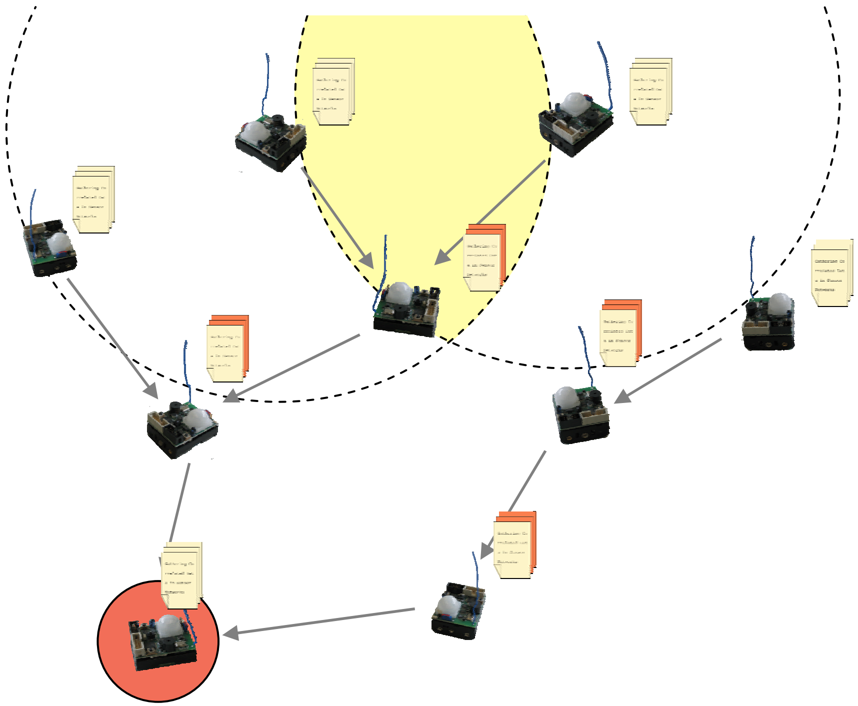
- Different sensor nodes partially monitor the same spatial region.

  ➡ Data correlation

- Data might be processed as it is routed to the information sink.

  ➡ In-network coding

At which node is node u's data encoded?



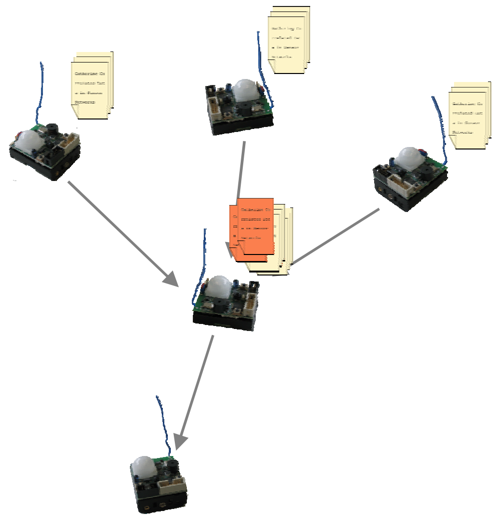
Find a routing scheme and a coding scheme to deliver data packets from all nodes to the sink such that the overall energy consumption is minimal.

Coding strategies

- Multi-input coding
  – Exploit correlation among several nodes.
  – Combined aggregation of all incoming data.

  ➡ Recoding at intermediate nodes

  ➡ Synchronous communication model



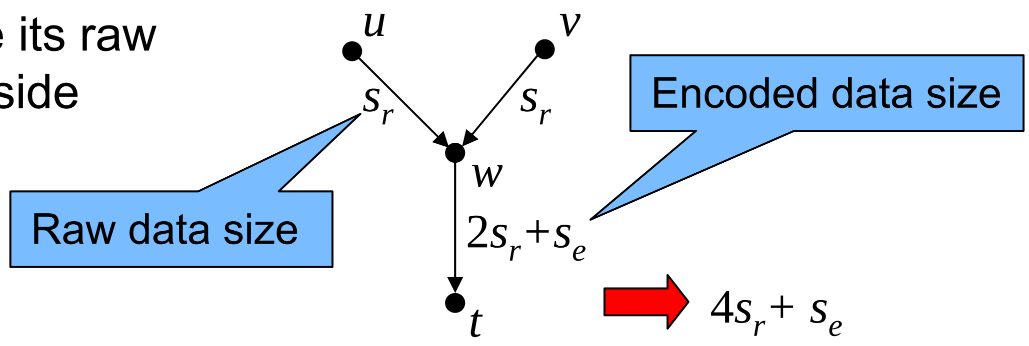
- Single-input coding
  – Encoding of a nodes data only depends on the side information of one other node.

  ➡ No recoding at intermediate nodes

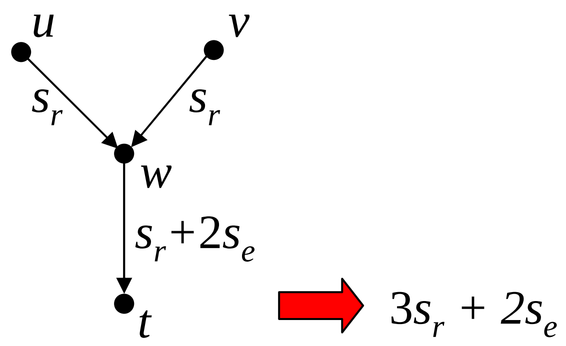
  ➡ No waiting for belated information at intermediate nodes

Single-input coding

- Self-coding
  - A node can only encode its raw data in the presence of side information.



- Foreign coding
  - A node can use its raw data to encode data it is relaying.



Self-coding

- Lower-bound the cost of an optimal

$$c_{opt} = \sum_{u \in B} \left(s_r \cdot \mathsf{ST}(S_u, u, t) + \sum_{v \in S_u} s_e \cdot \mathsf{SP}(v, t) \right) .$$

Set of nodes that encode with data from *u*

Set of nodes with no side information

Steiner tree

Shortest path

- Two ways to lower-bound this equation:
  - $c_{opt} \geq \sum_{u \in V} s_e \cdot \mathsf{SP}(u, t) \qquad (1)$
  - $c_{opt} \geq s_r \cdot c(\mathsf{MST}) \qquad (1)$

Algorithm

- LEGA (Low Energy Gathering Algorithm)
- Based on the shallow light tree (SLT)
- Compute SLT rooted at the sink t.
- The sink t transmits its packet p_t
- Upon reception of a data packet p_j at node v_i
  - Encode p_i with $p_j \rightarrow p_i^j$
  - Transmit p_i^j to the sink t
  - Transmit p_i to all children

Size = s_r

Size = s_e

Excursion: Shallow-Light Tree (SLT)

- Introduced by [Awerbuch, Baratz, Peleg, PODC 1990]
- Improved by [Khuller, Raghavachari, Young, SODA 1993]
  - new name: Light-Approximate-Shortest-Path-Tree (LAST)

- Idea: Construct a spanning tree for a given root r that is both a MST-approximation as well as a SPT-approximation for the root r. In particular, for any $\gamma > 0$
  - $c(\text{SLT}) \leq (1 + \sqrt{2}/\gamma) \cdot c(\text{MST})$
  - $d_{SLT}(v_i, r) \leq (1 + \sqrt{2}\gamma) \cdot \text{SP}(v_i, r)$

- Remember:
  - MST: Easily computable with e.g. Prim’s greedy edge picking algorithm
  - SPT: Easily computable with e.g. Dijkstra’s shortest path algorithm

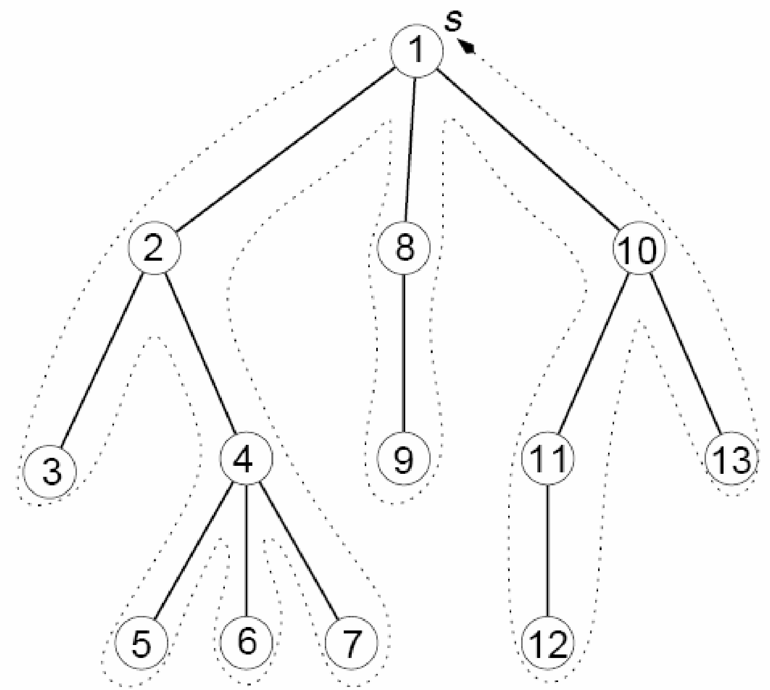
MST vs. SPT

- Is a good SPT not automatically a good MST (or vice versa)?

Result & Preordering

- Main Theorem: Given an $\alpha > 1$, the algorithm returns a tree T rooted at r such that all shortest paths from r to u in T have cost at most α the shortest path from r to u in the original graph (for all nodes u). Moreover the total cost of T is at most $\beta = 1+2/(\alpha-1)$ the cost of the MST.

- We need an ingredient: A preordering of a rooted tree is generated when ordering the nodes of the tree as visited by a depth-first search algorithm.

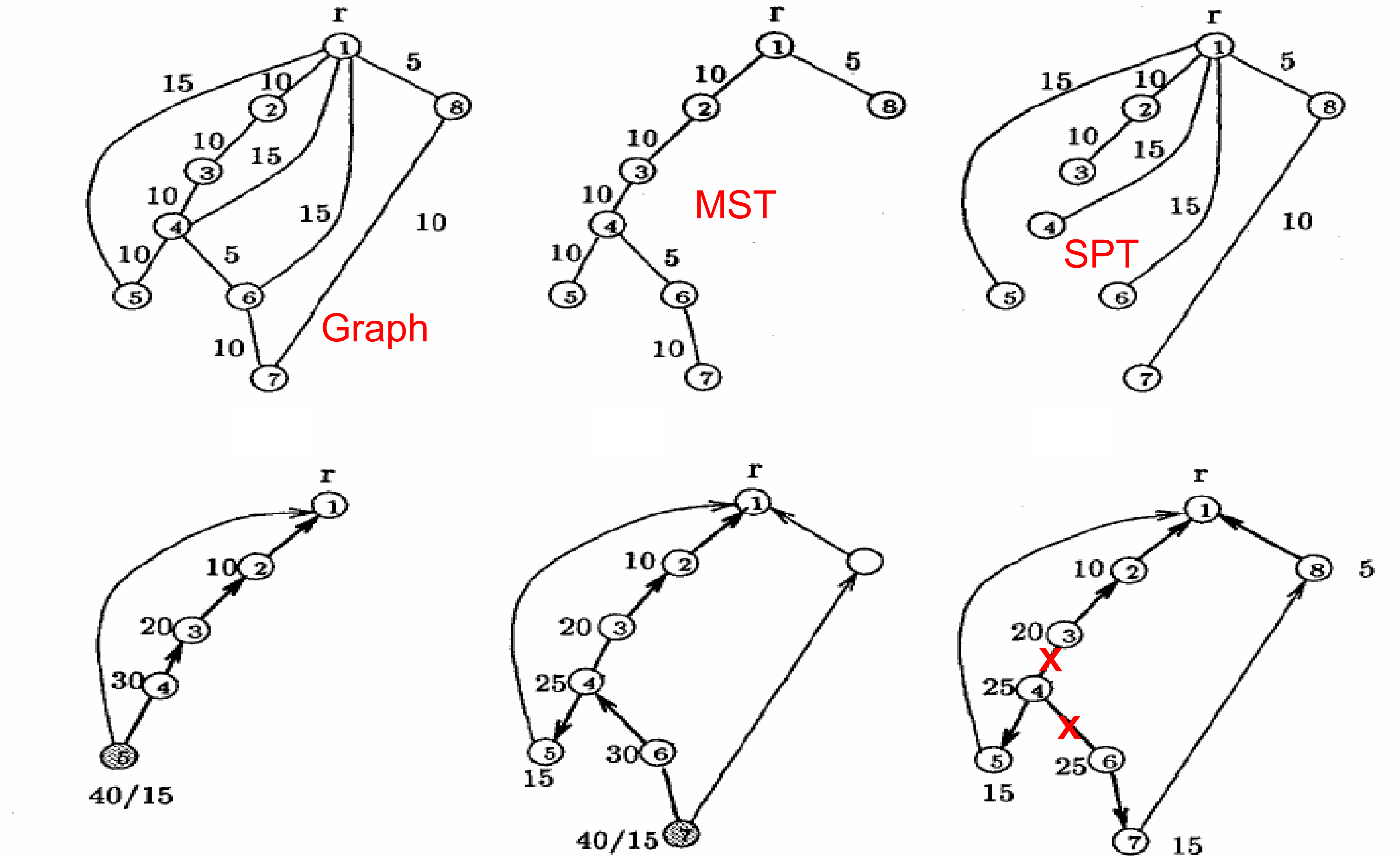


The SLT Algorithm

1. Compute MST H of Graph G;
2. Compute all shortest paths (SPT) from the root r.
3. Compute preordering of MST with root r.
4. For all nodes v in order of their preordering do
   - Compute shortest path from r to u in H. If the cost of this shortest path in H is more than a factor α more than the cost of the shortest path in G, then just add the shortest path in G to H.
5. Now simply compute the SPT with root r in H.

- Sounds crazy… but it works!

An example, $\alpha = 2$



Proof of Main Theorem

- The SPT α-approximation is clearly given since we included all necessary paths during the construction and in step 5 only removed edges which were not in the SPT.

- We need to show that our final tree is a β-approximation of the MST. In fact we show that the graph H before step 5 is already a β-approximation!

- For this we need a little helper lemma first…

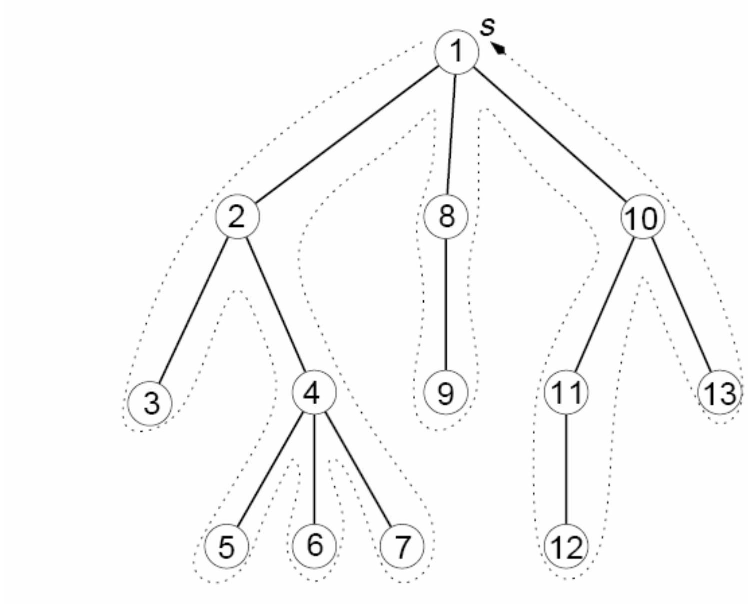
A preordering lemma

- Lemma: Let T be a rooted spanning tree, with root r, and let z_0, z_1, …, z_k be arbitrary nodes of T in preorder. Then,

$$\sum_{i=1}^{k} d_T(z_{i-1}, z_i) \leq 2 \cdot cost(T).$$

- "Proof by picture": Every edge is traversed at most twice.

- Remark: Exactly like the 2-approximation algorithm for metric TSP.



Proof of Main Theorem (2)

- Let $z_1, z_2, \ldots, z_k$ be the set of k nodes for which we added their shortest paths to the root r in the graph in step 4. In addition, let z_0 be the root r. The node z_i can only be in the set if (for example) $d_G(r,z_{i-1}) + d_{MST}(z_{i-1},z_i) > \alpha d_G(r,z_i)$, since the shortest path (r,z_{i-1}) and the path on the MST (z_{i-1},z_i) are already in H when we study z_i.

- We can rewrite this as $\alpha d_G(r,z_i) - d_G(r,z_{i-1}) < d_{MST}(z_{i-1},z_i)$. Summing up:

$$
\begin{array}{lcll}
\alpha d_G(r,z_1) - d_G(r,z_0) & < & d_{MST}(z_0,z_1) & (i=1) \\
\alpha d_G(r,z_2) - d_G(r,z_1) & < & d_{MST}(z_1,z_2) & (i=2) \\
\ldots & \ldots & \ldots & \\
\alpha d_G(r,z_k) - d_G(r,z_{k-1}) & < & d_{MST}(z_{k-1},z_k) & (i=k) \\
\hline
\Sigma_{i=1\ldots k}(\alpha-1)\, d_G(r,z_i) \quad \cancel{+\, d_G(r,z_k)} & < & \Sigma_{i=1\ldots k}\, d_{MST}(z_{i-1},z_i) &
\end{array}
$$

Proof of Main Theorem (3)

- In other words, $(\alpha-1)\ \Sigma_{i=1\ldots k}\ d_G(r,z_i) < \Sigma_{i=1\ldots k}\ d_{MST}(z_{i-1},z_i)$

- All we did in our construction of H was to add exactly at most the cost $\Sigma_{i=1\ldots k}\ d_G(r,z_i)$ to the cost of the MST. In other words, $cost(H) \leq cost(MST) + \Sigma_{i=1\ldots k}\ d_G(r,z_i)$.

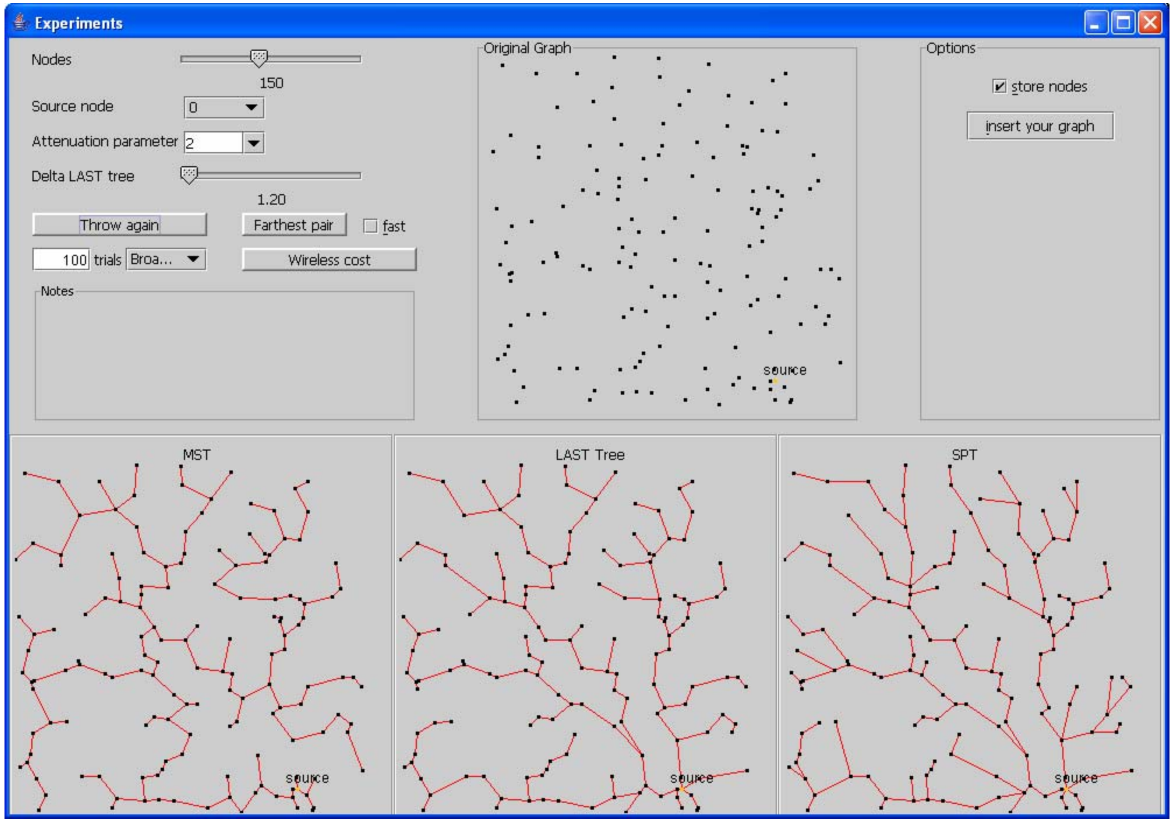
- Using the inequality on the top of this slide we have $cost(H) < cost(MST) + 1/(\alpha-1)\ \Sigma_{i=1\ldots k}\ d_{MST}(z_{i-1},z_i)$.

- Using our preordering lemma we have $cost(H) \leq cost(MST) + 1/(\alpha-1)\ 2cost(MST) = 1+2/(\alpha-1)\ cost(MST)$

- That's exactly what we needed: $\beta = 1+2/(\alpha-1)$.

How the SLT can be used

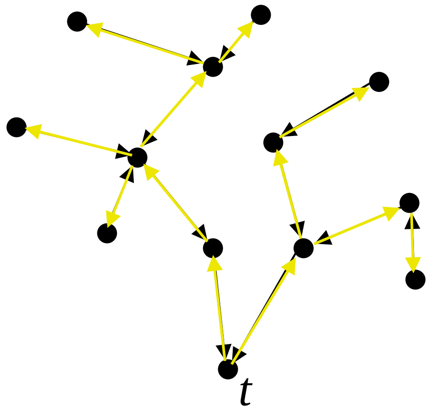
- The SLT has many applications in communication networks.
- Essentially, it bounds the cost of unicasting (using the SPT) and broadcasting (using the MST).
- Remark: If you use $\alpha = 1 + \sqrt{2}$, then $\beta = 1+2/(\alpha-1) = \alpha$.



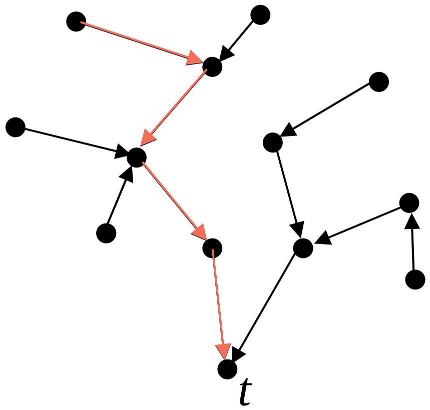
www.dia.unisa.it/~ventre

Analysis of LEGA

Theorem: LEGA achieves a $2(1+\sqrt{2})$-approximation of the optimal topology. (We use $\alpha = 1+\sqrt{2}$.)



$\Rightarrow s_r \cdot c(\mathsf{SLT})$



$\Rightarrow \sum_{v_i \in V} s_e \cdot d_{SLT}(v_i, t)$

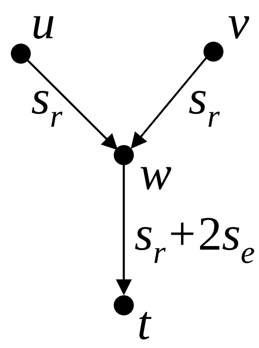
$$c_{LEGA} \leq s_r \cdot (1+\sqrt{2})c(\mathsf{MST}) + (1+\sqrt{2}) \sum_{v_i \in V} s_e \cdot \mathsf{SP}(v_i, t)$$

$$\leq 2(1+\sqrt{2})c_{opt}$$

Slide 9/10

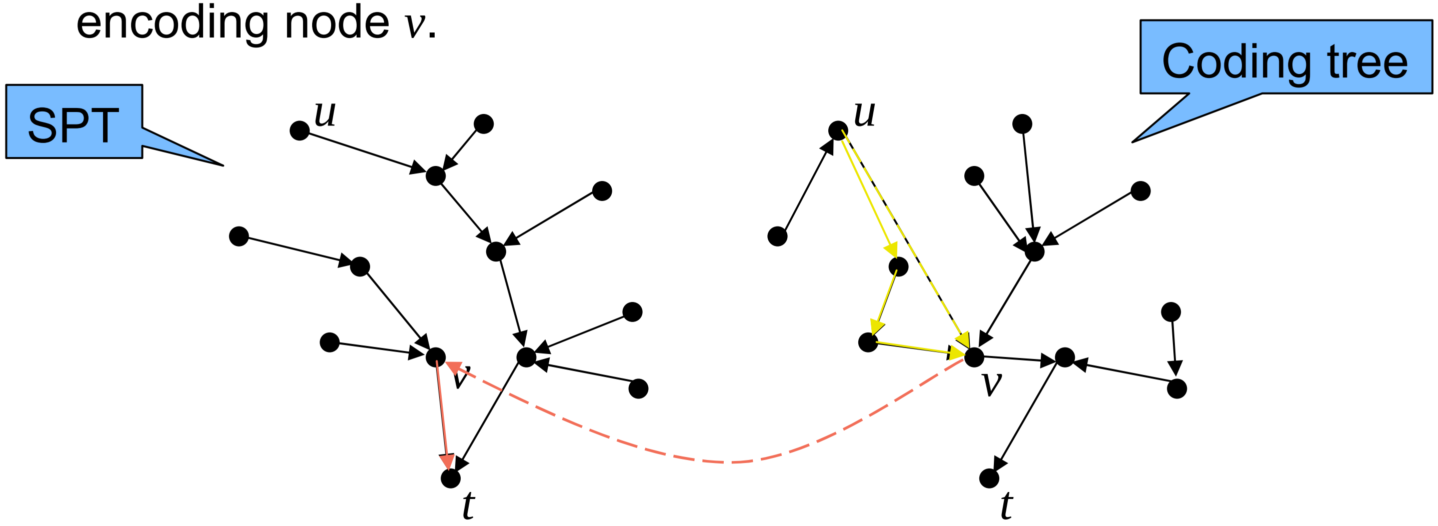
Foreign coding

- MEGA (Minimum-Energy Gathering Algorithm)
  – Superposition of two tree constructions.



- Compute the shortest path tree (SPT) rooted at t.
- Compute a coding tree.
  – Determine for each node u a corresponding encoding node v.

Encoding must not result in cyclic dependencies.



Coding tree construction

- Build complete directed graph
- Weight of an edge $e=(v_i,v_j)$

Cost from v_j to the sink t.

$$w(e) = s_i \cdot \mathsf{SP}(v_i, v_j) + s_i^j \cdot \mathsf{SP}(v_j, t)$$

Cost from v_i to the encoding node v_j.

Number of bits when encoding v_i's info at v_j

- Compute a directed minimum spanning tree (arborescence) of this graph. (This is not trivial, but possible.)

Theorem: MEGA computes a minimum-energy data gathering topology for the given network.

All costs are summarized in the edge weights of the directed graph.

Summary

- Self-coding:
  – The problem is NP-hard [Cristescu et al, INFOCOM 2004]
  – LEGA uses the SLT and gives a $2(1+\sqrt{2})$-approximation.
  – Attention: We assumed that the raw data resp. the encoded data always needs s_r resp. s_e bits (no matter how far the encoding data is!). This is quite unrealistic as correlation is usually regional.

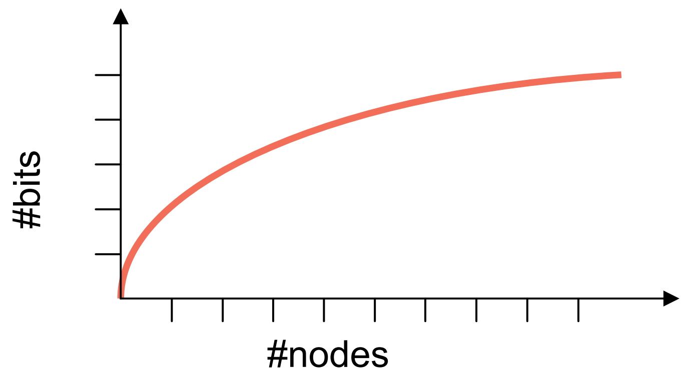
- Foreign coding
  – The problem is in P, as computed by MEGA.

- What if we allow both coding strategies at the same time?
- What if multicoding is still allowed?

Multicoding

- Hierarchical matching algorithm [Goel & Estrin SODA 2003].

- We assume to have concave, non-decreasing aggregation functions. That is, to transmit data from k sources, we need f(k) bits with f(0)=0, f(k) $\geq$ f(k-1), and f(k+1)/f(k) $\leq$ f(k)/f(k-1).



- The nodes of the network must be a metric space*, that is, the cost of sending a bit over edge (u,v) is c(u,v), with
  - Non-negativity: c(u,v) $\geq$ 0
  - Zero distance: c(u,u) = 0 (*we don't need the identity of indescernibles)
  - Symmetry: c(u,v) = c(v,u)
  - Triangle inequality: c(u,w) $\leq$ c(u,v) + c(v,w)

The algorithm

- Remark: If the network is not a complete graph, or does not obey the triangle inequality, we only need to use the cost of the shortest path as the distance function, and we are fine.

- Let S be the set of source nodes. Assume that S is a power of 2. (If not, simply add copies of the sink node until you hit the power of 2.) Now do the following:

1. Find a min-cost perfect matching in S.
2. For each of the matching edges, remove one of the two nodes from S (throw a regular coin to choose which node).
3. If the set S still has more than one node, go back to step 1. Else connect the last remaining node with the sink.

The result

- Theorem: For any concave, non-decreasing aggregation function f, and for [optimal] total cost C[*], the hierarchical matching algorithm guarantees

$$E\left[\max_{\forall f} \frac{C(f)}{C^*(f)}\right] \leq 1 + \log k.$$

- That is, the expectation of the worst cost overhead is logarithmically bounded by the number of sources.

- Proof: Too intricate to be featured in this lecture.

Remarks

- For specific concave, non-decreasing aggregation functions, there are simpler solutions.
  - For f(x) = x the SPT is optimal.
  - For f(x) = const (with the exception of f(0) = 0), the MST is optimal.
  - For anything in between it seems that the SLT again is a good choice.
  - For any a priori known f one can use a deterministic solution by [Chekuri, Khanna, and Naor, SODA 2001]
  - If we only need to minimize the maximum expected ratio (instead of the expected maximum ratio), [Awerbuch and Azar, FOCS 1997] show how it works.

- Again, sources are considered to aggregate equally well with other sources. A correlation model is needed to resemble the reality better.

Other work using coding

- LEACH [Heinzelman et al. HICSS 2000]: randomized clustering with data aggregation at the clusterheads.
  - Heuristic and simulation only.
  - For provably good clustering, see the next chapter.

- Correlated data gathering [Cristescu et al. INFOCOM 2004]:
  - Coding with Slepian-Wolf
  - Distance independent correlation among nodes.
  - Encoding only at the producing node in presence of side information.
  - Same model as LEGA, but heuristic & simulation only.
  - NP-hardness proof for this model.

TinyDB and TinySQL

- Use paradigms familiar from relational databases to simplify the “programming” interface for the application developer.

- TinyDB then supports in-network aggregation to speed up communication.

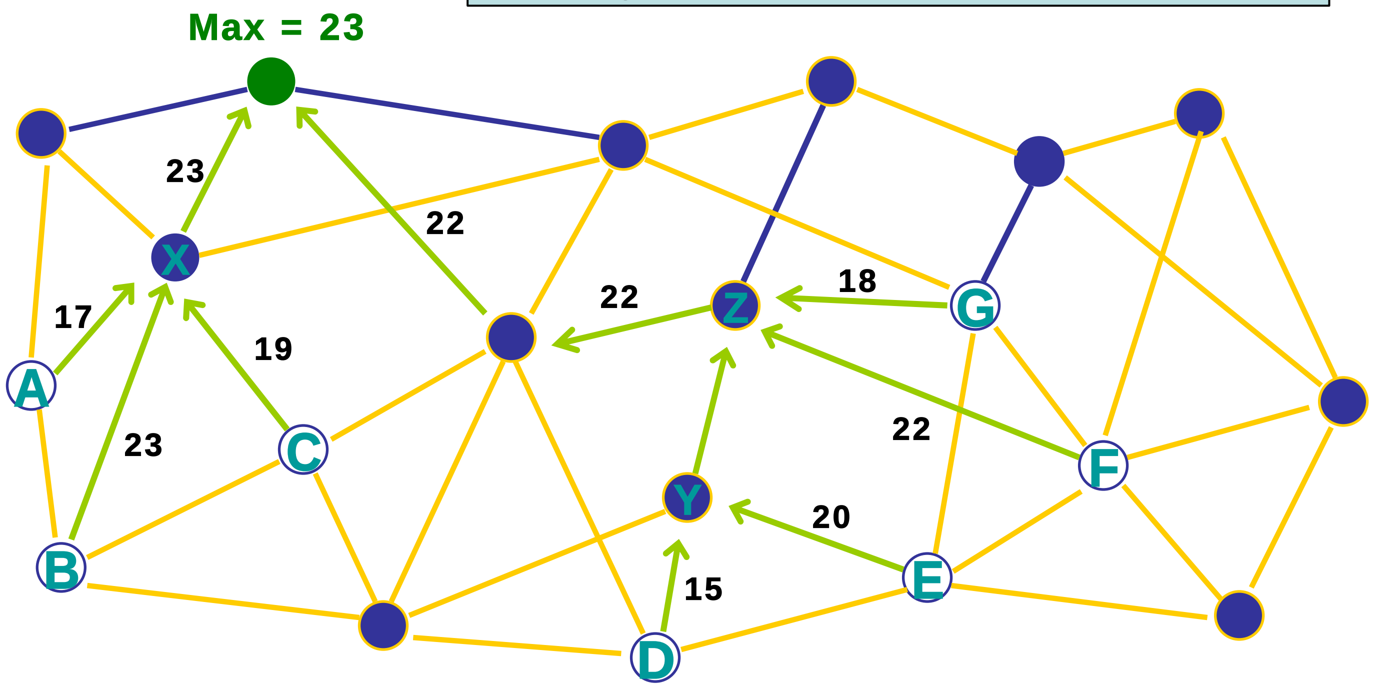
```
SELECT roomno, AVERAGE(light), AVERAGE(volume)
FROM sensors
GROUP BY roomno
HAVING AVERAGE(light) > l AND AVERAGE(volume) > v
EPOCH DURATION 5min
```

SELECT <aggregates>, <attributes>
[FROM {sensors | <buffer>}]
[WHERE <predicates>]
[GROUP BY <exprs>]
[SAMPLE PERIOD <const> | ONCE]
[INTO <buffer>]
[TRIGGER ACTION <command>]

Data Aggregation: N-to-1 Communication

- SELECT MAX(temp) FROM sensors WHERE temp > 15.

Average, Median, Count Distinct, ...?!



Selective data aggregation

- In sensor network applications
  - Queries can be frequent
  - Sensor groups are time-varying
  - Events happen in a dynamic fashion
- Option 1: Construct aggregation trees for each group
  - Setting up a good tree incurs communication overhead
- Option 2: Construct a single spanning tree
  - When given a sensor group, simply use the induced tree

Group-Independent (a.k.a. Universal) Spanning Tree

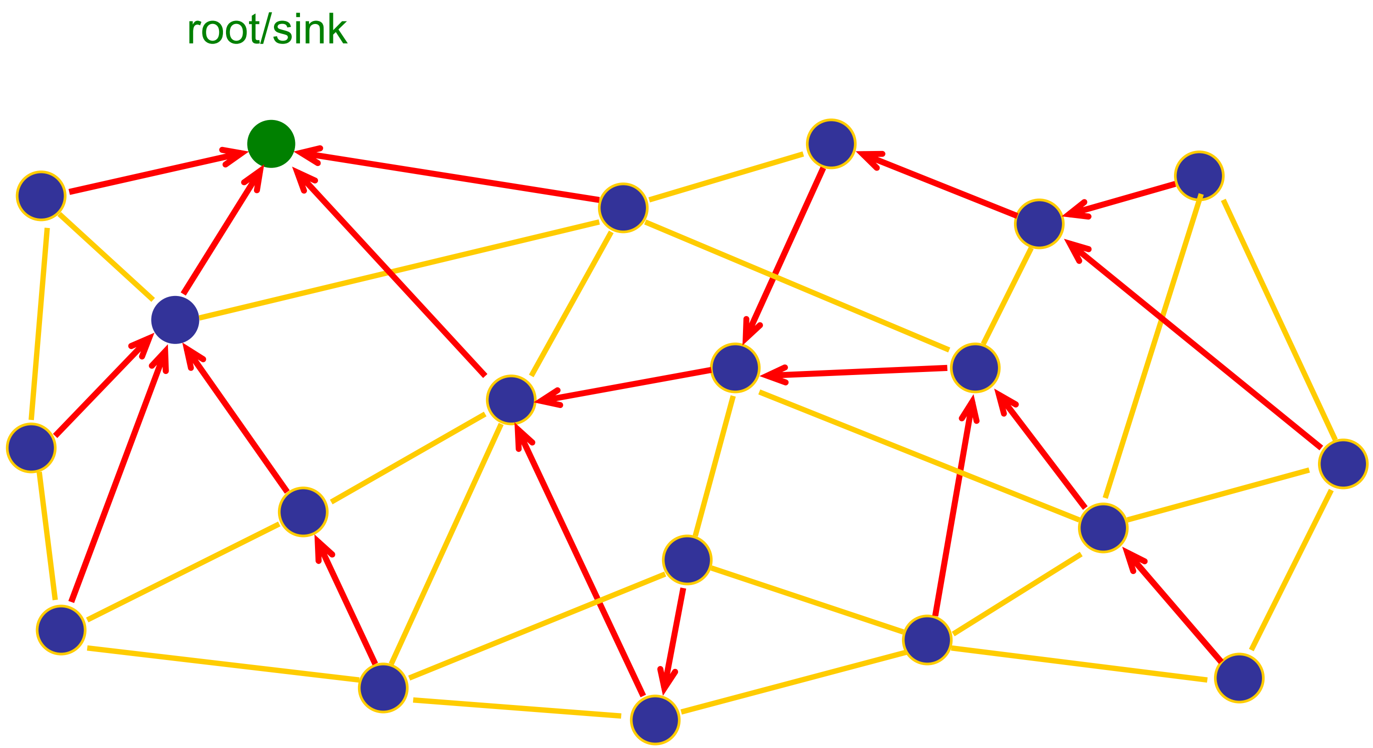
- Given
  - A set of nodes V in the Euclidean plane (or forming a metric space)
  - A root node $r \in V$
  - Define stretch of a universal spanning tree T to be

$$\max_{S \subseteq V} \frac{\text{cost(induced tree of S+r on T)}}{\text{cost(minimum Steiner tree of S+r)}}.$$

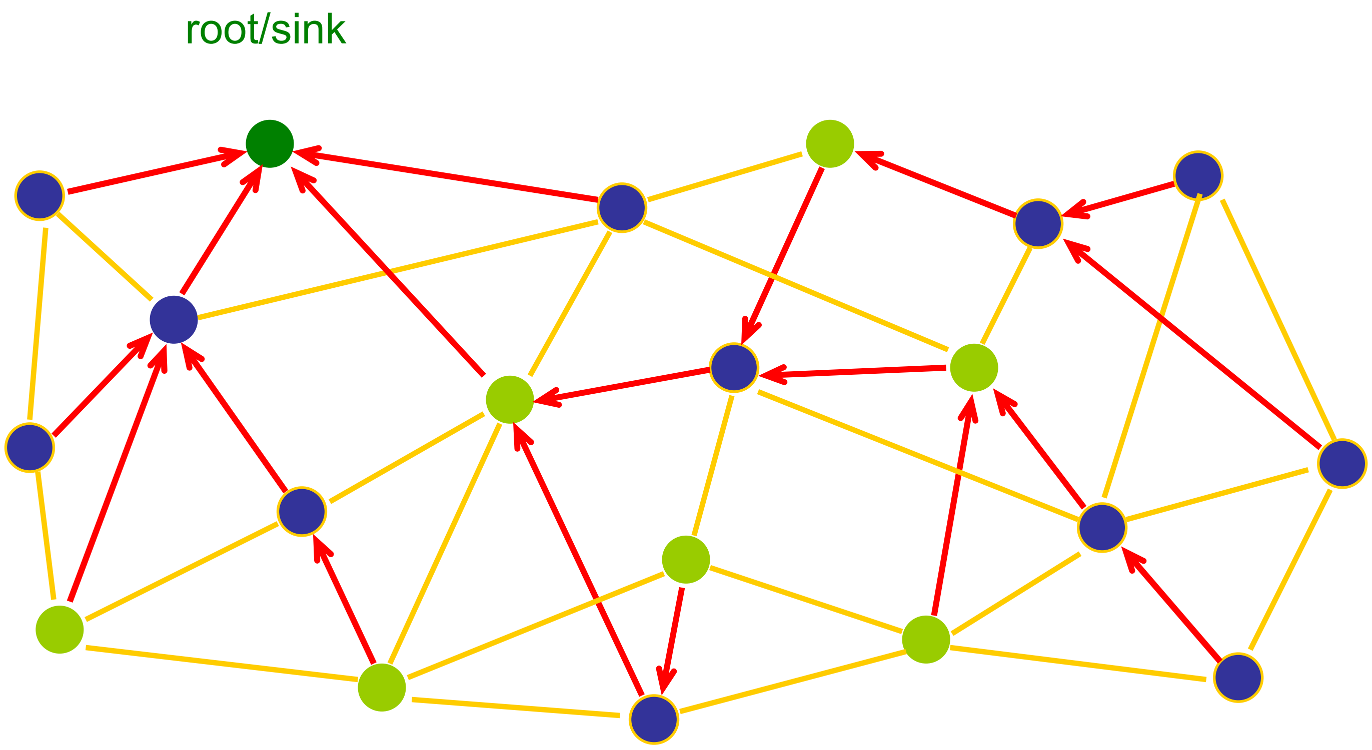
- We're looking for a spanning tree T on V with minimum stretch.

Example

- The red tree is the universal spanning tree. All links cost 1.

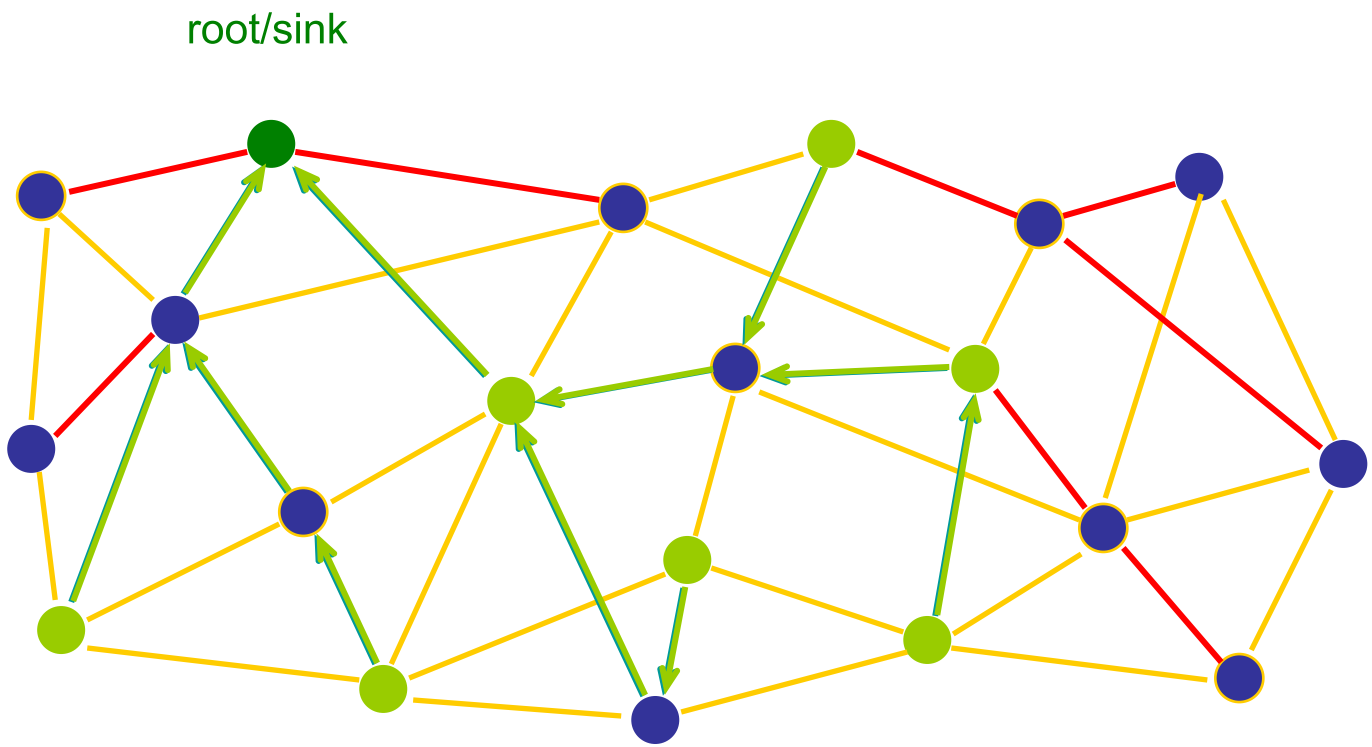


Given the lime subset…



Induced Subtree

- The cost of the induced subtree for this set S is 11. The optimal was 8.



Main results

- [Jia, Lin, Noubir, Rajaraman and Sundaram, STOC 2005]

- Theorem 1: (Upper bound)

  For the minimum UST problem on Euclidean plane, an approximation of O(log n) can be achieved within polynomial time.

- Theorem 2: (Lower bound)

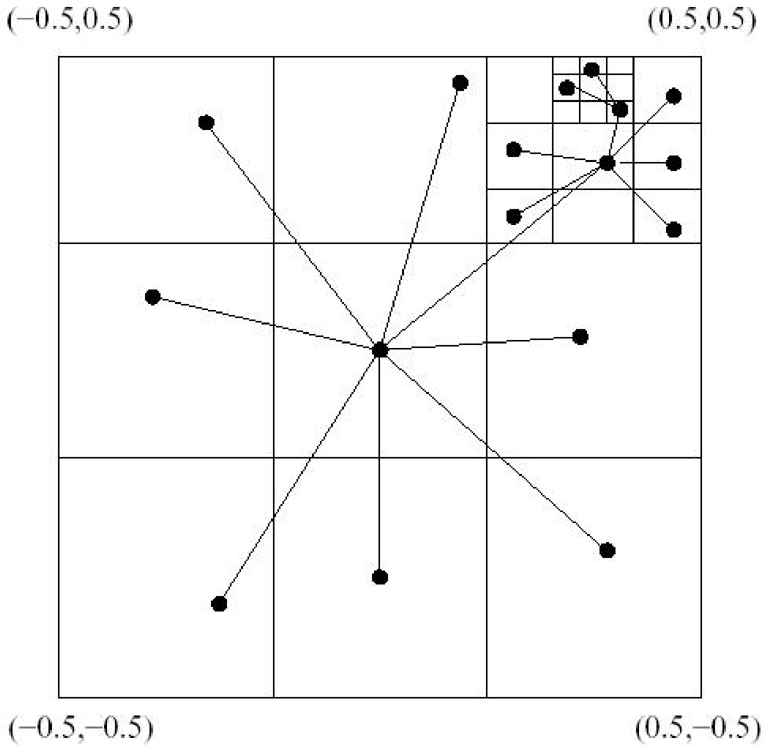
  No polynomial time algorithm can approximate the minimum UST problem with stretch better than $\Omega(\log n / \log \log n)$.

- Proofs: Not in this lecture.

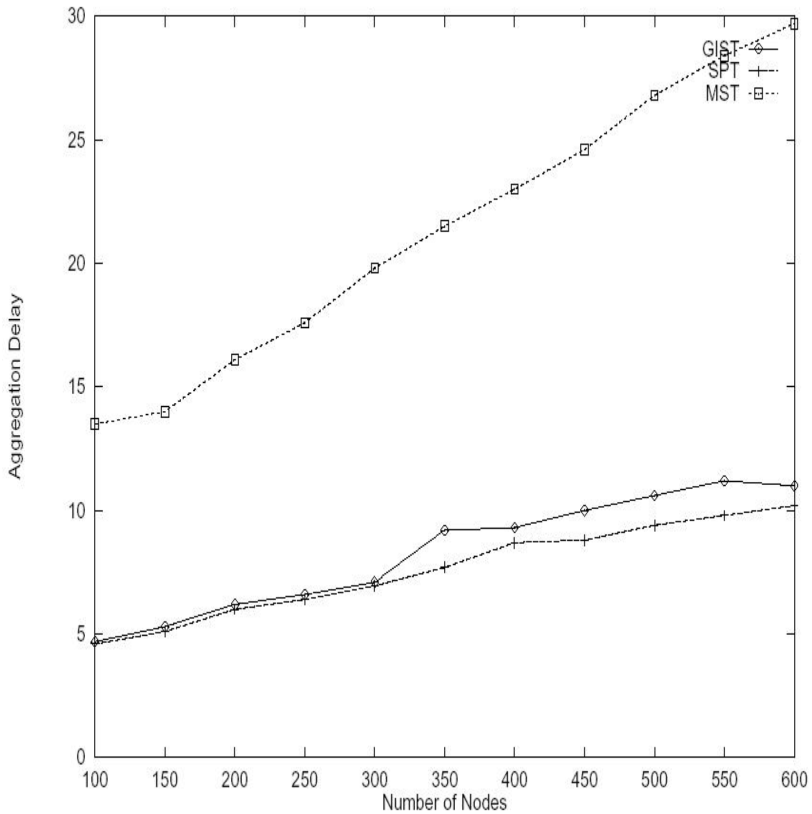
Algorithm sketch

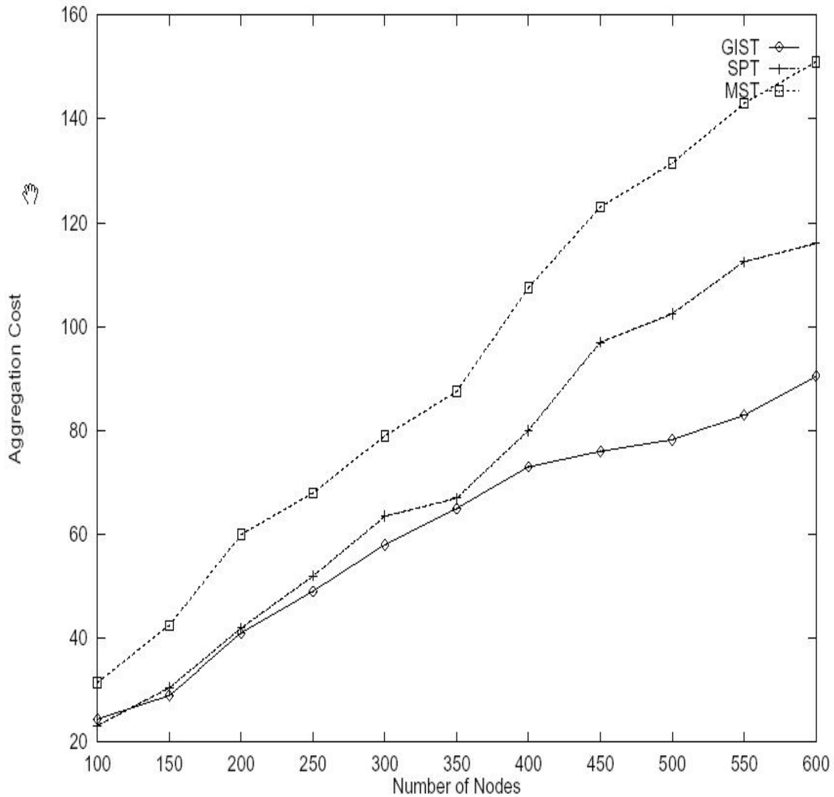
- For the simplest Euclidean case:
- Recursively divide the plane and select random node.

- Results: The induced tree has logarithmic overhead. The aggregation delay is also constant.



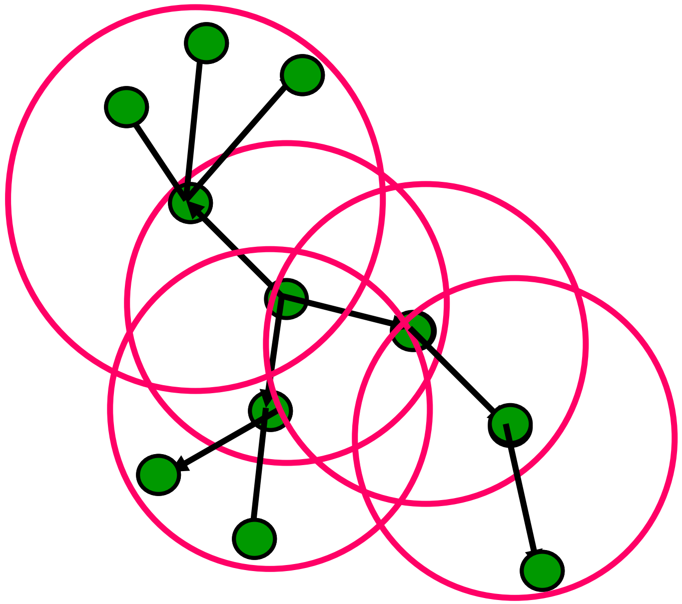
Simulation with random node distribution & random events





Minimum Energy Broadcasting

- First step for data gathering, sort of.
- Given a set of nodes in the plane
- Goal: Broadcast from a source to all nodes
- In a single step, a node may transmit within a range by appropriately adjusting transmission power.
- Energy consumed by a transmission of radius r is proportional to r^α, with $\alpha \geq 2$.
- Problem: Compute the sequence of transmission steps that consume minimum total energy, even in a centralized way.



[Rajomohan Rajaraman]

Three natural greedy heuristics

- In a tree, power for each parent node proportional to α'th exponent of distance to farthest child in tree:

- Shortest Paths Tree (SPT)
- Minimum Spanning Tree (MST)
- Broadcasting Incremental Power (BIP)
  - "Node" version of Dijkstra's SPT algorithm
  - Maintains an arborescence rooted at source
  - In each step, add a node that can be reached with minimum increment in total cost.

- Results:
  - NP, not even PTAS, there is a constant approximation. [Clementi, Crescenzi, Penna, Rossi, Vocca, STACS 2001]
  - Analysis of the three heuristics. [Wan, Calinescu, Li, Frieder, Infocom 2001]
  - Better and better approximation constants, e.g. [Ambühl, ICALP 2005]

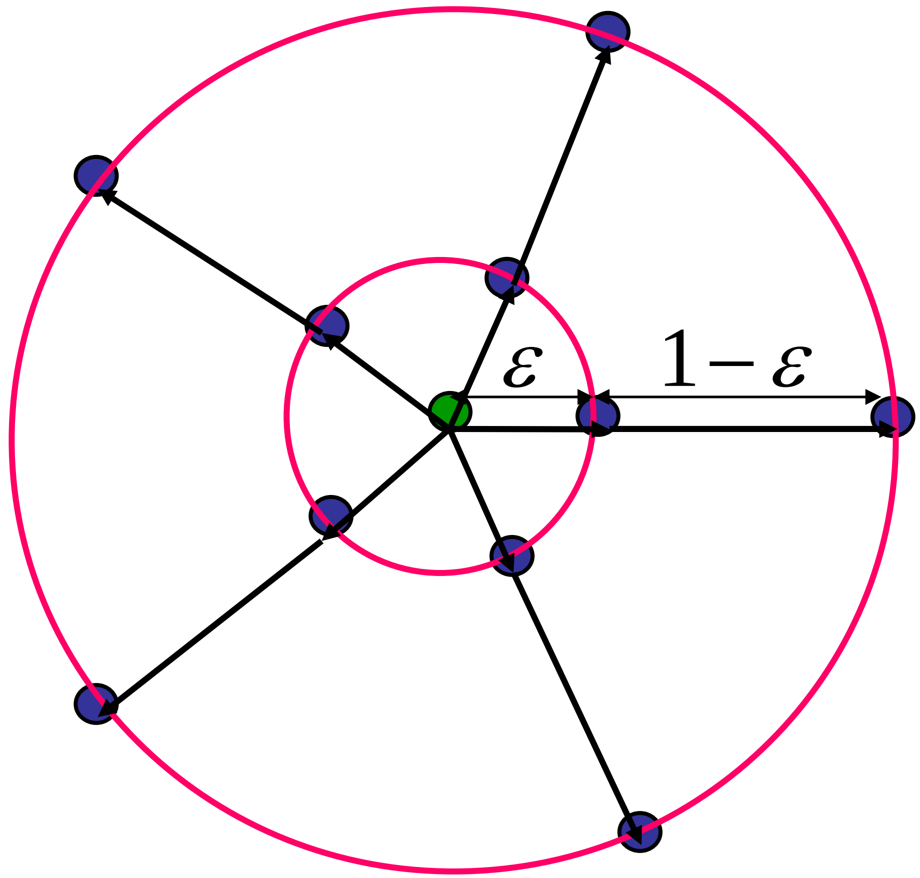
Lower Bound on SPT

- Assume (n-1)/2 nodes per ring

- Total energy of SPT:
$(n-1)(\varepsilon^{\alpha}+(1-\varepsilon)^{\alpha})/2$

- Better solution:
- Broadcast to all nodes
- Cost 1

- Approximation ratio Ω(n).



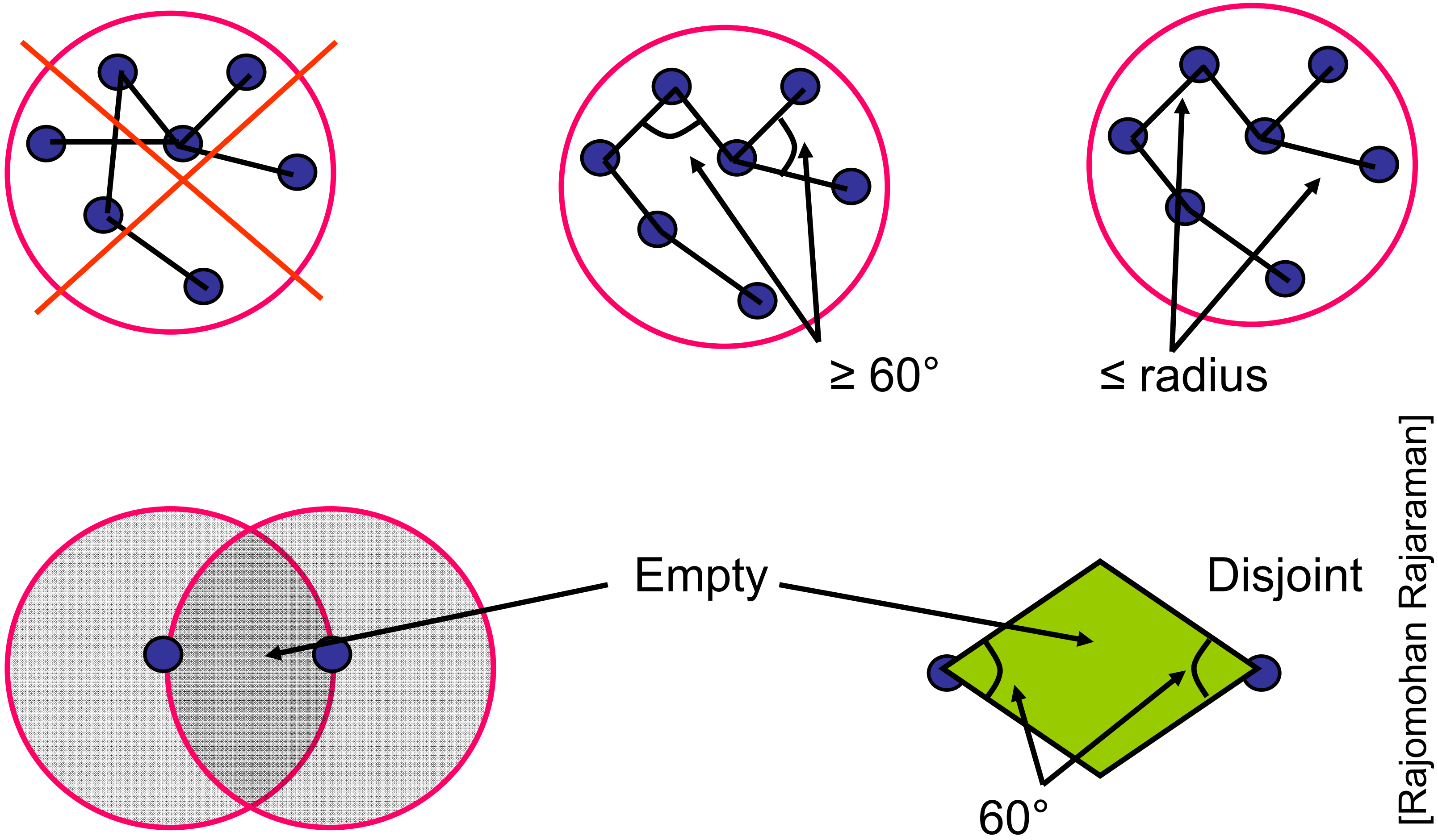
Performance of the MST Heuristic

- Weight of an edge (u,v) equals $d(u,v)^\alpha$.

- MST for these weights same as Euclidean MST
  - Weight is an increasing function of distance
  - Follows from correctness of Prim's algorithm

- Upper bound on total MST weight
- Lower bound on optimal broadcast tree

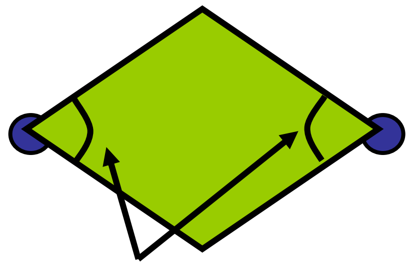
Structural Properties of MST



[Rajomohan Rajaraman]

Upper Bound on Weight of MST

- Assume $\alpha = 2$
- For each edge e, its diamond accounts for an area of exactly $\frac{|e|^2}{2\sqrt{3}}$



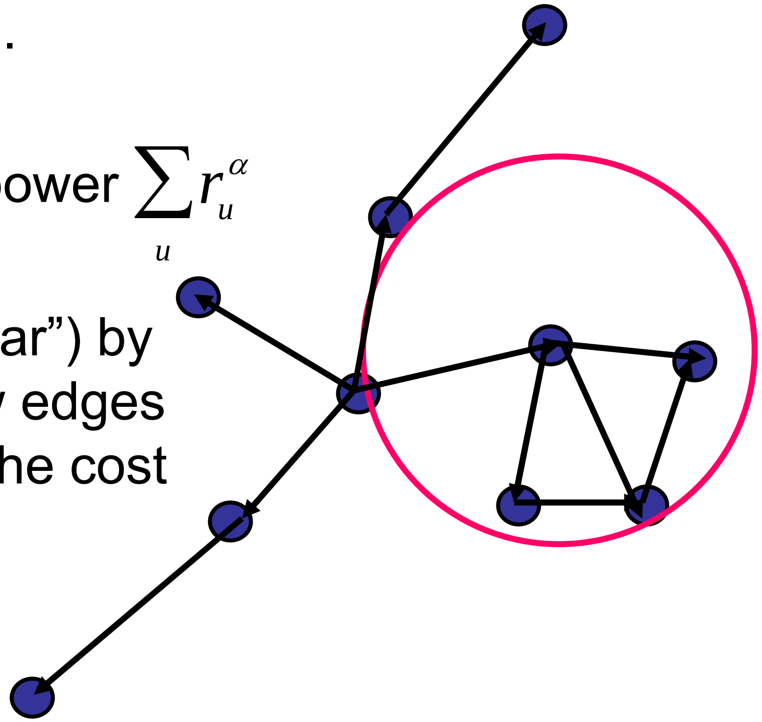
- Diamonds for edges in circle can be slightly outside circle, but not too much: The radius factor is at most $2/\sqrt{3}$, hence the total area accounted for is at most $\pi(2/\sqrt{3})^2 = 4\pi/3$
- Now we can bound the cost of the MST in a unit disk with

$$\text{cost}(MST) \leq \sum_e |e|^2 = 2\sqrt{3}\sum_e \frac{|e|^2}{2\sqrt{3}} \leq 2\sqrt{3}\frac{4\pi}{3} = \frac{8\pi}{\sqrt{3}} \approx 14.51.$$

- This analysis can be extended to $\alpha > 2$, and improved to 12.

Lower Bound on Optimal and Conclusion of Proof

- Also the optimal algorithm needs a few transmissions. Let u_0, u_1, …, u_k be the nodes which need to transmit, each u_i with radius r_i. These transmissions need to form a spanning tree since each node needs to receive at least one transmission.



- Then the optimal algorithm needs power $\sum_u r_u^\alpha$
- Now replace each transmission ("star") by an MST of the nodes. Since all new edges are part of the transmission circle, the cost of the new graph is at most $12\sum_u r_u^\alpha$
- Since the cost of the global MST is at most the cost of this spanner, the MST is 12-competitive.