Spanners: Quality Measures and Efficient Construction

Ferienakademie im Sarntal — Course 2
Distance Problems: Theory and Praxis

Dmitriy Traytel

Fakultät für Informatik
TU München

26. September 2010

Friedrich-Alexander-Universität Erlangen-Nürnberg

Technische Universität München

Universität Stuttgart

Outline

1. Introduction
   - Definition
   - Motivation
   - State-of-the-art
2. Purely additive (1,6)-spanner
   - Construction
   - Theoretical considerations
3. Multiplicative spanners
   - $(2k-1,0)$-spanner
   - $(k,k-1)$-spanner
   - Algorithmic properties
4. Conclusion

Spanners

Definition (Spanner)

Let $G = (V, E)$ be an unweighted graph. A subgraph $H = (V, E')$ of G is called (α, β)-spanner if for all $u, v \in V$ it holds:

$$\delta_H(u, v) \leq \alpha \delta_G(u, v) + \beta.$$

H is called additive if $\alpha = 1$ and purely additive if $\alpha = 1$ and $\beta \in \mathcal{O}(1)$.

Spanners

Definition (Spanner)

Let $G = (V, E)$ be an unweighted graph. A subgraph $H = (V, E')$ of G is called (α, β)-spanner if for all $u, v \in V$ it holds:

$$\delta_H(u, v) \leq \alpha\delta_G(u, v) + \beta.$$

H is called additive if $\alpha = 1$ and purely additive if $\alpha = 1$ and $\beta \in \mathcal{O}(1)$.

Notation

- $n = |V|$, $m = |E|$
- $\delta_G(u, v)$ - length of the shortest path between u and v in G
- $\delta_G(X, Y) = \min_{x \in X, y \in Y} \delta(x, y)$
- $\mathrm{diam}(D) = \max_{x, y \in D} \delta(x, y)$

So what are spanners good for?

- calculate approximate distances. . .

So what are spanners good for?

- calculate approximate distances…
- in a smaller graph constructed from the original graph
- with explicitly given approximation quality bounds

What makes a "good" spanner?

So what are spanners good for?

- calculate approximate distances...
- in a smaller graph constructed from the original graph
- with explicitly given approximation quality bounds

What makes a "good" spanner?

- approximation quality (best case: $\alpha = 1, \beta = 0$)
- size of the spanner (worst case: $\mathcal{O}(n^2)$)
- construction time (best case: linear)

The spanners zoo

(α, β)	Number of edges	Construction time
$(2k-1, 0)$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(m)$
$(k-1, 2k-\mathcal{O}(1))$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(mn^{1-1/k})$
$(k, k-1)$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(m)$
$(1+\epsilon, 4)$	$\mathcal{O}(\epsilon^{-1} n^{4/3})$	$\mathcal{O}(mn^{2/3})$
$(1, 6)$	$\mathcal{O}(n^{4/3})$	$\mathcal{O}(mn)$
$(1, 2)$	$\mathcal{O}(n^{3/2})$	$\mathcal{O}(m\sqrt{n})$

The spanners zoo

(α, β)	Number of edges	Construction time
$(2k-1, 0)$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(m)$
$(k-1, 2k-\mathcal{O}(1))$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(mn^{1-1/k})$
$(k, k-1)$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(m)$
$(1+\epsilon, 4)$	$\mathcal{O}(\epsilon^{-1}n^{4/3})$	$\mathcal{O}(mn^{2/3})$
$(1, 6)$	$\mathcal{O}(n^{4/3})$	$\mathcal{O}(mn)$
$(1, 2)$	$\mathcal{O}(n^{3/2})$	$\mathcal{O}(m\sqrt{n})$

The spanners zoo

(α, β)	Number of edges	Construction time
$(2k-1, 0)$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(m)$
$(k-1, 2k-\mathcal{O}(1))$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(mn^{1-1/k})$
$(k, k-1)$	$\mathcal{O}(n^{1+1/k})$	$\mathcal{O}(m)$
$(1+\epsilon, 4)$	$\mathcal{O}(\epsilon^{-1}n^{4/3})$	$\mathcal{O}(mn^{2/3})$
$(1, 6)$	$\mathcal{O}(n^{4/3})$	$\mathcal{O}(mn)$
$(1, 2)$	$\mathcal{O}(n^{3/2})$	$\mathcal{O}(m\sqrt{n})$

New Constructions of (α, β)-Spanners and Purely Additive Spanners (2005)




S.Baswana T.Kavitha
Indian Institute of Technology Kanpur

K.Mehlhorn
Max-Planck-Institut für Informatik

S.Pettie
University of Michigan

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \ldots, v \rangle$ - shortest path between u and v in G

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \ldots, v \rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \ldots, v\rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u,v) = \langle u, \ldots, v \rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices
- clustering $\mathcal{C}$ - set of disjoint clusters

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u,v) = \langle u,\ldots,v\rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices
- clustering $\mathcal{C}$ - set of disjoint clusters
- $\mathcal{C}(v)$ - cluster of clustering $\mathcal{C}$ that contains v

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \ldots, v \rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices
- clustering $\mathcal{C}$ - set of disjoint clusters
- $\mathcal{C}(v)$ - cluster of clustering $\mathcal{C}$ that contains v
- $\mathcal{C}(D) = \{\mathcal{C}(v) \mid v \in D\}$

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \ldots, v \rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices
- clustering $\mathcal{C}$ - set of disjoint clusters
- $\mathcal{C}(v)$ - cluster of clustering $\mathcal{C}$ that contains v
- $\mathcal{C}(D) = \{\mathcal{C}(v) \mid v \in D\}$
- $\mathcal{E}_{(V,E)}(X, Y) = (X \times Y) \cap E$

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \dots, v \rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices
- clustering $\mathcal{C}$ - set of disjoint clusters
- $\mathcal{C}(v)$ - cluster of clustering $\mathcal{C}$ that contains v
- $\mathcal{C}(D) = \{\mathcal{C}(v) \mid v \in D\}$
- $\mathcal{E}_{(V,E)}(X, Y) = (X \times Y) \cap E$
- $\mathcal{E}_{(V,E)}(x, Y) = (\{x\} \times Y) \cap E$

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \ldots, v \rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices
- clustering $\mathcal{C}$ - set of disjoint clusters
- $\mathcal{C}(v)$ - cluster of clustering $\mathcal{C}$ that contains v
- $\mathcal{C}(D) = \{\mathcal{C}(v) \mid v \in D\}$
- $\mathcal{E}_{(V,E)}(X, Y) = (X \times Y) \cap E$
- $\mathcal{E}_{(V,E)}(x, Y) = (\{x\} \times Y) \cap E$
- $\text{value}_H(D) = |\{\{C, C'\} \subseteq \mathcal{C}(D) \mid \delta_D(C, C') \leq \delta_H(C, C')\}|$

Some additional notation

- $\Gamma_G(v)$ - neighbourhood of the vertex v in the graph G
- $G[S]$ - subgraph of G induced by the vertex set S
- $P_G(u, v) = \langle u, \ldots, v \rangle$ - shortest path between u and v in G
- $\mathcal{P}_G$ - set of all $\binom{n}{2}$ shortest paths in G
- cluster C - set of vertices
- clustering $\mathcal{C}$ - set of disjoint clusters
- $\mathcal{C}(v)$ - cluster of clustering $\mathcal{C}$ that contains v
- $\mathcal{C}(D) = \{\mathcal{C}(v) \mid v \in D\}$
- $\mathcal{E}_{(V,E)}(X, Y) = (X \times Y) \cap E$
- $\mathcal{E}_{(V,E)}(x, Y) = (\{x\} \times Y) \cap E$
- $\text{value}_H(D) = |\{\{C, C'\} \subseteq \mathcal{C}(D) \mid \delta_D(C, C') \leq \delta_H(C, C')\}|$
- $\text{cost}_H(D) = |D \setminus H|$

Algorithm 1: $(1,6)$-spanner construction

input : Graph $G = (V, E)$
output: $(1,6)$-spanner H of G
begin

$S \leftarrow V$ `// phase 1 - clustering`

for $i \leftarrow 1$ **to** $n^{2/3}$ **do**

$v_i \leftarrow \arg\max_{x \in S} |\Gamma_{G[S]}(x)|$

$C_i \leftarrow \{v_i\} \cup \Gamma_{G[S]}(v_i)$

$S \leftarrow S \setminus C_i$

$E' \leftarrow \{(v_i, x_{v_i}) \mid 1 \leq i \leq n^{2/3}, x_{v_i} \in C_i\} \cup \{(u, w) \mid u \in S, w \in \Gamma_G(u)\}$

$H \leftarrow (V, E')$ `// phase 2 - path buying`

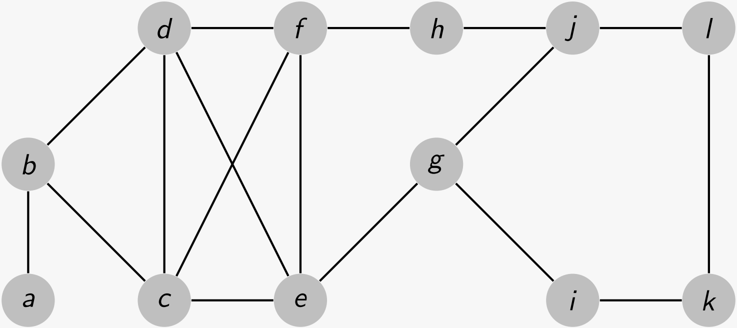
foreach *path* $P \in \mathcal{P}_G$ **do**

if $2 \cdot value_H(P) \geq cost_H(P)$ **then** $H \leftarrow H \cup P$

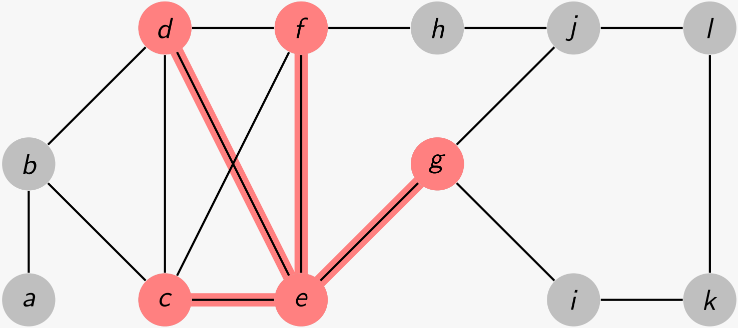
return H

end

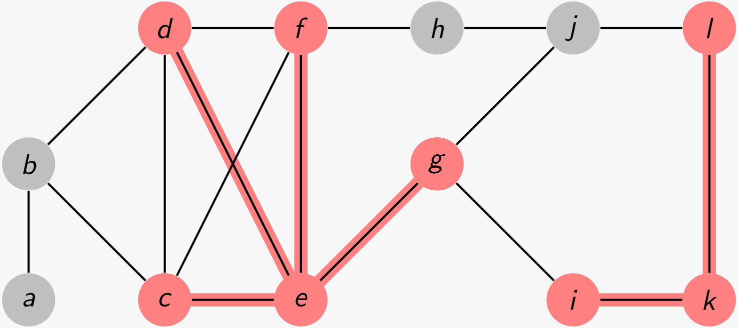
Let's build a $(1, 6)$-spanner



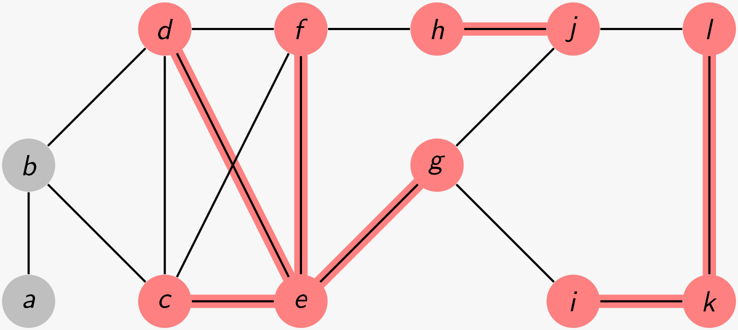
Clustering



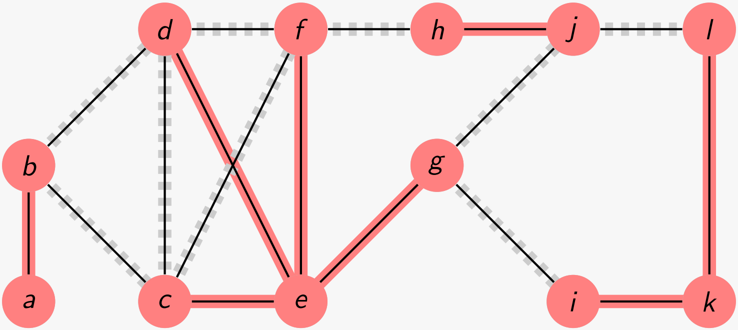
Clustering



Clustering

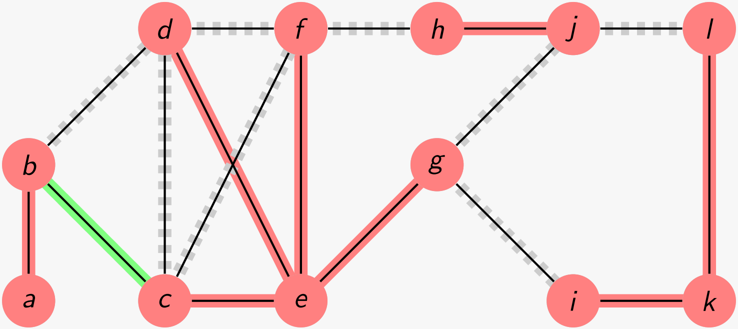


Graph resulting from clustering



$$\mathcal{C} = \{\{a, b\}, \{c, d, e, f, g\}, \{h, j\}, \{i, k, l\}\}$$

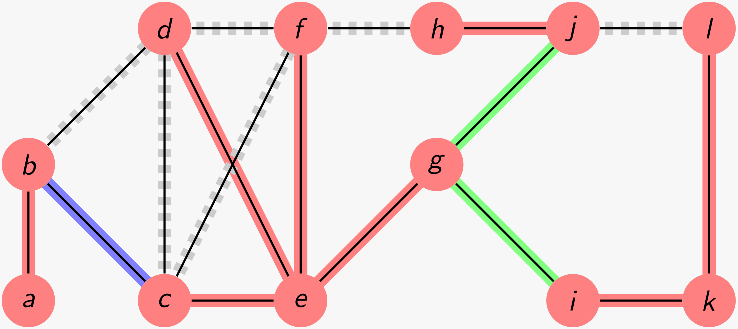
Some value-cost trade-off



$$\mathcal{C} = \{\{a, b\}, \{c, d, e, f, g\}, \{h, j\}, \{i, k, l\}\}$$

$\text{value}\langle b, c\rangle = 1$ $\text{cost}\langle b, c\rangle = 1$

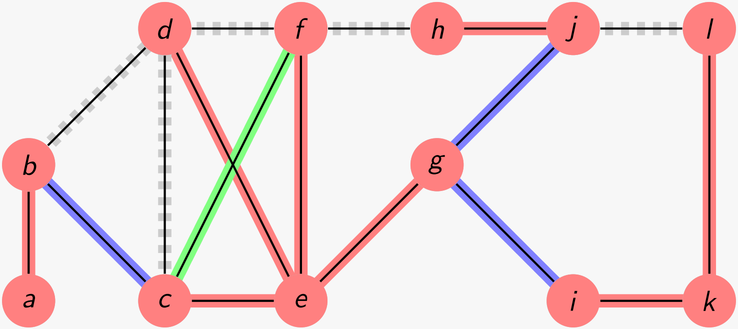
Some value-cost trade-off



$$\mathcal{C} = \{\{a, b\}, \{c, d, e, f, g\}, \{h, j\}, \{i, k, l\}\}$$

value$\langle i, g, j\rangle = 3$ cost$\langle i, g, j\rangle = 2$

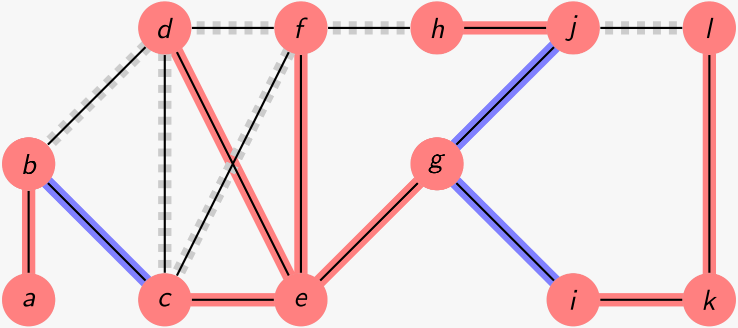
Some value-cost trade-off



$$\mathcal{C} = \{\{a, b\}, \{c, d, e, f, g\}, \{h, j\}, \{i, k, l\}\}$$

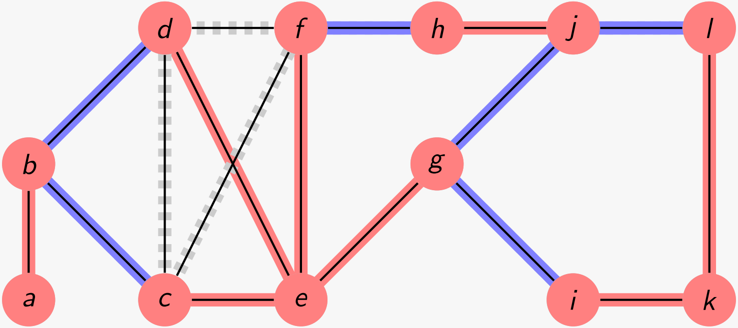
$\text{value}\langle c, f\rangle = 0$ $\text{cost}\langle c, f\rangle = 1$

Iterate. . .



$$\mathcal{C} = \{\{a, b\}, \{c, d, e, f, g\}, \{h, j\}, \{i, k, l\}\}$$

Resulting spanner



Why is the result always a $(1, 6)$-spanner?

Definition (Contented)

We call a subgraph H of G that contains the resulting graph of the clustering phase contented if for any two clustered vertices u_0, u_q there exists a shortest path $P_G(u_0, u_q)$ and a cluster $C \in \mathcal{C}(P)$ such that for $i \in \{0, q\}$:

$$\delta_H(\mathcal{C}(u_i), C) \leq \delta_P(\mathcal{C}(u_i), C)$$

Lemma

H is contented $\Rightarrow$ H is $(1, 6)$-spanner of G

Proof.

- Enough to consider only clustered vertices (Why?)

Why is the result always a $(1, 6)$-spanner?

Definition (Contented)

We call a subgraph H of G that contains the resulting graph of the clustering phase contented if for any two clustered vertices u_0, u_q there exists a shortest path $P_G(u_0, u_q)$ and a cluster $C \in \mathcal{C}(P)$ such that for $i \in \{0, q\}$:

$$\delta_H(\mathcal{C}(u_i), C) \leq \delta_P(\mathcal{C}(u_i), C)$$

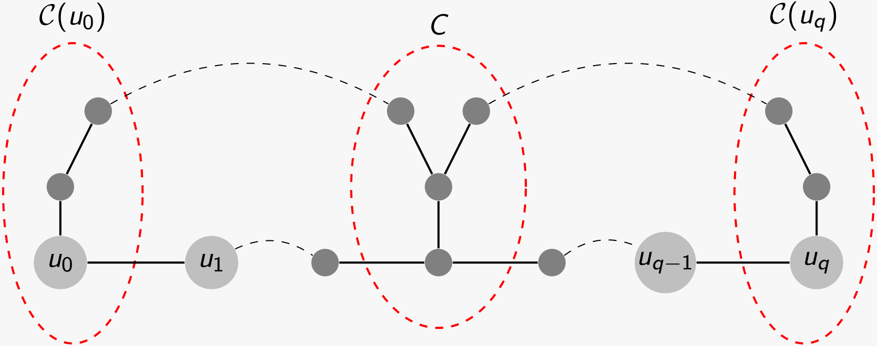
Lemma

H is contented $\Rightarrow$ H is $(1, 6)$-spanner of G

Proof.

- Enough to consider only clustered vertices (Why?)
- Hint: what happens to unclustered vertices at the end of the clustering phase?

Proof(continued).



$$
\begin{aligned}
\delta_H(u_0, u_q) &\leq \operatorname{diam}(\mathcal{C}(u_0)) + \delta_H(\mathcal{C}(u_0), C) + \operatorname{diam}(C) + \\
&\quad \delta_H(C, \mathcal{C}(u_q)) + \operatorname{diam}(\mathcal{C}(u_q)) \\
&\leq 2 + \delta_H(\mathcal{C}(u_0), C) + 2 + \delta_H(C, \mathcal{C}(u_q)) + 2 \\
&\overset{H \text{ contented}}{\leq} \delta_P(u_0, u_q) + 6
\end{aligned}
$$

□

Lemma (without proof)

For $P \in \mathcal{P}_G$ it holds: either $|\mathcal{C}(P)| = 1$ or $\exists$ subpath $P' \subseteq P$ s.t. $\mathcal{C}(P) = \mathcal{C}(P')$ and $cost_H(P') \leq 2|\mathcal{C}(P')| - 3$

Lemma

The given algorithm computes a contented subgraph of G.

Proof.

- Consider a shortest path $P = P_G(u_0, u_q)$

Lemma (without proof)

For $P \in \mathcal{P}_G$ it holds: either $|\mathcal{C}(P)| = 1$ or $\exists$ subpath $P' \subseteq P$ s.t. $\mathcal{C}(P) = \mathcal{C}(P')$ and $cost_H(P') \leq 2|\mathcal{C}(P')| - 3$

Lemma

The given algorithm computes a contented subgraph of G.

Proof.

- Consider a shortest path $P = P_G(u_0, u_q)$
- Interesting case: u_0 and u_q are clustered in different clusters in H

Lemma (without proof)

For $P \in \mathcal{P}_G$ it holds: either $|\mathcal{C}(P)| = 1$ or $\exists$ subpath $P' \subseteq P$ s.t. $\mathcal{C}(P) = \mathcal{C}(P')$ and $cost_H(P') \leq 2|\mathcal{C}(P')| - 3$

Lemma

The given algorithm computes a contented subgraph of G.

Proof.

- Consider a shortest path $P = P_G(u_0, u_q)$
- Interesting case: u_0 and u_q are clustered in different clusters in H
- Previous lemma provides a subpath P' of P.

Lemma (without proof)

For $P \in \mathcal{P}_G$ it holds: either $|\mathcal{C}(P)| = 1$ or $\exists$ subpath $P' \subseteq P$ s.t. $\mathcal{C}(P) = \mathcal{C}(P')$ and $cost_H(P') \leq 2|\mathcal{C}(P')| - 3$

Lemma

The given algorithm computes a contented subgraph of G.

Proof.

- Consider a shortest path $P = P_G(u_0, u_q)$
- Interesting case: u_0 and u_q are clustered in different clusters in H
- Previous lemma provides a subpath P' of P.
- Again the interesting case is when $P' \notin H$

Lemma (without proof)

For $P \in \mathcal{P}_G$ it holds: either $|\mathcal{C}(P)| = 1$ or $\exists$ subpath $P' \subseteq P$ s.t. $\mathcal{C}(P) = \mathcal{C}(P')$ and $cost_H(P') \leq 2|\mathcal{C}(P')| - 3$

Lemma

The given algorithm computes a contented subgraph of G.

Proof.

- Consider a shortest path $P = P_G(u_0, u_q)$
- Interesting case: u_0 and u_q are clustered in different clusters in H
- Previous lemma provides a subpath P' of P.
- Again the interesting case is when $P' \notin H$
- Have

$$2 \cdot \text{value}_H(P') < \text{cost}_H(P') \leq 2 \cdot |\mathcal{C}(P')| - 3$$

Proof(continued).

- Have $2 \cdot \text{value}_H(P') < \text{cost}_H(P') \leq 2 \cdot |\mathcal{C}(P')| - 3$

Proof(continued).

- Have $2 \cdot \text{value}_H(P') < \text{cost}_H(P') \leq 2 \cdot |\mathcal{C}(P')| - 3$
- Consider the set $A = \{\{C, C'\} \mid C \in \{\mathcal{C}(u_0), \mathcal{C}(u_q)\}, C' \in \mathcal{C}(P') \setminus \{C\}, \delta_{P'}(C, C') < \delta_H(C, C')\}$

Proof(continued).

- Have $2 \cdot \text{value}_H(P') < \text{cost}_H(P') \leq 2 \cdot |\mathcal{C}(P')| - 3$
- Consider the set $A = \{\{C, C'\} \mid$
  $C \in \{\mathcal{C}(u_0), \mathcal{C}(u_q)\}, C' \in \mathcal{C}(P') \setminus \{C\}, \delta_{P'}(C, C') < \delta_H(C, C')\}$
- A has at most $2 \cdot |\mathcal{C}(P)| - 3$ elements

Proof(continued).

- Have $2 \cdot \text{value}_H(P') < \text{cost}_H(P') \leq 2 \cdot |\mathcal{C}(P')| - 3$
- Consider the set $A = \{\{C, C'\} \mid$
  $C \in \{\mathcal{C}(u_0), \mathcal{C}(u_q)\}, C' \in \mathcal{C}(P') \setminus \{C\}, \delta_{P'}(C, C') < \delta_H(C, C')\}$
- A has at most $2 \cdot |\mathcal{C}(P)| - 3$ elements
- On the other hand: $|A| \leq \text{value}_H(P') \leq |\mathcal{C}(P)| - 2$

Proof(continued).

- Have $2 \cdot \text{value}_H(P') < \text{cost}_H(P') \leq 2 \cdot |\mathcal{C}(P')| - 3$
- Consider the set $A = \{\{C, C'\} \mid$
  $C \in \{\mathcal{C}(u_0), \mathcal{C}(u_q)\}, C' \in \mathcal{C}(P') \setminus \{C\}, \delta_{P'}(C, C') < \delta_H(C, C')\}$
- A has at most $2 \cdot |\mathcal{C}(P)| - 3$ elements
- On the other hand: $|A| \leq \text{value}_H(P') \leq |\mathcal{C}(P)| - 2$
- There must exist al least $|\mathcal{C}(P')| - 1$ pairs C, C' s.t. $\delta_{P'}(C, C') \geq \delta_H(C, C')$

Proof(continued).

- Have $2 \cdot \text{value}_H(P') < \text{cost}_H(P') \leq 2 \cdot |\mathcal{C}(P')| - 3$
- Consider the set $A = \{\{C, C'\} \mid C \in \{\mathcal{C}(u_0), \mathcal{C}(u_q)\}, C' \in \mathcal{C}(P') \setminus \{C\}, \delta_{P'}(C, C') < \delta_H(C, C')\}$
- A has at most $2 \cdot |\mathcal{C}(P)| - 3$ elements
- On the other hand: $|A| \leq \text{value}_H(P') \leq |\mathcal{C}(P)| - 2$
- There must exist al least $|\mathcal{C}(P')| - 1$ pairs C, C' s.t. $\delta_{P'}(C, C') \geq \delta_H(C, C')$
- By the pigeonhole principle there must exist a $C'' \in \mathcal{C}(P') = \mathcal{C}(P)$ s.t.

$$\delta_P(C(u_0), C'') = \delta_{P'}(C(u_o), C'') \geq \delta_H(C(u_0), C'')$$

$$\delta_P(C(u_q), C'') = \delta_{P'}(C(u_q), C'') \geq \delta_H(C(u_q), C'')$$

A word on size and time bounds

Spanner size

- $\mathcal{O}(n^{4/3})$
- first purely additive spanner of size $o(n^{3/2})$

A word on size and time bounds

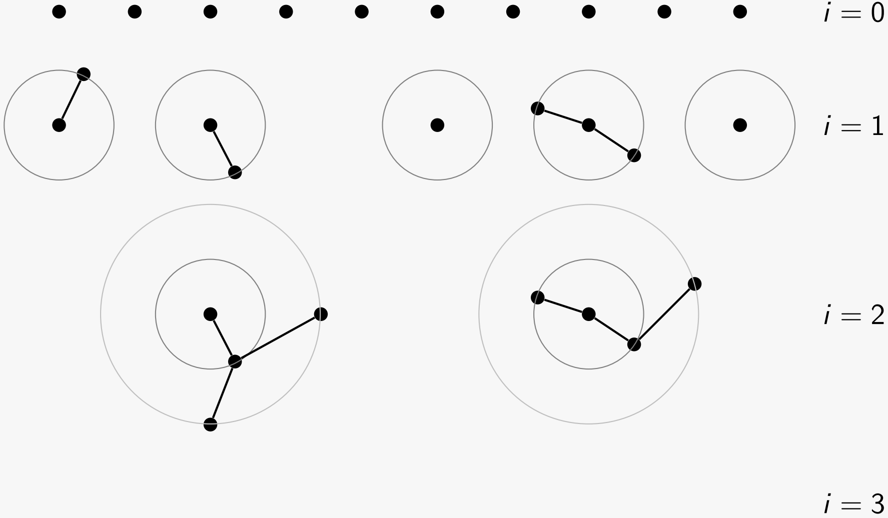
Spanner size

- $\mathcal{O}(n^{4/3})$
- first purely additive spanner of size $o(n^{3/2})$

Construction time

- $\mathcal{O}(nm)$
- Clustering linear in n when using priority queue for the "arg max"
- Path buying needs to compute all shortest paths and to know the distances in H
- Still some tweaks are needed to get the bound (e.g. using an upper bound function instead the value function)

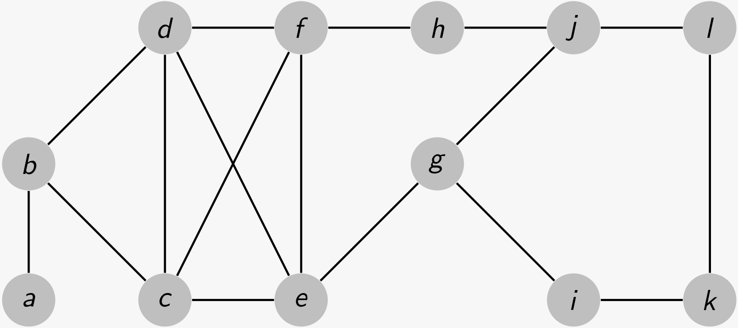
Take-home picture



Algorithm 2: Randomized $(2k-1,0)$-spanner construction

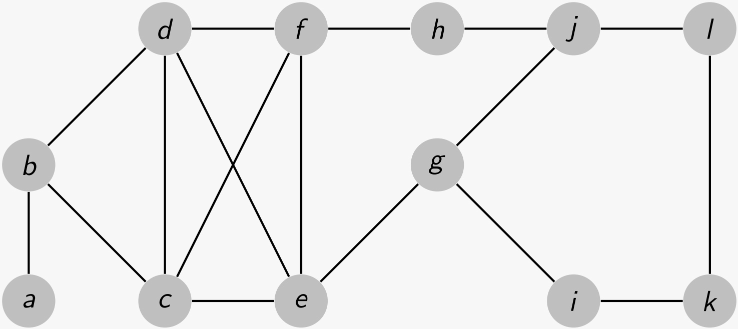
```
input  : Graph G = (V, E) and integer k
output: (2k − 1, 0)-spanner H of G
begin
    S ← ∅
    C_0 ← {{v} | v ∈ V}
    for i ← 1 to k do
        if i = k then  C_i ← ∅
        else  C_i ← {C ∈ C_{i−1} | randomBool(n^{−1/k})}
        concurrently foreach v ∈ V \ C_i do
            if ∃C ∈ C_i : ℰ(v, C) ≠ ∅ then                          // (R1)
                C ← C ∪ {v}
                S ← S ∪ random(ℰ(v, C))
            else                                                    // (R2)
                foreach C ∈ C_{i−1} do S ← S ∪ random(ℰ(v, C))
    return (V, S)
end
```

Again our favourite graph



- Let k $= 3$
- Sampling probability $n^{-1/k} = 12^{-1/3} \approx 43,6\%$
- I used my favourite 100-sided dice for this ☺

Again our favourite graph



- Let k = 3
- Sampling probability $n^{-1/k} = 12^{-1/3} \approx 43,6\%$
- I used my favourite 100-sided dice for this ☺
- `bash>for i in 1..12; do echo $(($RANDOM%100+1)); done`

Example(continued)

To keep it short write xyz instead of $\{x, y, z\}$

First dice roll $47, 24, 2, 26, 76, 3, 88, 33, 40, 16, 4, 21$

i	$\mathcal{C}_i$	new in S
0	$\{a, b, c, d, e, f, g, h, i, j, k, l\}$	$\emptyset$

Example(continued)

To keep it short write xyz instead of $\{x, y, z\}$

First dice roll $47, 24, 2, 26, 76, 3, 88, 33, 40, 16, 4, 21$

i	$\mathcal{C}_i$	new in S
0	$\{a, b, c, d, e, f, g, h, i, j, k, l\}$	$\emptyset$
1	$\{ba, ce, d, f, h, ig, j, k, l\}$	ab, ce, gi

Example(continued)

To keep it short write xyz instead of $\{x, y, z\}$

Second dice roll 96, 10, 83, 99, 27, 35, 69, 59, 85

i	$\mathcal{C}_i$	new in S
0	$\{a, b, c, d, e, f, g, h, i, j, k, l\}$	$\emptyset$
1	$\{ba, ce, d, f, h, ig, j, k, l\}$	ab, ce, gi
2	$\{cebdf, hj, igk\}$	$bc, dc, fc, jh, ki, lj, lk$

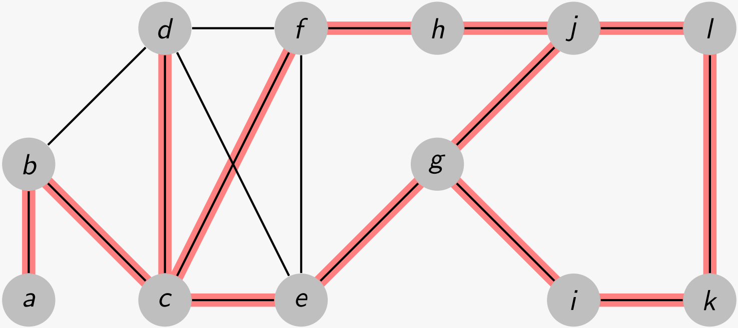
Example(continued)

To keep it short write xyz instead of $\{x, y, z\}$

i	$\mathcal{C}_i$	new in S
0	$\{a, b, c, d, e, f, g, h, i, j, k, l\}$	$\emptyset$
1	$\{ba, ce, d, f, h, ig, j, k, l\}$	ab, ce, gi
2	$\{cebdf, hj, igk\}$	$bc, dc, fc, jh, ki, lj, lk$
3	$\emptyset$	eg, fh, gj

Example(continued)

The result



Lemma

The given algorithm computes $(2k-1,0)$*-spanner* H *of* G*.*

Proof.

- Show that every single edge is stretched by at most $2k-1$

Lemma

The given algorithm computes $(2k-1,0)$-spanner H of G.

Proof.

- Show that every single edge is stretched by at most $2k-1$
- Let $\{u,v\}$ be an arbitrary edge

Lemma

The given algorithm computes $(2k-1,0)$-spanner H of G.

Proof.

- Show that every single edge is stretched by at most $2k-1$
- Let $\{u,v\}$ be an arbitrary edge
- Let l be minimal, s.t. either u or v is unclustered in $\mathcal{C}_l$

Lemma

The given algorithm computes $(2k-1,0)$-spanner H of G.

Proof.

- Show that every single edge is stretched by at most $2k-1$
- Let $\{u, v\}$ be an arbitrary edge
- Let l be minimal, s.t. either u or v is unclustered in $\mathcal{C}_l$
- Wlog. u is unclustered in $\mathcal{C}_l$

Lemma

The given algorithm computes $(2k-1,0)$*-spanner* H *of* G.

Proof.

- Show that every single edge is stretched by at most $2k-1$
- Let $\{u, v\}$ be an arbitrary edge
- Let l be minimal, s.t. either u or v is unclustered in $\mathcal{C}_l$
- Wlog. u is unclustered in $\mathcal{C}_l$
- By `(R2)`: $\exists w \in \mathcal{C}_{l-1}(v) : \{u, w\} \in H$

Lemma

The given algorithm computes $(2k-1,0)$-spanner H of G.

Proof.

- Show that every single edge is stretched by at most $2k-1$
- Let $\{u,v\}$ be an arbitrary edge
- Let l be minimal, s.t. either u or v is unclustered in $\mathcal{C}_l$
- Wlog. u is unclustered in $\mathcal{C}_l$
- By `(R2)`: $\exists w \in \mathcal{C}_{l-1}(v) : \{u,w\} \in H$
- By `(R1)`: $\delta_H(w,v) \leq 2(l-1) = \mathrm{diam}(\mathcal{C}_{l-1}(v))$

Lemma

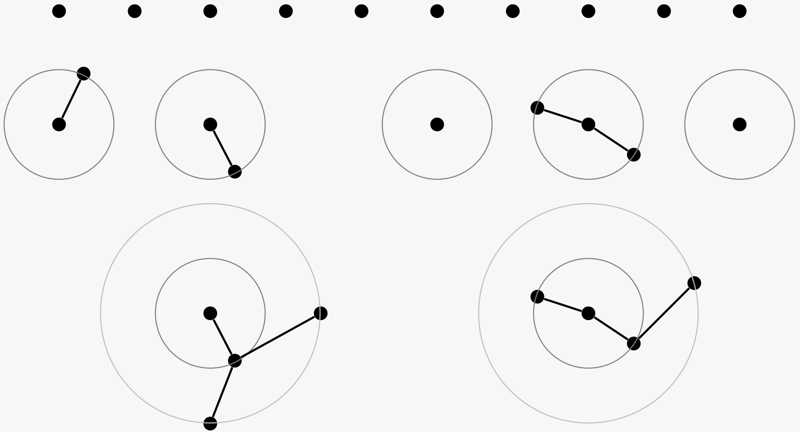
The given algorithm computes $(2k-1,0)$-spanner H of G.

Proof.

- Show that every single edge is stretched by at most $2k-1$
- Let $\{u,v\}$ be an arbitrary edge
- Let l be minimal, s.t. either u or v is unclustered in $\mathcal{C}_l$
- Wlog. u is unclustered in $\mathcal{C}_l$
- By (R2): $\exists w \in \mathcal{C}_{l-1}(v) : \{u,w\} \in H$
- By (R1): $\delta_H(w,v) \leq 2(l-1) = \mathrm{diam}(\mathcal{C}_{l-1}(v))$
- It follows:

$$\delta_H(u,v) \leq 1 + 2(l-1) \leq 2k-1 = (2k-1)\delta_G(u,v)$$

Take-home picture



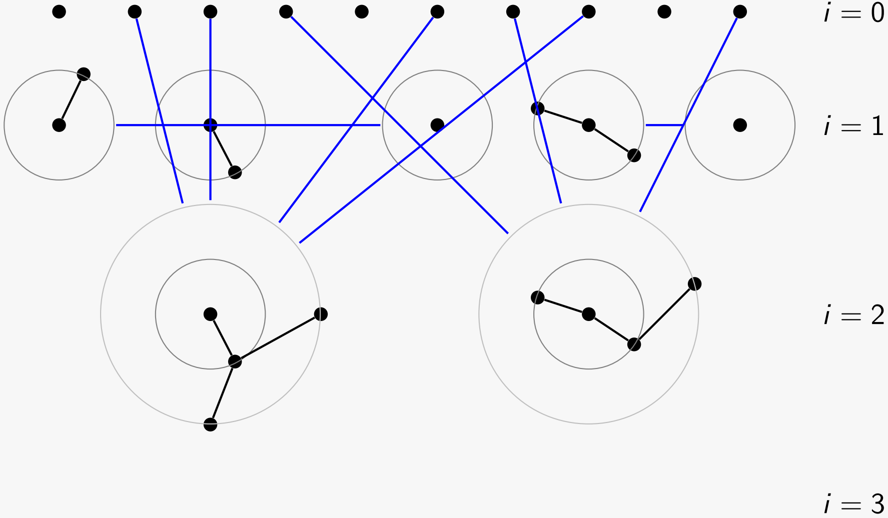
$i = 0$

$i = 1$

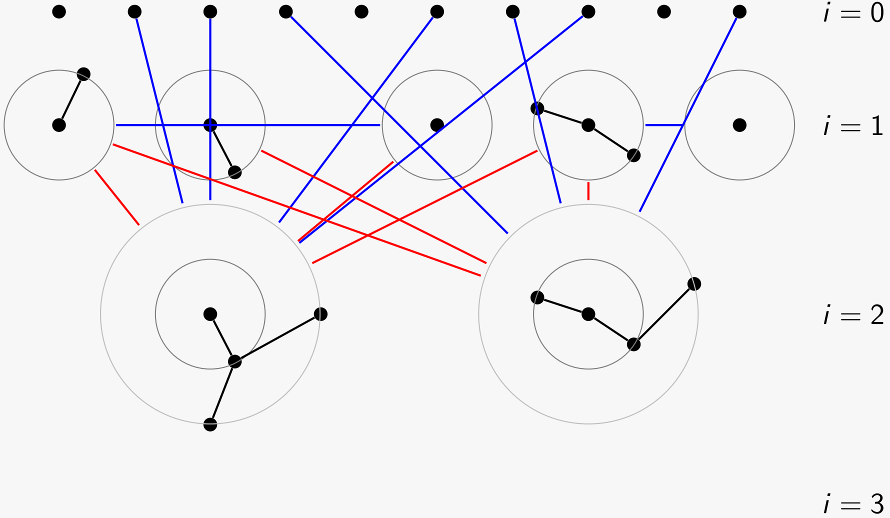
$i = 2$

$i = 3$

Take-home picture



Take-home picture



Construction

Algorithm 3: Randomized $(k, k-1)$-spanner construction

input : Graph $G = (V, E)$ and integer k
output: $(k, k-1)$-spanner H of G
begin
  construct the $(2k-1, 0)$-spanner (V,S)
  for $i \leftarrow 1$ **to** $k-1$ **do** // (R3)
    $S \leftarrow S \cup \{\text{random}(\mathcal{E}(C, C')) \mid C \in \mathcal{C}_i, C' \in \mathcal{C}_{k-1-i}, \mathcal{E}(C, C') \neq \emptyset\}$
  for $i \leftarrow \lceil k/2 \rceil$ **to** $k-1$ **do** // (R4)
    $S \leftarrow S \cup \{\text{random}(\mathcal{E}(C, C')) \mid C \in \mathcal{C}_i, C' \in \mathcal{C}_{i-1}, \mathcal{E}(C, C') \neq \emptyset\}$
end

Again time and space

Spanner size

- both algorithms have the bound $\mathcal{O}(kn^{1+1/k})$
- (R3) and (R4) add each $n^{1+1/k}$ edges as the expected value to the initial $(2k-1,0)$-spanner
- Probabilistic algorithm: bound is the expected value

Again time and space

Spanner size

- both algorithms have the bound $\mathcal{O}(kn^{1+1/k})$
- (R3) and (R4) add each $n^{1+1/k}$ edges as the expected value to the initial $(2k-1, 0)$-spanner
- Probabilistic algorithm: bound is the expected value

Construction time

- both algorithms need $\mathcal{O}(km)$ deterministic time
- (R3) and (R4) only need linear time

In networks...

- no "global" information like distances or shortest paths needed
- `(R1)` - `(R4)` can be reformulated as local rules for a node in a synchronized distributed network
- after $\mathcal{O}(k)$ (constant!) rounds the magic happens ☺
  - every node executes the local rules...
  - constructing globally a spanner

Applications

Some examples

- basis of space-efficient routing tables
- simulation of synchronized protocols in unsynchronized networks
- approximate shortest paths in distributed networks
- construction of approximate distance oracles

Applications

Some examples

- basis of space-efficient routing tables
- simulation of synchronized protocols in unsynchronized networks
- approximate shortest paths in distributed networks
- construction of approximate distance oracles

Variations

- simulating directed graphs by using the roundtrip distance instead of the shortest path
- (α, β)-Steiner spanner

An open question

The girth conjecture (Erdös)

There exist graphs with $\Omega(n^{1+1/k})$ edges and girth[1] $2k+2$.

Consequences

[1] length of the shortest cycle

An open question

The girth conjecture (Erdös)

There exist graphs with $\Omega(n^{1+1/k})$ edges and girth[1] $2k+2$.

Consequences

- Assumption: $G = (V, E)$ fulfils the girth conjecture
- Let $(u, v) = e \in E$, $H = (V, E \setminus \{e\})$
- It holds:

$$\delta_H(u, v) \geq$$

[1] length of the shortest cycle

An open question

The girth conjecture (Erdös)

There exist graphs with $\Omega(n^{1+1/k})$ edges and girth[1] $2k+2$.

Consequences

- Assumption: $G = (V, E)$ fulfils the girth conjecture
- Let $(u, v) = e \in E$, $H = (V, E \setminus \{e\})$
- It holds:

$$\delta_H(u, v) \geq 2k + 1 \geq (2k + 1)\delta_G(u, v)$$

- Thus, any (α, β)-spanner with $\alpha + \beta \leq 2k$ must have at least $\Omega(n^{1+1/k})$ edges.

[1] length of the shortest cycle

An open question

The girth conjecture (Erdös)

There exist graphs with $\Omega(n^{1+1/k})$ edges and girth[1] $2k+2$.

Consequences

- Assumption: $G = (V, E)$ fulfils the girth conjecture
- Let $(u, v) = e \in E$, $H = (V, E \setminus \{e\})$
- It holds:

$$\delta_H(u, v) \geq 2k + 1 \geq (2k + 1)\delta_G(u, v)$$

- Thus, any (α, β)-spanner with $\alpha + \beta \leq 2k$ must have at least $\Omega(n^{1+1/k})$ edges.

Only for $k = 1, 2, 3, 5$ there exist a proof of the girth conjecture.

[1] length of the shortest cycle

Thank you!