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

Prof. Tiziana Calamoneri
Network Algorithms
A.y. 2013/14

The Problem

The Problem (1)

- 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 betwee 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 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 envoirnment)

In the empty space $\alpha = 2$

- Hence, in order to have a communication from s to t, the power P_s is 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.
- …

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.

The Problem (7)

Given a source node s, the transmission graph is required to include a spanning tree rooted at s.

- 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.

The Problem (8)

- 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 (9)

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 (10)

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 withing a constant then even *MinSetCover* is. Contradiction.

The Problem (11)

Proof (cntd). Reduction:

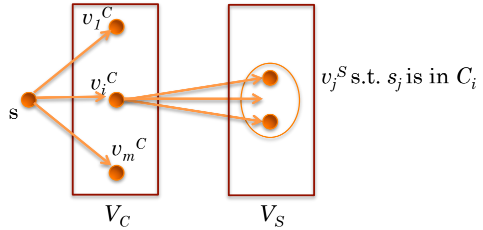
$x=(S,C)$ instance of *MinSetCover* with:

$S=\{s_1, s_2, \ldots, s_n\}$ 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 (12)

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 sol. is $|C'|+1$.

The Problem (15)

Proof (cntd).

…

Conversely, assume that r is a feasible sol. for y, (wlog $r(v)$ 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 sol. C' for x selecting all subsets C_i s.t. $r(v_i^C)=1$.

It holds that $|C'|=cost(r)-1$. ■

The Problem (16)

Note

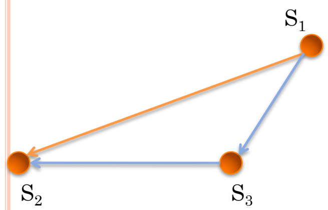
We proved that *Min Broadcast* is not approximable within a constant factor, but we have dealt with the general problem.

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 this special case…

The Problem (17)

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



  - S_1 needs to communicate with S_2
  - let $\alpha=2$
  - 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 (18)

- 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:

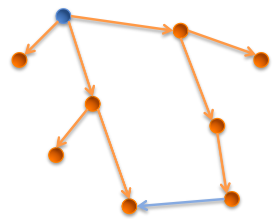
$$\cos t(D) = \sum_{i=1}^{l} r_i^{\alpha}$$

where r_i is the radius of D_i.

The Problem (19)

- 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, in view of some important properties…

The Problem (20)

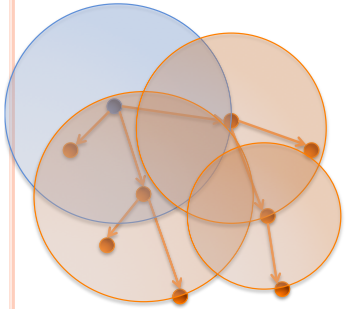


The set of connections used to perform a broadcast from s:

- 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 (21)

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



  - The energy wasted 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 (22)

- The Minimum Broadcast problem is NP-hard in its general version and it is neither approximable within $(1-\varepsilon)\Delta$ where Δ is the maximum degree of T and ε is an arbitrary constant
- Nothing is known for what concerns the geometric version.

The Problem (23)

- 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 (revision)

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. **QED**

- Obs. 2: 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: by contradiction…

Minimum Spanning Tree (2)

(proof - cntd)

- Assume by contradiction that MST A is not unique. So, there is another MST with equal weight, say B.
- Let e_1 be an edge that is in A but not in B. As B is a MST, $\{e_1\} \cup B$ must contain a cycle C and there is at least one edge e_2 in B that is not in A and lies on C.
- Assume the weight of e_1 is less than that of e_2.

  Replace e_2 with e_1 in B yields the spanning tree $\{e_1\} \cup B \setminus \{e_2\}$ which has a smaller weight compared to B.

  Contradiction, as we assumed B 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 $\{e_2\} \cup A \setminus \{e_1\}$ also leads to a contradiction.
- We conclude that the assumption that there is a further MST was false. **QED**

Minimum Spanning Tree (3)

Three classical algorithms:

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

Minimum Spanning Tree (4)

- 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 (5)

- 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 A is not a MST
    find a safe arc e w.r.t. A
    A=$A \cup e$

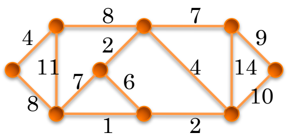
Among those connecting two different connected components in G_A, choose the one with minimum weight

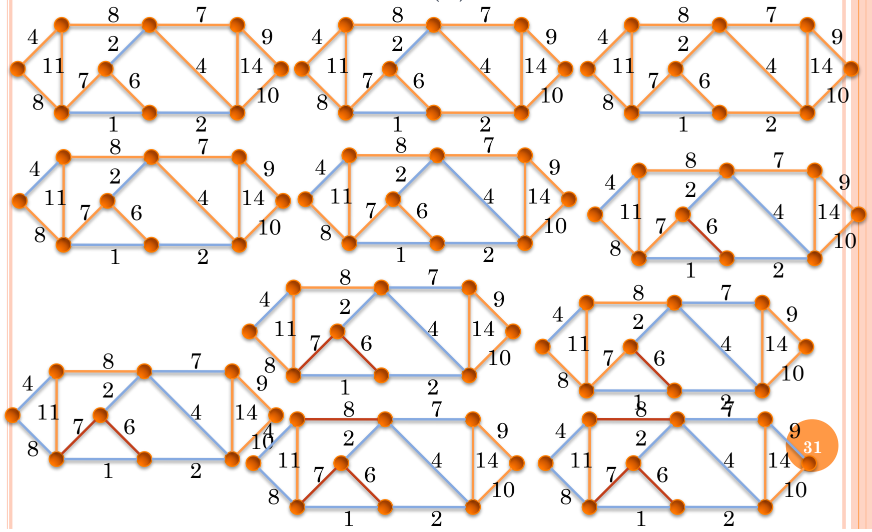
Implementation using:

- 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 $A$is not a MST
    find a safe arc e w.r.t. A
    A=$A \cup e$

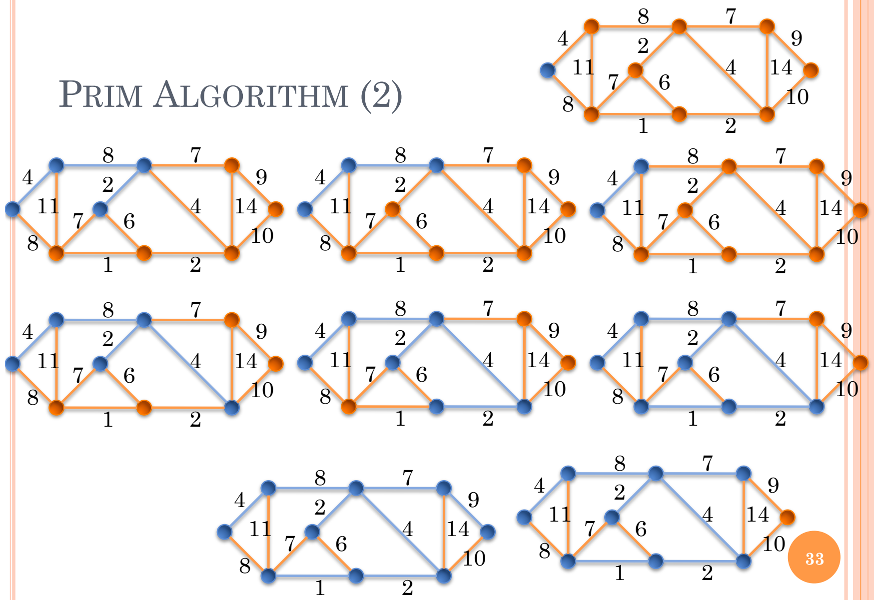
Among those connecting the main connected component with an isolated node, choose the one with minimum weight

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

Hipothesis: each arc has a distinct weight

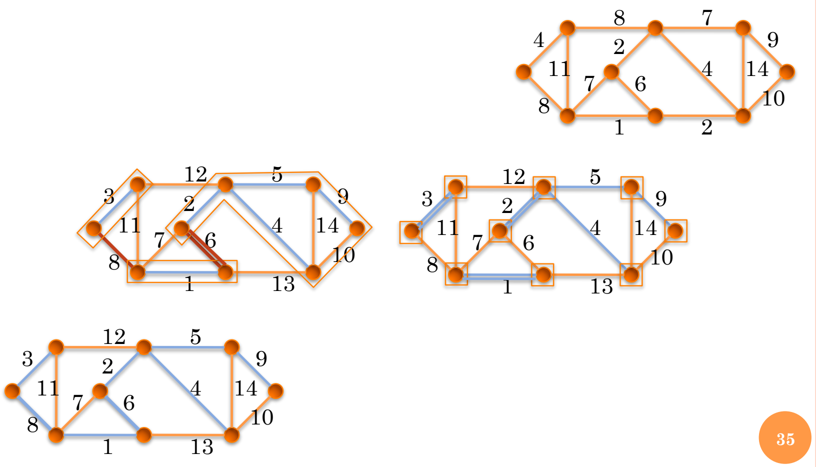
- *A*=empty set
  While *A* is not a MST
    foreach connected component C_i of A
      find a safe arc e_i w.r.t. C_i
      $A = A \cup \{e_i\}$

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 asimptotic notations hides a huge constant
- [Matsui '95] Linear time algorithm for planar graphs (possible lesson)

ALTRI ALGORITMI (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.

AGAIN ON MINIMUM ENERGY BROADCAST

HEURISTICS (1)

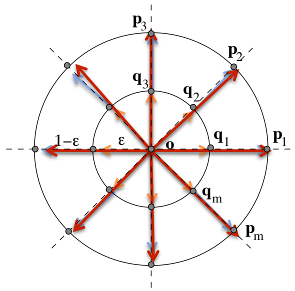
In [Wieselthier, Nguyen, Ephremides, 00]: three heuristics all based on the greedy technique:

- 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
- 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)

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



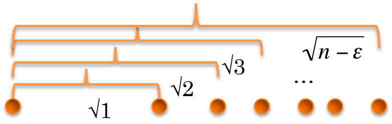
(let $\alpha=2$)

- 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 minimum average cost= energy increasing/# of added nodes
- It has been designed to solve the problems of SPT

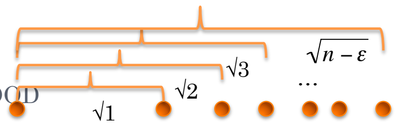
Heuristics (4)
Greedy is not always good



(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



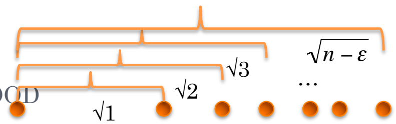
- BAIP will let the source to transmit at power $\sqrt{n-\varepsilon}$ to reach all nodes in a single step.
- However, the opt. routing is a path consisting of all nodes from left to right. Its min power is:

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

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

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

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 $i=j+1$:

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

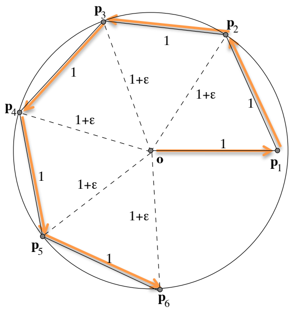
Thus the approx ratio of BAIP is at least:

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

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.

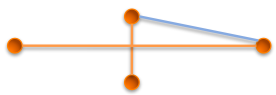
Heuristics (8)

- We have just proved 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

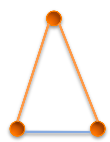
The blue edge is necessarily shorter than at least one of the two crossing edges



Heuristics (10)

(properties of the geometric MST - cntd)

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



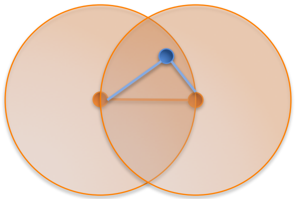
The blue edge is necessarily shorter than at least one of the two orange edges

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

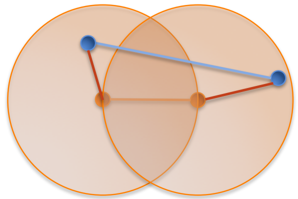


HEURISTICS (12)

(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