

A randomized approximation scheme for metric MAX-CUT

W. Fernandez de la Vega and Claire Kenyon

Journal of Computer and System Sciences, **63**, 531–541 (2001).

Speaker: Chuang-Chieh Lin

National Chung Cheng University

Outline

- Introduction
- The results
- The Euclidean case
 - Analysis of running time and correctness
- The general metric case (omitted)

Outline

- Introduction
- The results
- The Euclidean case
 - Analysis of running time and correctness
- The general metric case (omitted)

MAX-CUT

- MAX-CUT is the problem of finding a 2-partition of vertices of a (possibly weighted) graph which maximizes the number of edges (or sum of edge weights) across the partition.
- It has been known for a long time that this basic optimization problem is **NP**-hard but has a 2-approximation algorithm.

Background

- The best approximation ratio for the general case is 1.138 due to Goemans and Williamson [GW94, GW95].
- Unfortunately, there is not much room for improvement since this problem is Max-SNP-hard [PY91], and hence has no ϵ -approximation scheme if $\mathbf{P} \neq \mathbf{NP}$ [ALMSS98].

Background (cont'd)

- Thus one is led to consider restricted versions of MAX-CUT.

Background (cont'd)

- For dense *unweighted* graphs (i.e., graphs with $\Omega(n^2)$ edges), polynomial time approximation schemes were presented by Arora et al. [AKK95] and Vega [V96].
- In [VK98], dense *weighted* instances are dealt with.
- Related results also appear in [GGR98, FK99].

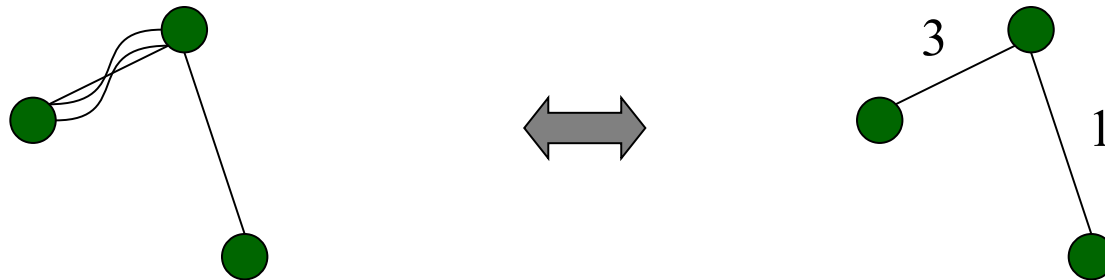
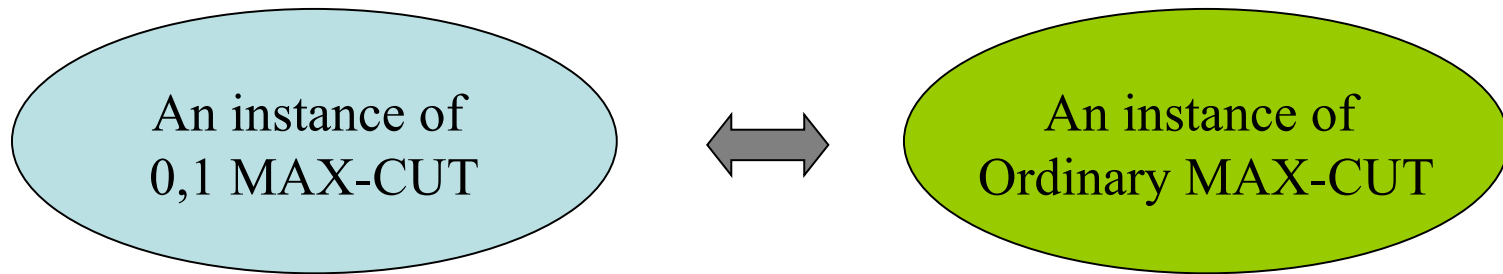
Remarks

- For an **unweighted** graph G , G is called *dense* if it has $\Omega(n^2)$ edges.
- For **weighted** graphs, “**dense**” refers usually to the 0, 1 case.
 - For a weighted graph G with 0,1 weights, G is called *dense* if its average degree at least cn where c is a constant and n denotes the number of vertices of G .

Remarks (cont'd)

- A PTAS for dense instances of MAX-CUT was found independently by Arora, Karger and Karpinski [AKK 95] and Fernandez de la Vega [V96].
- Actually, we will reduce metric MAX-CUT to an instance of ordinary MAX-CUT in which the maximum weight exceeds the average weight by at most a constant factor.
 - It is almost immediate to check that the algorithms for dense 0,1 MAX-CUT work for this case with trivial modifications.

Remarks (cont'd)



Remarks (cont'd)

- Thus in this paper, we say that a **weighted** graph G is *dense* if its maximum weight exceeds its average weight by at most a constant factor.
 - Consider the case that there are only few edges with very large weight yet the average weight of the graph is small.

Remarks (cont'd)

- What is the physical meaning that a **weighted** graph G is *dense* if its maximum weight exceeds its average weight by at most a constant factor (say “ λ ”)?
- Let \hat{w} be the average edge weight of G and let w^* be the maximum edge weight of G .

Remarks (cont'd)

- Let $G = (V, E)$ and for $e \in V \times V$, we denote the weight of e by $w(e)$. We have

$$w^* < \lambda \hat{w} \Rightarrow w^* < \lambda \cdot \frac{\sum_{e \in V \times V} w(e)}{|V|^2}$$

- Thus the number of edges which have positive weights is at least

$$\frac{\sum_{e \in V \times V} w(e)}{w^*} > \frac{1}{\lambda} \cdot |V|^2.$$

Remarks (cont'd)

- In this paper, we focus on metric instances of MAX-CUT.
- That is, the vertices correspond to points in **metric space**, the graph is the **complete** graph, and edge $\{x, y\}$ has a weight equal to the **distance** between x and y .
- Throughout the paper, we denote by $d(x, y)$ the distance between two points x and y . X is our set of n points. **MAX-CUT(X)** denotes the value of an optimum cut of X .

Outline

- Introduction
- **The results**
- The Euclidean case
 - Analysis of running time and correctness
- The general metric case (omitted)

The results

- Metric MAX-CUT is **NP**-complete.
- Metric MAX-CUT has a (randomized) polynomial time approximation scheme.

The results

- Metric MAX-CUT is **NP**-complete.
- Metric MAX-CUT has a (randomized) polynomial time approximation scheme.

These results are stated in the following theorems.

Theorem 1 [due to Luca Trevisan]

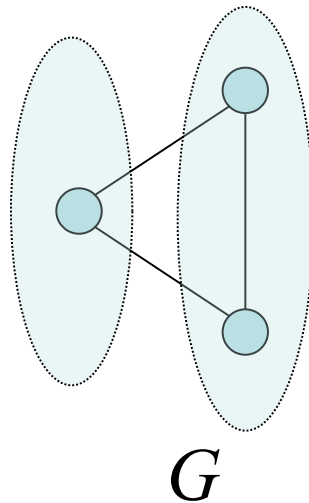
- Metric MAX-CUT is **NP**-hard.

Theorem 2

- Metric MAX-CUT has a (randomized) polynomial time approximation scheme.
 - That is, for any given $\varepsilon > 0$, there is a randomized algorithm which takes as input a discrete metric space given by its distance matrix, runs in time polynomial in the size of the space, and output a bipartition whose value is at least $(1 - \varepsilon)$ times the value of the maximum cut.

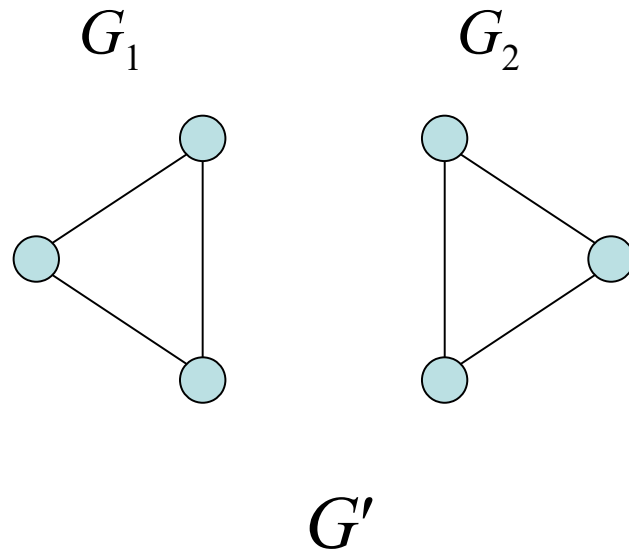
Proof of Theorem 1

- The proof is a reduction from MAX-CUT.
- Consider an instance G of MAX-CUT with n vertices.



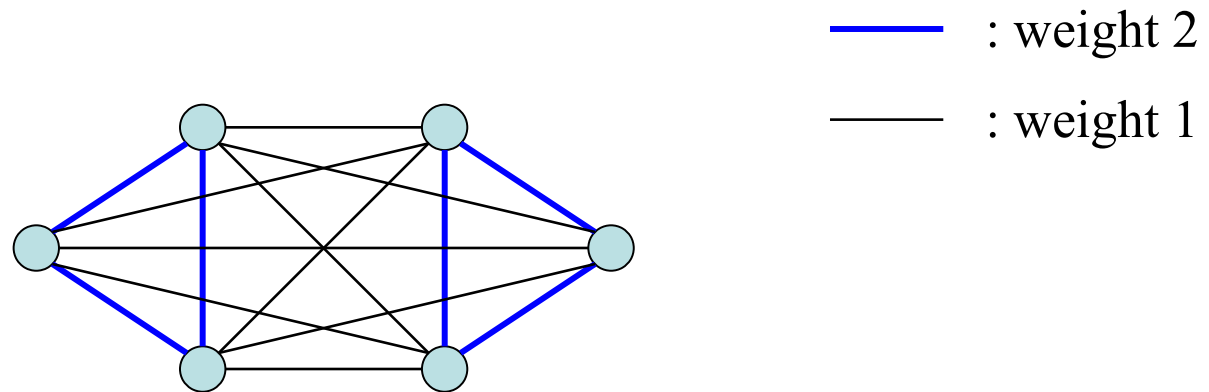
Proof of Theorem 1 (cont'd)

- Create a new graph G' with $2n$ vertices by taking two independent copies of G_1 and G_2 of G .



Proof of Theorem 1 (cont'd)

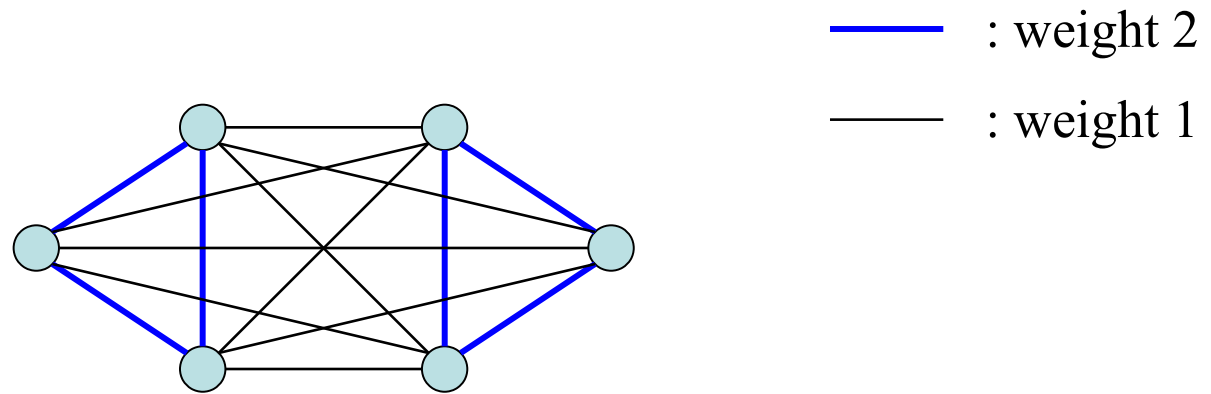
- Create a new weighted complete graph H with $2n$ vertices by giving weight 2 to every edge which was presented in G' and weight 1 to all other edges.



H

Proof of Theorem 1 (cont'd)

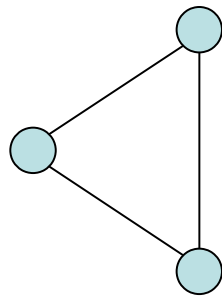
- H now is a metric graph.



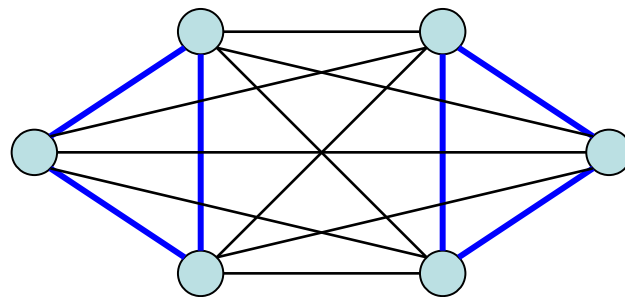
H

Proof of Theorem 1 (cont'd)

- It is **easy** to see that maximum cuts of H correspond to taking a maximum cut (A, B) of G_1 and the complementary maximum cut (B, A) of G_2 .



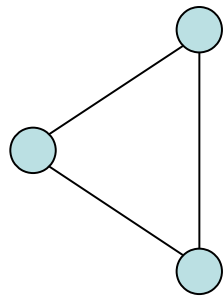
G



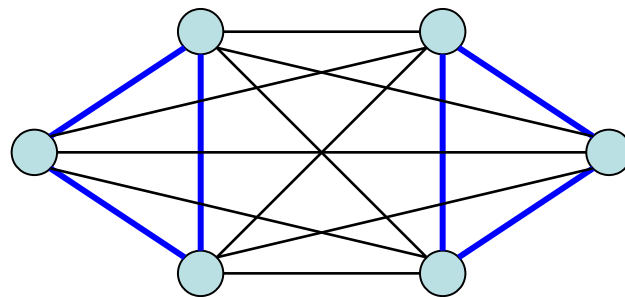
H

Proof of Theorem 1 (cont'd)

- If the maximum cut of G has value v , then the maximum cut of G' has value $2v$ and the maximum cut of H has value $2v + n^2$.



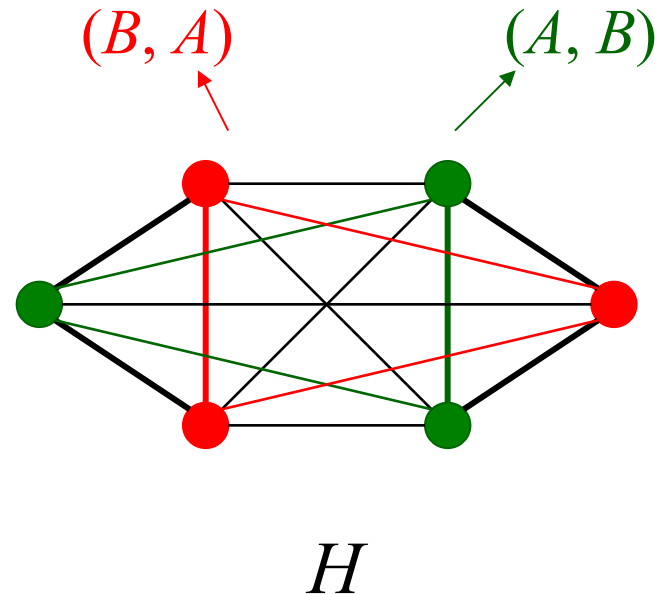
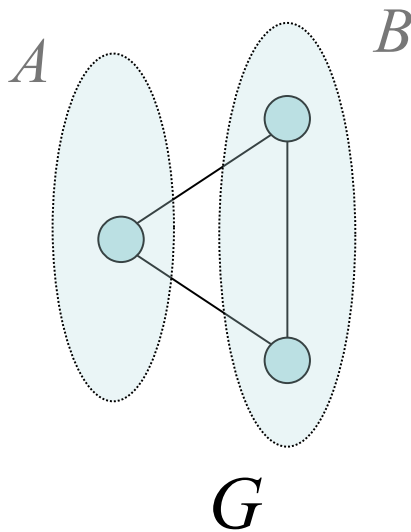
G



H

Proof of Theorem 1 (cont'd)

- If the maximum cut of G has value v , then the maximum cut of G' has value $2v$ and the maximum cut of H has value $2v + n^2$.



Proof ideas and techniques

- In the Euclidean case and in small dimension, we have a different algorithm which is a **PTAS**. It is based on:
 - changing coordinates by moving the origin to the center of gravity of the point set,
 - using polar coordinates, suitable rounding to simplify the point set, and
 - using brute force to solve the simplified instance.

Proof ideas and techniques (cont'd)

- In the general metric case, we will obtain our approximation theorem as a consequence of the following reduction (i.e., Theorem 3).

Theorem 3

- Approximating Metric MAX-CUT reduces to approximating Dense MAX-CUT.

Proof ideas and techniques (cont'd)

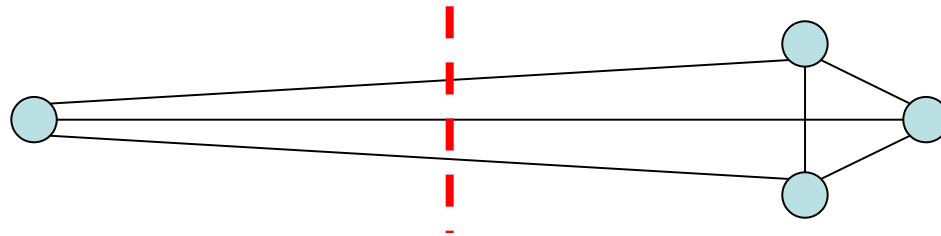
- We will reduce metric MAX-CUT to an instance of ordinary (i.e., weighted) MAX-CUT which is dense weighted.
 - Recall that an instance of ordinary MAX-CUT is dense if its maximum weight exceeds its average weight by at most a constant factor.

The problem of a naïve adaptation of the algorithm for dense weighted graphs

- We have a naïve algorithm for dense 0-1 weighted graphs.
 - The first step consists in taking a constant size sample of the vertices.
(vertices with large degree??)
 - In the dense graphs setting, all significant vertices have the same number of edges (up to a constant factor), hence contribute the same number of edges to MAX-CUT (up to a constant factor).
 - Hence a sample of constant size is sufficient to get a fairly good picture of the whole graph.

The problem of a naïve adaptation of the algorithm for dense weighted graphs (cont'd)

- In the metric setting, the situation is completely different.
- Outliers (points which are really far from the rest of the set) may contribute much more to MAX-CUT than other points.'



The problem of a naïve adaptation of the algorithm for dense weighted graphs (cont'd)

- A constant size sample is bound to miss the few outliers, and examining the sample will not give good information about MAX-CUT.
- Thus a naïve adaptation of the sense graph algorithm to metric MAX-CUT is doomed.

The problem of a naïve adaptation of the algorithm for dense weighted graphs (cont'd)

- The solution to this problem is simple:
 - The contribution of a point x to **(metric)** MAX-CUT is roughly proportional to the average distance from x to the rest of the set.
 - Thus in the metric setting, one should NOT use a uniform sample of the set of points, but a **biased sample**, where the probability of taking x in the sample is proportional to the average distance from x to the rest of the set.

A fatal error in the paper??

- Given an arbitrary positive integer n and a real number x . If x is at least $(n-1)/2$, then we can obtain $\lfloor x \rfloor \geq x(1 - 1/n)$.
 - Is this always true??
 - For example, let $n = 101$, we have $x \geq (101 - 1)/2 = 50$. If we pick $x = 50.9$, we have $\lfloor x \rfloor = 50$ and $x(1 - 1/n) = 50.3960396$.
 - So we have a counterexample.

Outline

- Introduction
- The results
- **The Euclidean case**
 - Analysis of running time and correctness
- The general metric case (omitted)

The Euclidean case

- In the presentation today, let us consider the Euclidean case (the second part of the paper).
 - The third part is the general metric case.
- When the dimension of the underlying space is fixed, a PTAS for MAX-CUT can easily be obtained.
- Here, we describe the PTAS for MAX-CUT in the plane. The cases of higher dimension are completely similar (replacing polar coordinates by spherical coordinates).

The algorithm for the Euclidean case

Input: A set X of n points in the Euclidean plane.

1. Scale the problem so that the average interpoint distance is equal to 1.
2. Compute $g = \sum_{x \in X} x/n$, the center of gravity of X .
3. If $(d(x, g), \theta(x))$ denote the polar coordinates of x with respect to g , define the domains:

$$D_{r,k} = \left\{ x \in \mathbb{R}^2 : \begin{array}{l} \epsilon(1 + \epsilon)^{r-1} \leq d(x, g) < \epsilon(1 + \epsilon)^r \text{ and} \\ k\pi\epsilon \leq \theta(x) < (k + 1)\pi\epsilon \end{array} \right\},$$

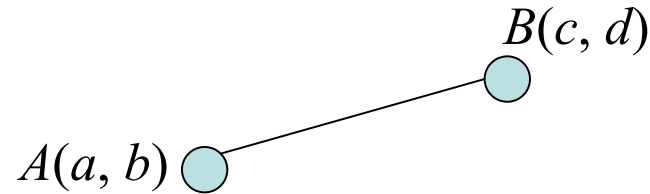
where $r \geq 1$ and $0 \leq k < 2\pi/\epsilon$. Let

$$D_0 = \{x \in \mathbb{R}^2 : d(x, g) < \epsilon\}.$$

4. Construct a point (multi)set X' obtained by replacing each element of $X \cap D_{r,k}$ by $y_{r,k}$, the point with polar coordinates $d(y_{r,k}, g) = \varepsilon(1 + \varepsilon)^{r-1}$ and $\theta(y_{r,k}) = k\pi\varepsilon$. Hence $y_{r,k}$ has multiplicity equal to the number of points of $X \cap D_{r,k}$. Moreover, each element of $X \cap D_0$ is replaced by g .
5. Solve MAX-CUT on X' by doing exhaustive search on the family of all cuts such that points which have the same coordinates are placed on the same side of the cut.

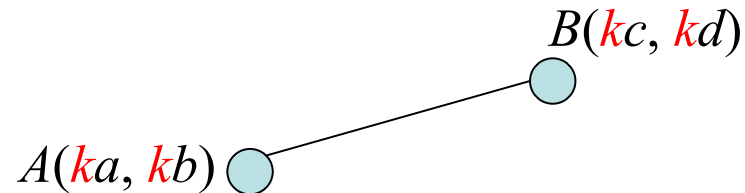
Output: the corresponding cut of X .

How to “scale the problem” so that the average interpoint distance is equal to 1?

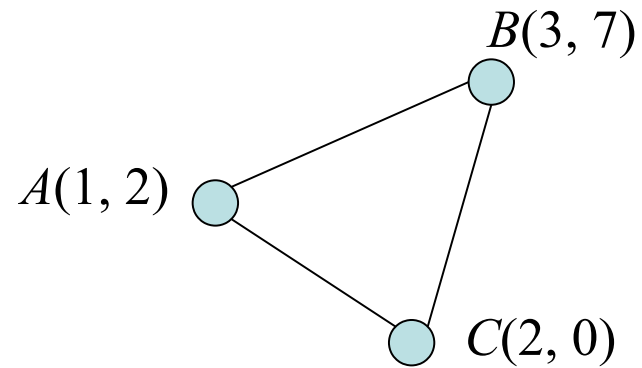


$$d(A, B) = \sqrt{(a - c)^2 + (b - d)^2}.$$

If we want to modify $d(A, B)$ to $k \cdot d(A, B)$, we can change the coordinates of A and B to be



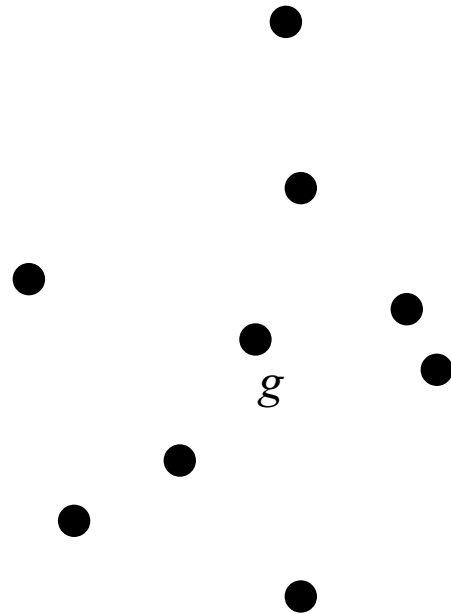
How to compute the center of gravity of X ?



We have $g = ((1+3+2)/3, (2+7+0)/3)$
 $= (2, 3)$

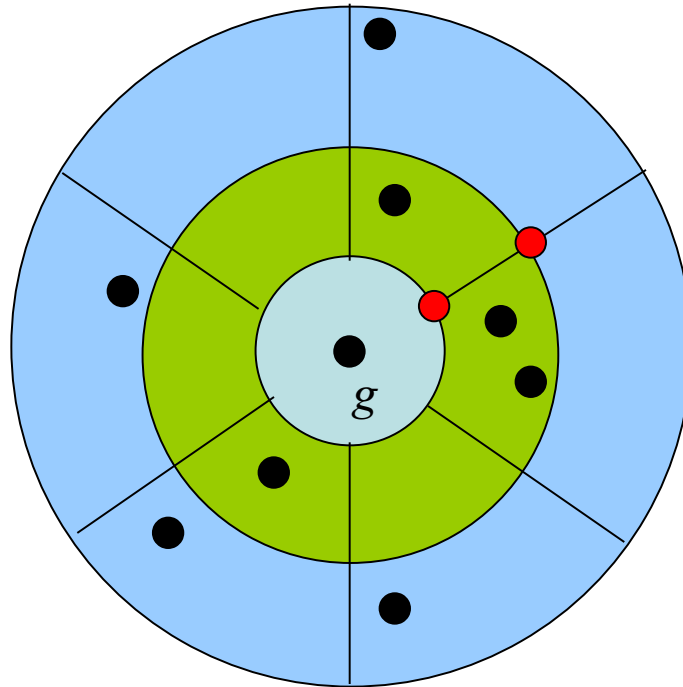
Let us see some illustration to make clear the idea of the algorithm.

● : point of X



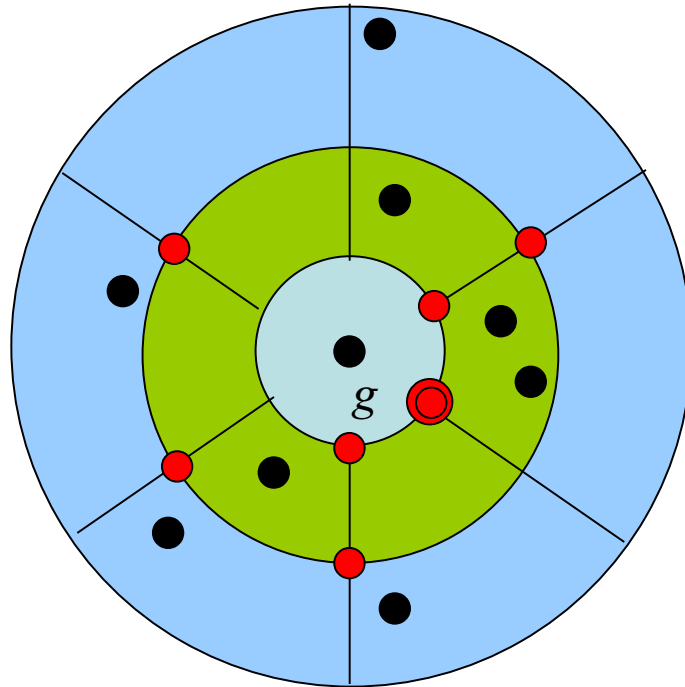
● : point of X

● : point of X'



● : point of X

● : point of X'



Outline

- Introduction
- The results
- The Euclidean case
 - Analysis of running time and correctness
- The general metric case (omitted)

Analysis of the running time

- The running time of the algorithm is clearly poly-nomial, with possible exception of the exhaustive search. The running time of the exhaustive search is exponential in the number of non-empty domains $D_{r,k}$.
- The following lemma will help us analyze this quantity.

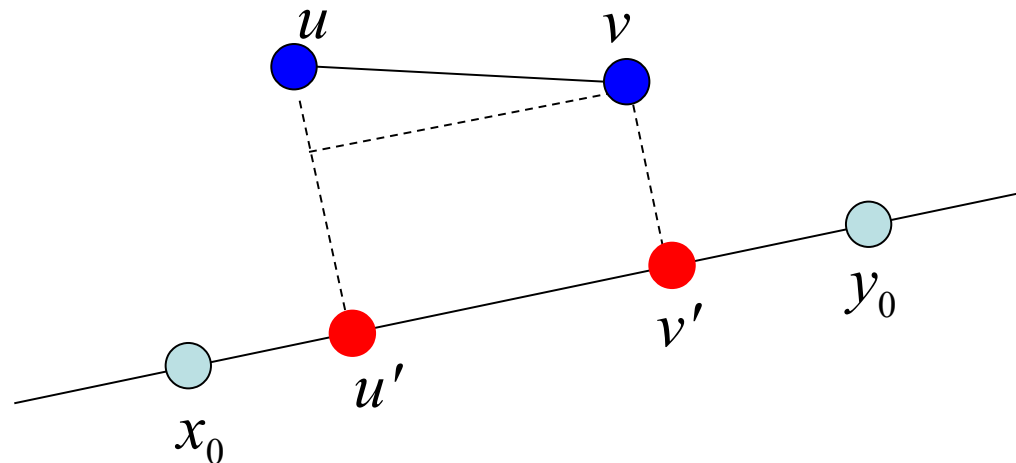
Lemma 1

- Let $d_{\max} = \max_{x, y \in X} d(x, y)$ denote the diameter of the point set. Then the sum of all interpoint distances is at least

$$\sum_{\{x, y\} \subset X} d(x, y) \geq (n - 1)d_{\max}.$$

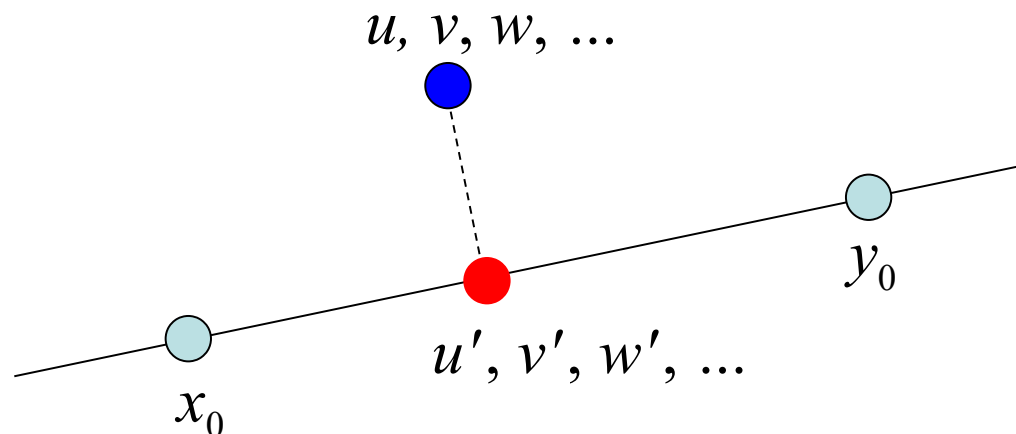
Proof of Lemma 1

- Let x_0, y_0 be such that $d(x_0, y_0) = d_{\max}$ is maximum.
- Let X' be obtained from X by orthogonal projection onto line (x_0, y_0) .
 - This can only decrease distances while keeping d_{\max} unchanged.



Proof of Lemma 1 (cont'd)

- By definition of d_{\max} , all points of X' other than x_0 and y_0 must lie between x_0 and y_0 .
- And it is easy to see that the sum of all distances is minimized **when all the points of $X' \setminus \{x_0, y_0\}$ are equal.**



Proof of Lemma 1 (cont'd)

- Then the sum of all interpoint distances is exactly $(n-1)d_{\max}$, hence the lemma.
 - Since $(n-2) \cdot d(x_0, y_0) + d(x_0, y_0) = (n-1) d_{\max}$. ■

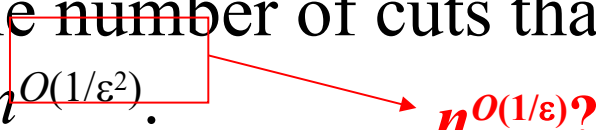
Corollary 1

- If the average interpoint distance of X is 1, then the diameter of X is at most $n/2$.
- *Proof:*

Since $\sum_{\{x,y\} \subset X} d(x,y) = \frac{n(n-1)}{2} \geq (n-1)d_{\max}$,
we have $d_{\max} \leq n/2$.



Analysis of the running time (cont'd)

- Thus every point is at distance at most $n/2$ from g .
- If a domain $D_{r,k}$ contains points of X , it must be the case that $\varepsilon(1+\varepsilon)^{r-1} \leq n/2$.
 - So we have $r \leq 1 + \log_{1+\varepsilon}(n/2\varepsilon)$.
- The total number of non-empty domains, including D_0 , is then at most $1 + (1 + \log_{1+\varepsilon}(n/2\varepsilon))2\pi/\varepsilon$.
- Thus the number of cuts that needs to be examined is at most $n^{O(1/\varepsilon^2)}$.
 $n^{O(1/\varepsilon)}?$

Analysis of correctness

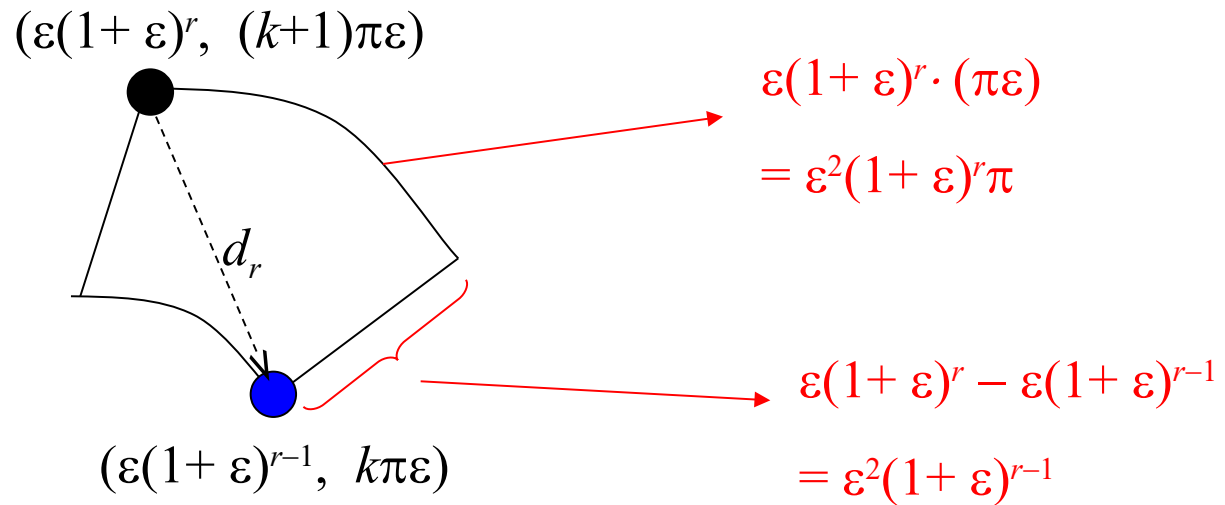
- Next we will show that the cut output by the algorithm is close to optimal.
- First, it is easy to see that if x and y are two points of X' which have the same coordinates, then there is a maximum cut of X' which places them on the same side of the cut.
 - Otherwise, moving either x or y to the other side would improve the cut.
- Thus the algorithm does indeed compute MAX-CUT(X').

Analysis of correctness (cont'd)

- The main question is thus comparing $\text{MAX-CUT}(X')$ to $\text{MAX-CUT}(X)$.
- The idea is that points do not move very far when going from X to X' .

Analysis of correctness (cont'd)

- In fact, if $x \in X \cap D_{r,k}$, then x is moved by at most the diameter d_r of $D_{r,k}$.



Analysis of correctness (cont'd)

- Thus

$$\begin{aligned}d_r &\leq \epsilon^2(1 + \epsilon)^{r-1} + \epsilon^2(1 + \epsilon)^r \pi \\ &\leq 5\epsilon^2(1 + \epsilon)^{r-1} \\ 7? & \leftarrow \\ &= 5\epsilon d(x, g).\end{aligned}$$

- On the other hand, if $x \in D_0$, then x is moved by at most ϵ .
- Clearly, moving one point x at distance δ from its original position does not change the value of the optimum cut by more than $\delta(n-1)$.

Analysis of correctness (cont'd)

- Thus we have

$$|\text{MAX-CUT}(X) - \text{MAX-CUT}(X)| \leq \epsilon n(n-1) + 5\epsilon(n-1) \sum_x d(x, g).$$

- Consider the following lemma.

Lemma 2

$$\sum_{x \in X} d(x, g) \leq \frac{n}{2}.$$

Proof:

- It is easy to see that

$$d(x, g) \leq \frac{1}{n} \sum_{y \in X} d(x, y).$$

- In one dimension this is clear.

Proof of Lemma 2 (cont'd)

$$d(x, g) \leq \frac{1}{n} \sum_{y \in X} d(x, y).$$

Let us consider the case in one dimension first.

- LHS = $d(x, g) = d\left(x, \frac{\sum_{y \in X} y}{n}\right)$
 $= x - \frac{\sum_{y \in X} y}{n}.$

Proof of Lemma 2 (cont'd)

$$d(x, g) \leq \frac{1}{n} \sum_{y \in X} d(x, y).$$

- RHS = $\frac{1}{n} \sum_{y \in X} d(x, y) = \sum_{y \in X} \frac{d(x, y)}{n}$
 $= \sum_{y \in X} d\left(\frac{x}{n}, \frac{y}{n}\right)$
 $= \sum_{y \in X} \left(\frac{x}{n} - \frac{y}{n}\right)$
 $= x - \frac{\sum_{y \in X} y}{n}.$

Proof of Lemma 2 (cont'd)

- In higher dimension it suffices to perform an orthogonal projection of X onto line (xg) , which does not affect the LHS and can only decrease the RHS.
- Then summing over all x yields the lemma (theorem).

$$d(x, g) \leq \frac{1}{n} \sum_{y \in X} d(x, y) \Rightarrow \sum_{x \in X} d(x, g) \leq \frac{n}{2}.$$



Analysis of correctness (cont'd)

- Using the lemma, we get

$$|\text{MAX-CUT}(X) - \text{MAX-CUT}(X)| \leq 4\epsilon n^2.$$

7/2 ?

- The expected value of a random cut of X is $n(n-1)/4$, and so

$$|\text{MAX-CUT}(X) - \text{MAX-CUT}(X)| \leq 17\epsilon \text{MAX-CUT}(X).$$

- Hence the algorithm is a PTAS.

It is safe only for $n \geq 6$.

By the way, why is the expected value of a random cut of X is $n(n-1)/4$?

Thank you.