The Minimum Energy Broadcast Problem
i.e.
The Minimum Spanning Tree Problem

Prof. Tiziana Calamoneri
Network Algorithms
A.y. 2017/18

The Problem

The Problem (1)

- As we already know, a wireless *ad-hoc* network consists of a set S of (fixed) radio stations joint by wireless connections.
- We assume that stations are located on the Euclidean plane (only partially realistic hp).
- Nodes have omnidirectional antennas: each transmission is listened by all the neighborhood (natural broadcast)
- …

The Problem (2)

What does it means "sufficiently close"?…

- Two stations communicate either directly (single-hop) -if they are sufficiently close- or through intermediate nodes (multi-hop).
- A transmission range is assigned to every station: a *range assignment* $r : S \rightarrow R$ determines a directed communication graph $G=(S,E)$, where edge $(i, j) \in E$ iff $\text{dist}(i, j) \leq r(i)$ ($\text{dist}(i, j)$= euclidean distance between i and j).
- In other words, $(i, j) \in E$ iff j belongs to the disk centered at i and having radius $r(i)$.

The Problem (3)

- For reasons connected with energy saving, each station can dynamically modulate its own transmission power.
- In fact, the transmission radius of a station depends on the energy power supplied to the station.
- The general aim is to save energy as much as possible.

The Problem (4)

- In particular, the power P_s required by a station s to transmit data to another station t must satisfy:

$$\frac{P_s}{dist(s,t)^{\alpha}} \geq 1$$

where $\alpha \geq 1$ is the distance-power gradient.

Usually $2 \leq \alpha \leq 4$ (it depends on the envorinment).

In the empty space $\alpha = 2$.

- Hence, in order to have a communication from s to t, power P_s must be proportional to $dist(s,t)^{\alpha}$

The Problem (5)

- Stations of an *ad hoc* network cooperate in order to provide specific network connectivity properties by adapting their transmission ranges and, at the same time, they try to save energy.
- …

The Problem (6)

- … According to the required property, different problems are proposed.
- For example:
  - The transmission graph is required to be strongly connected. In such a case, the problem is NP-hard and there is a 2-approximate alg. in 2 dim. [Kirousis, Kranakis, Krizanc, Pelc '01]; there exists an $r>1$ s.t. the problem is not r-approximable.
  - The transmission graph is required to have diameter at most h. Not trivial approximate results are not known.
  - Given a source node s, the transmission graph is required to include a spanning tree rooted at s. …

THE PROBLEM (7)

In this latter case:

- A Broadcast Range Assignment (for short *Broadcast*) is a range assignment that yields a communication graph G containing a directed spanning tree rooted at a given source station s.
- A fundamental problem in the design of *ad-hoc* wireless networks is the Minimum-Energy Broadcast problem (for short *Min Broadcast*), that consists in finding a broadcast of minimal *overall energy*.

THE PROBLEM (8)

Th. *Min Broadcast is not approximable within any constant factor.*

Proof. Recall the *MinSetCover* problem:

given a collection C of subsets of a finite set S, find a subset C' of C with min cardinality, s.t. each element in S belongs to at least one element of C'.

Example:

$S=\{1,2,3,4,5\}$ $C=\{\{1,2\}, \{1,2,3\}, \{3\}, \{3,4,5\}\}$

$C'=\{\{1,2,3\},\{3,4,5\}\}$

THE PROBLEM (9)

Proof (cntd).

Note. *MinSetCover* is not approximable within $c \log n$ for some constant $c>0$, where $n=|S|$.

Given an instance x of *MinSetCover* it is possible to construct an instance y of *MinBroadcast* s.t. there exists a solution for x of cardinality k iff there exists a solution for y of cost $k+1$.

So, if *MinBroadcast* is approximable within a constant, then even *MinSetCover* is. Contradiction.

THE PROBLEM (10)

Proof (cntd). Reduction:

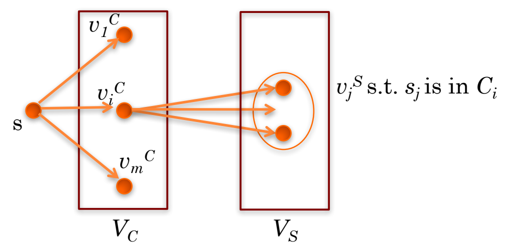
$x=(S,C)$ instance of *MinSetCover* with:

$S=\{s_1, s_2, \ldots, s_n\}$ and $C=\{C_1, C_2, \ldots, C_m\}$.

We construct $y=(G,w,s)$ of *MinBroadcast*.

Nodes of G: $\{s\} \cup \{V_C\} \cup \{V_S\}$

Edges of G: $\{(s, v_i^C), 1\le i\le m\} \cup \{(v_i^C, v_j^S), 1\le i\le m, \text{ s.t. } s_j \text{ in } C_i\}$



The Problem (11)

Proof (cntd).

Finally, define $w(e)=1$ for any edge e.

Let C' be a solution for x.

A sol. for y assigns 1 to s and to all nodes of V_C in C'.

The resulting transmission graph contains a spanning tree rooted at s because each element in S is contained in at least one element of C'. The cost of such a solution is $|C'|+1$.

The Problem (14)

Proof (cntd).

…

Conversely, assume that r is a feasible sol. for y, (w.l.o.g. $r(v)$ is either 0 or 1 if v is in V_C: other values would be meaningless) and $r(v)=0$ if v is in V_S.

We derive a solution C' for x selecting all subsets C_i s.t. $r(v_i^C)=1$.

It holds that $|C'|=cost(r)-1$. ■

The Problem (15)

Note

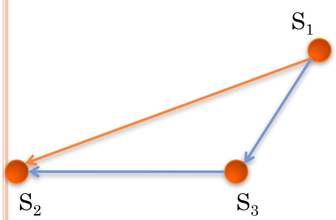
We proved that *Min Broadcast* is not approximable within a constant factor, but we have dealt with the general problem.

There are some special cases (e.g. the Euclidean bidimensional one) that are particularly interesting and that behave better!

In the following, we restrict to the special case of Euclidean plane…

The Problem (16)

- Collaborating in order to minimize the overall energy is crucial:



  - S_1 needs to communicate with S_2
  - let $\boldsymbol{\alpha}=2$
  - Cost of $S_1 \rightarrow S_2 = dist(S_1, S_2)^2$
  - Cost of $S_1 \rightarrow S_3 \rightarrow S_2 =$ $dist(S_1, S_3)^2+dist(S_3, S_2)^2$
  - When angle $S_1S_3S_2$ is obtuse: $dist(S_1, S_2)^2 >$ $dist(S_1, S_3)^2+dist(S_3, S_2)^2$

The Problem (17)

- In the Euclidean case, a range assignment r can be represented by the correspondent family $D = \{D_1, \ldots, D_l\}$ of disks, and the overall energy is defined as:

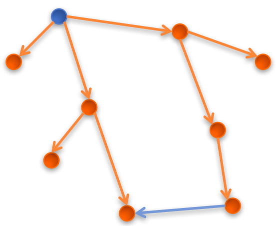
$$\cos t(D) = \sum_{i=1}^{l} r_i^{\alpha}$$

where r_i is the radius of D_i.

The Problem (18)

- Consider the complete and weighted graph $G^{(\alpha)}$ where the weight of each arc $e=(u,v)$ is $dist(u,v)^{\alpha}$.
- The broadcast problem is strictly related with the minimum spanning tree on $G^{(\alpha)}$, in view of some important properties…

The Problem (19)



The set of connections used to perform a broadcast from s:

- cannot generate a cycle, because nodes do not need to be informed twice

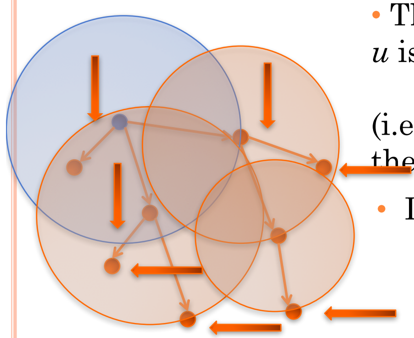
tree

- minimizes the overall energy

long arcs waste more energy than short ones.

The Problem (20)

- Nevertheless, the Minimum Broadcast problem is not the same as the Min Spanning Tree problem:



- The energy used by each node u is

$$\max_{(u,v)\in T}\{dist(u,v)\}^{\alpha}$$

(i.e. not all the arcs appear with their contribution)

- Leaves waste no energy

The Problem (21)

- The Minimum Broadcast problem is NP-hard in its general version and it is neither approximable within (1-ε)Δ where Δ is the maximum degree of T and ε is an arbitrary constant
- Nothing is known about the hardness of the geometric version (i.e. on the Euclidean plane).

The Problem (22)

- An approx algorithm is based on the computation of the MST:
  - compute the MST of the complete graph induced by S,
  - Assign a direction to arcs (from s to the leaves)
  - Assign to each node i a radius equal to the length of the longest arc outgoing from i
- Easy to implement ➔ deep analysis of the approx ratio.
  - [Clementi+al.'01] the first constant approx ratio (about 40)
  - [Ambüehl '05] the best (tight) known approx ratio (6)

The Minimum Spanning Tree Problem (recup)

Minimum Spanning Tree (1)

- Obs. 1: If the weights are *positive*, then a MST is in fact a minimum-cost subgraph connecting all nodes.
- Proof: A subgraph containing cycles necessarily has a higher total weight. ■

- Obs. 2: There may be several minimum spanning trees of the same weight having a minimum number of edges.
- In particular, if all the edge weights of a given graph are the same, then every spanning tree of that graph is minimum.

Minimum Spanning Tree (2)

- Obs. 3: If each edge has a distinct weight, then there is a unique MST.
- This is true in many realistic situations, where it's unlikely that any two connections have *exactly* the same cost
- Proof: Assume by contradiction that MST T is not unique. So, there is another MST with equal weight, say T'.

  …

Minimum Spanning Tree (3)

(proof – cntd)

- Let e_1 be an edge that is in T but not in T'. As T' is a MST, $\{e_1\} \cup T'$ contains a cycle C and there is at least one edge e_2 in T' that is not in T and lies on C.
- If the weight of e_1 is less than that of e_2:

  replacing e_2 with e_1 in T' yields tree $\{e_1\} \cup T' \setminus \{e_2\}$ which has a smaller weight compared to T'.

  Contradiction, as we assumed T' is a MST but it is not.
- If the weight of e_1 is larger than that of e_2:
  a similar argument involving tree $\{e_2\} \cup T \setminus \{e_1\}$ also leads to a contradiction.
- We conclude that the assumption that there is a further MST was false. ■

Minimum Spanning Tree (4)

- Obs. 4: For any cycle C in the graph, if the weight of an edge e of C is larger than the weights of all other edges of C, then this edge cannot belong to an MST.
- Proof: Assuming the contrary, i.e. that e belongs to an MST T_1, then deleting e will break T_1 into two subtrees with the two endpoints of e in different subtrees. The remainder of C reconnects the subtrees, in particular there is an edge f of C with endpoints in different subtrees, i.e., it reconnects the subtrees into a tree T_2 with weight less than that of T_1, because the weight of f is less than the weight of e. ■

Minimum Spanning Tree (5)

- Obs. 5: If the edge of a graph with the minimum cost e is unique, then this edge is included in any MST.
- Proof: If e was not included in the MST, removing any of the (larger cost) edges in the cycle formed after adding e to the MST, would yield a spanning tree of smaller weight. ■

Minimum Spanning Tree (6)

- Obs. 6: For any cut C in the graph, if the weight of an edge e of C is strictly smaller than the weights of all other edges of C, then this edge belongs to all MSTs of the graph.
- Proof: If e was not included in the MST, adding e to the MST produces a cycle. Removing any of the (larger cost) edges of the cut in the cycle, would yield a spanning tree of smaller weight. ■
- By similar arguments, if more than one edge is of minimum weight across a cut, then each such edge is contained in a minimum spanning tree.

Minimum Spanning Tree (7)

Three classical algorithms:

- Kruskal ['56]
- Prim ['57]
- Boruvka ['26]

Minimum Spanning Tree (8)

- The three algorithms are all greedy algorithms and based on the same structure:
  - Given a set of arcs A containing some MST arcs, e is a safe arc w.r.t. A if $A \cup \{e\}$ contains only MST arcs, too.
  - A=empty set

    While A is not a MST

    find a safe arc e w.r.t. A "difficult" issue

    $A=A \cup e$

Minimum Spanning Tree (9)

- A=empty set

  while A is not a MST

  find a safe arc e w.r.t. A

  $A=A \cup e$

whenever:

- A is acyclic
- graph $G_A=(V, A)$ is a forest whose each connected component is either a node or a tree
- Each safe arc connects different connected components of G_A
- the while loop is run $n-1$ times

KRUSKAL ALGORITHM (1)

- *A*=empty set

  While G_A is not a MST

  find a safe arc *e* w.r.t. *A*

  $A=A \cup \{e\}$

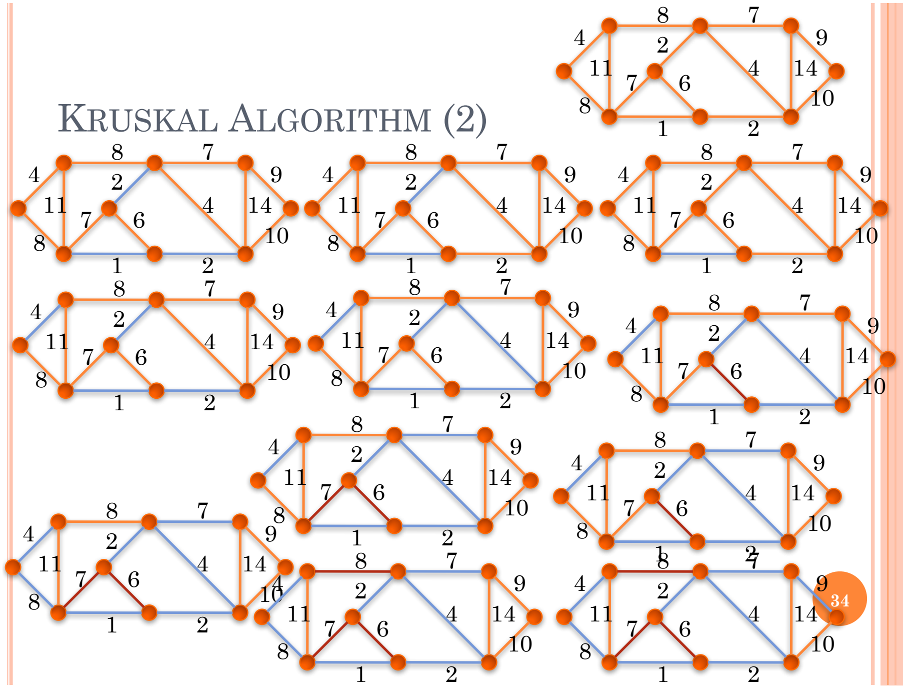
Among those connecting two different connected components in G_A, choose the one with minimum weight

Implementation using:

- Data structure Union-Find
- The set of the arcs of *G* is sorted w.r.t. their weight
- Time Complexity: $O(m \log n)$

[Johnson '75, Cheriton & Tarjan '76]

KRUSKAL ALGORITHM (2)



PRIM ALGORITHM (1)

- *A*=empty set

  While G_A is not a MST

  find a safe arc *e* w.r.t. *A*

  $A=A \cup \{e\}$

Among those connecting the main connected component with an isolated node, choose the one with minimum weight

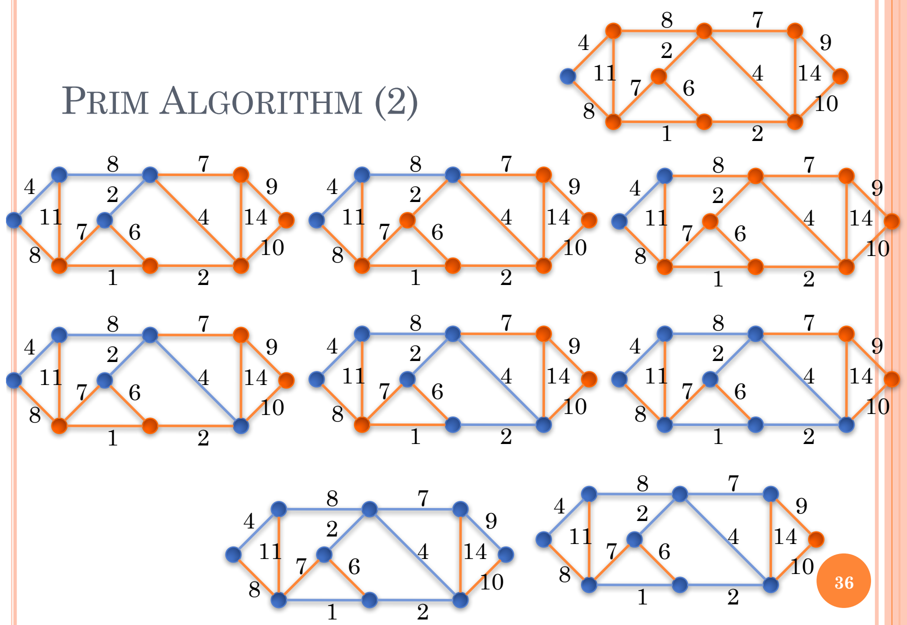
Implementation using:

- Nodes in a min-priority queue w.r.t. $key(v)$=min weight of an arc connecting v to a node of the main connected component; ∞ if it does not exist
- If the priority queue is a heap ➔ Complexity: $O(m \log n)$
- If the priority queue is a Fibonacci heap

  ➔ Complexity: $O(m+n \log n)$

[Ahuja, Magnanti & Orlin '93]

PRIM ALGORITHM (2)



BORUVKA ALGORITHM (1)

Among those connecting C_i to another component, the one with minimum weight

(purpose: an efficient electrical coverage of Moravia)

Hipothesis: each arc has a distinct weight

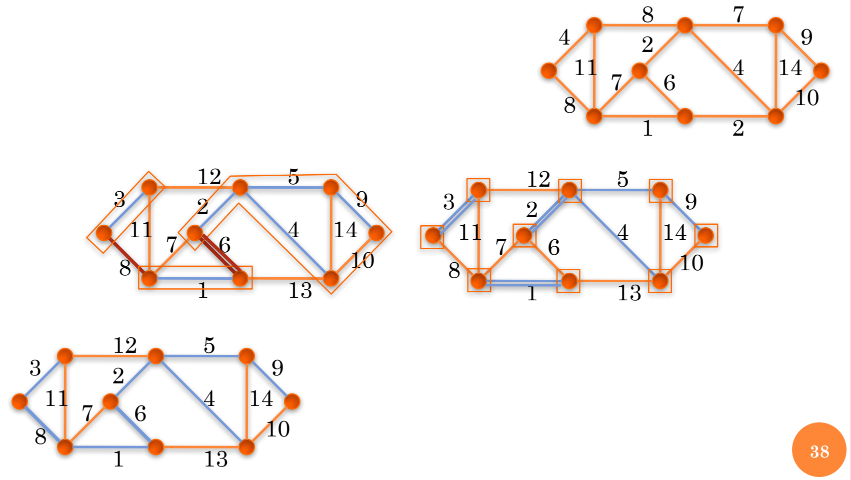
- A=empty set
  While A is not a MST
  for each connected component C_i of G_A
  find a safe arc e_i w.r.t. C_i
  $A=A \cup \{e_i\}$

Trick: handle many arcs (exactly log of the # of connected components) during the same loop

Impossible to introduce cycles, thanks to the hipothesis!

Complexity: $O(m \log n)$

BORUVKA ALGORITHM (2)



OTHER ALGORITHMS (1)

- [Friedman & Willard '94] Linear time algorithm, but it assumes the edges are already sorted w.r.t. their weight. Not used in practice, as the asymptotic notation hides a huge constant.
- [Matsui '95] Linear time algorithm for planar graphs (possible lesson)

OTHER ALGORITHMS (2)

- [Frederickson '85, Eppstein '94] Given a graph and its MST, it is even interesting to find a new MST after that the original graph has been slightly modified. It can be performed in average time $O(\log n)$
- Only $O(n+m)$ time is necessary to verify whether a given spanning tree is minimum.

Another Application

- A telecommunication company wants to lay cable to a new neighborhood.
- It is constrained to bury the cable only along certain paths (e.g. along roads).
- Model as a (not geometrical) graph:
  - nodes: represent points
  - edges: represent those paths
  - (edge) weight: cost of adding cable on that path.

    Note 1. some of those paths might be more expensive, because they are longer, or require the cable to be buried deeper

    Note 2. there is no requirement for edge lengths to obey normal rules of geometry such as the triangle inequality.
  - A *minimum spanning tree* for that graph would be a subset of those paths that has no cycles but still connects to every house with the lowest total cost, thus would represent the least expensive path for laying the cable.

Again on Minimum Energy Broadcast

Heuristics (1)

In [Wieselthier, Nguyen, Ephremides, 00]: three heuristics all based on the greedy technique:

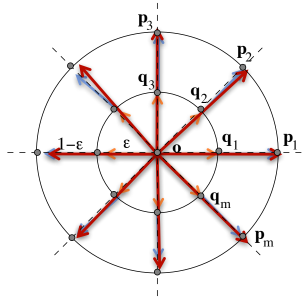
- SPT (spanning path tree): it runs Dijkstra algorithm to get the minimum path tree, then it directs the edges of the tree from the root to the leaves.
- BAIP (Broadcast Average Incremental Power): it is a modification of the Dijkstra algorithm based on the nodes (i.e. a new node is added to the tree on the basis of its minimum average cost).
- MST (min spanning tree): it runs Prim algorithm to get a MST, then it directs the edges of the tree from the root to the leaves.

Heuristics (2)

Greedy is not always good

Greedy is not always good [Wan, Calinescu, Li, Frieder '02]:

- SPT: it runs Dijkstra algorithm to get the minimum path tree, then it directs the edges of the tree from the root to the leaves



(letα=2)

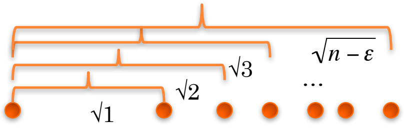
- SPT outputs a tree with total energy:

  $\varepsilon^2+n/2(1-\varepsilon)^2$
- If the root transmits with radius 1 the energy is 1
- When $\varepsilon \to 0$ SPT is far $n/2$ from the optimal solution.

HEURISTICS (3)
GREEDY IS NOT ALWAYS GOOD

- BAIP (Broadcast Average Incremental Power): it is a modification of the Dijkstra algorithm based on the nodes: a new node is added to the tree on the basis of the min average cost=energy increasing/# of added nodes.
- It has been designed to solve the problems of SPT.

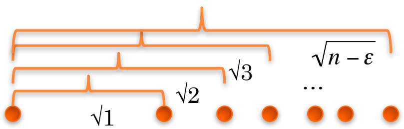
HEURISTICS (4)
GREEDY IS NOT ALWAYS GOOD



(let α=2):

- The min transmission power of the source to reach k receiving nodes is $\sqrt{k^2}=k$ and thus the average power efficiency is $k/k=1$
- On the other hand, the min transmission power of the source to reach all receiving nodes is $(\sqrt{n-\varepsilon})^2=n-\varepsilon$ and thus the average power efficiency is $(n-\varepsilon)/n=1-\varepsilon/n$...

HEURISTICS (5)
GREEDY IS NOT ALWAYS GOOD



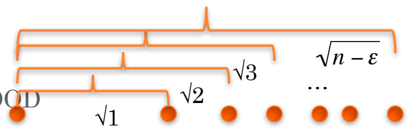
- BAIP will let the source to transmit at power $\sqrt{n-\varepsilon}$ to reach all nodes in a single step.
- However, the opt. routing is a path consisting of all nodes from left to right. Its min power is:

$$\sum_{i=1}^{n-1}(\sqrt{i}-\sqrt{i-1})^2+(\sqrt{n-\varepsilon}-\sqrt{n-1})^2<\sum_{i=1}^{n}(\sqrt{i}-\sqrt{i-1})^2=$$

$$\sum_{i=1}^{n}(\sqrt{i}-\sqrt{i-1})^2\frac{(\sqrt{i}+\sqrt{i-1})^2}{(\sqrt{i}+\sqrt{i-1})^2}=\sum_{i=1}^{n}\frac{((\sqrt{i}-\sqrt{i-1})(\sqrt{i}+\sqrt{i-1}))^2}{(\sqrt{i}+\sqrt{i-1})^2}=$$

$$=\sum_{i=1}^{n}\frac{(i-(i-1))^2}{(\sqrt{i}+\sqrt{i-1})^2}=\sum_{i=1}^{n}\frac{1}{(\sqrt{i}+\sqrt{i-1})^2}=1+\sum_{i=2}^{n}\frac{1}{(\sqrt{i}+\sqrt{i-1})^2}\le$$

HEURISTICS (6)
GREEDY IS NOT ALWAYS GOOD



(computation of the performance ratio of BAIP – cntd)

$$\le 1+\sum_{i=2}^{n}\frac{1}{i+(i-1)+2\sqrt{i}\sqrt{i-1})}\le 1+\sum_{i=2}^{n}\frac{1}{2i-1+2(i-1)}\le$$

$$\le 1+\sum_{i=2}^{n}\frac{1}{2i-1+2(i-1)}=1+\sum_{i=2}^{n}\frac{1}{4i-3}\le 1+\sum_{i=2}^{n}\frac{1}{4(i-1)}\le$$

Substituting $i=j+1$:

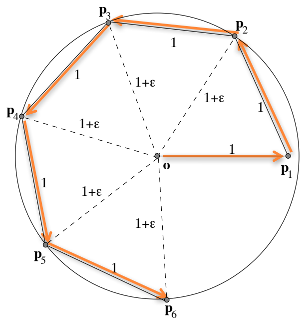
$$\le 1+\sum_{j=1}^{n-1}\frac{1}{4j}\le 1+\frac{1}{4}\sum_{j=1}^{n-1}\frac{1}{j}\le 1+\frac{1}{4}(\ln(n-1)+1)=\frac{\ln(n-1)+5}{4}$$

Thus the approx ratio of BAIP is at least:

$$\frac{n-\varepsilon}{\frac{\ln(n-1)+5}{4}}\to(\varepsilon\to 0)\frac{4n}{\ln(n-1)+5}=\frac{4n}{\ln n}+o(1)$$

Heuristics (7)
Greedy is not always good

MST: it runs Prim algorithm to get a MST, then it directs the edges of the tree from the root to the leaves



- Path $op_1 \ldots p_6$ is the unique MST, and its total energy is 6.
- On the other hand, the opt. routing is the star centered at o, whose energy is $(1+\varepsilon)^{\alpha}$.
- The approx. ratio converges to 6, as ε goes to 0.

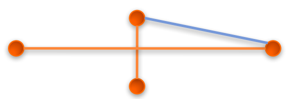
Heuristics (8)

- We have just shown a lower bound on the approximation ratio of MST.
- This ratio is a constant and an upper bound is 12.
- The proof involves complicated geometric arguments, and therefore we only sketch some of them:
  - …

Heuristics (9)

- Any pair of edges do not cross each other

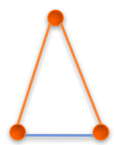
The blue edge is necessarily shorter than at least one of the two crossing edges



Heuristics (10)

(properties of the geometric MST – cntd)

- The angles between any two edges incident to a common node is at least $\pi/3$



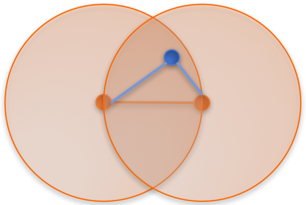
The blue edge is necessarily shorter than at least one of the two orange edges

Heuristics (11)

(properties of the geometric MST - cntd)

- The *lune* determined by each edge does not contain any other nodes.

The lune through points p_1 and p_2 is the intersection of the two open disks of radius $dist(p_1,p_2)$ centered at p_1 and p_2, respectively, hence an internal node would create a cycle

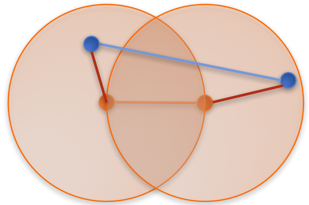


Heuristics (12)

(properties of the geometric MST – cntd)

- Let p_1p_2 be any edge. Then the two endpoints of any other edge are either both outside the open disk $D(p_1, dist(p_1, p_2))$ or both outside the open disk $D(p_2, dist(p_1, p_2))$

The red edges are added before than the blue edge because they are shorter. The blue edge would create a cycle.



Heuristics (13)

- Obs. The proof in [Wan, Calinescu, Li, Frieder '02] contains a small flaw that can be solved, arriving to an approximation ratio of 12,15 [Klasing, Navarra, Papadopoulos, Perennes '04]
- Indipendently, an approximation ratio of 20 has been stated in [Clementi, Crescenzi, Penna, Rossi, Vocca '01]
- Approx. ratio improved to 7,6 [Flammini, Klasing, Navarra, Perennes '04]
- Approx. ratio improved to 6,33 [Navarra '05]
- Optimal bound 6 [Ambüehl '05]

Heuristics (14)

- For realistic instances, experiments suggest that the tight approximation ratio is not 6 but 4 [Flammini, Navarra, Perennes '06] -> possible lesson