Spectral Sparsification

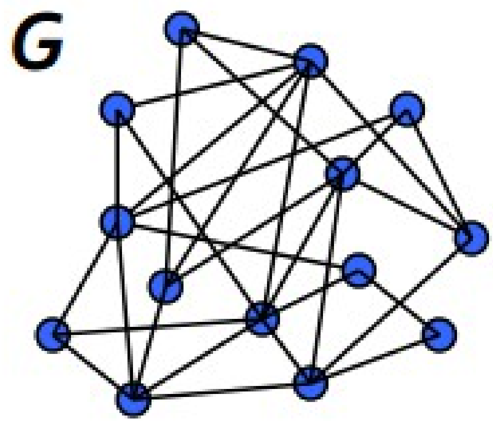
Talk: Eleni Bakali

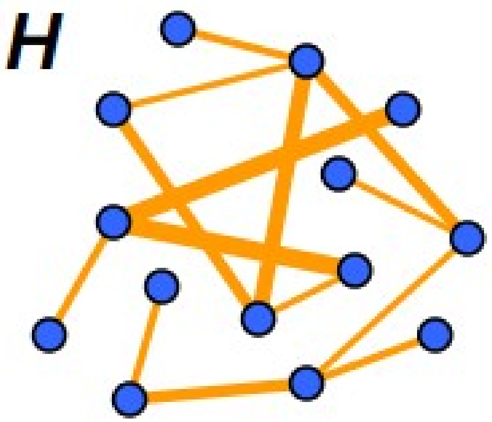
Αλγόριθμοι Δικτύων 2014

Slides: N.Srivastava, E.Bakali

Sparsification

Approximate any graph ***G*** by a sparse graph ***H***.

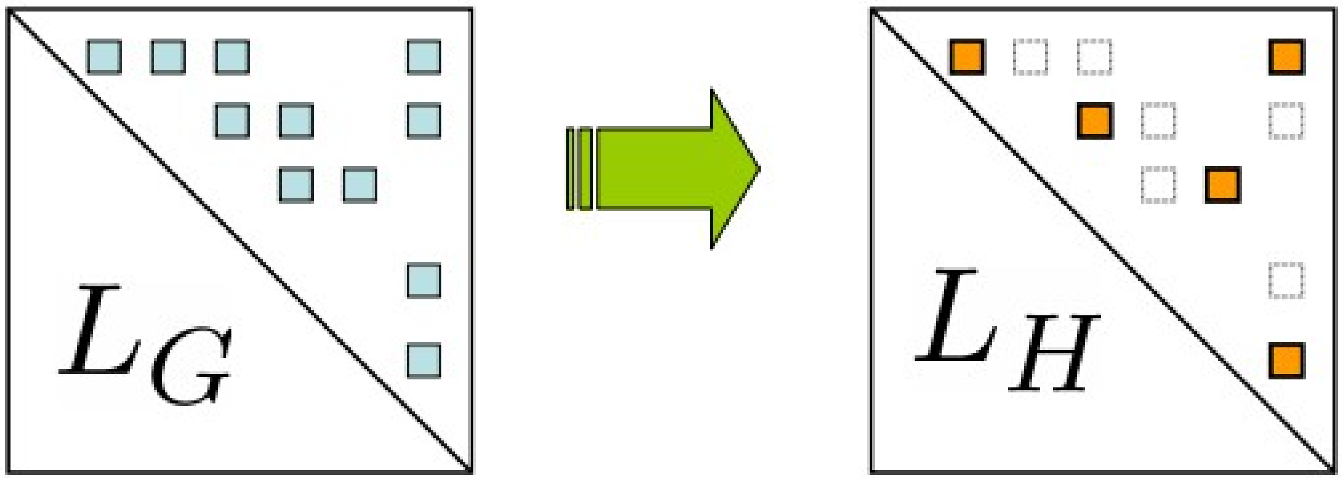




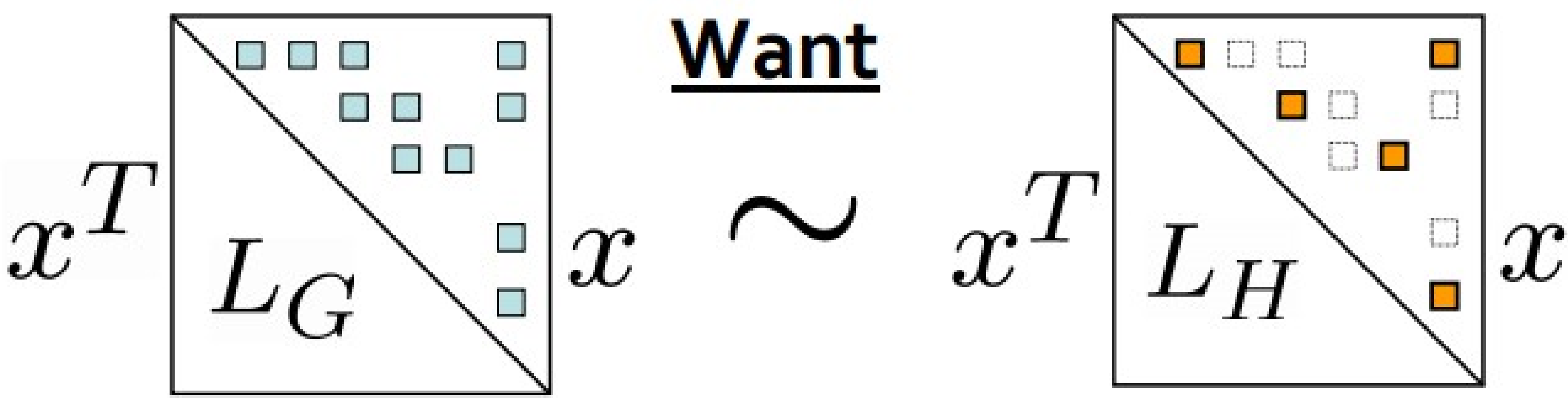
– Nontrivial statement about ***G***

– ***H*** is faster to compute with than ***G***

Goal



Want



The Laplacian (quick review)

$$L_G = D_G - A_G$$

Quadratic form

$$x : V \to \mathbb{R}$$

$$x^T L_G x = \sum_{uv \in E} c_{uv}(x(u) - x(v))^2$$

Positive semidefinite

Ker(L_G)=span(**1**) if ***G*** is connected

Cuts and the Quadratic Form

For characteristic vector $x_S \in \{0,1\}^n$ of $S \subseteq V$

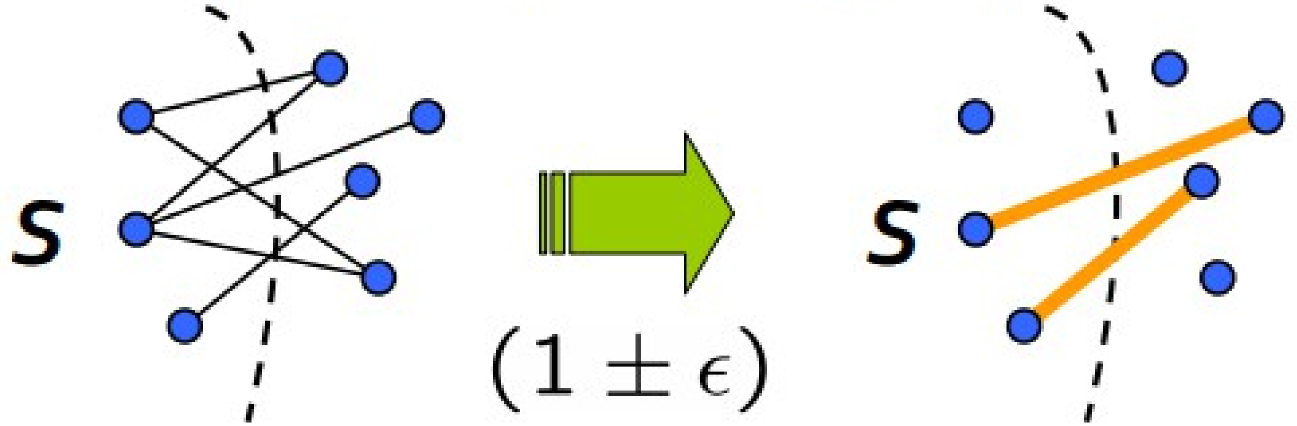
$$
\begin{aligned}
x_S^T L_G x_S &= \sum_{uv \in E} c_{uv}(x(u) - x(v))^2 \\
&= \sum_{uv \in (S,\overline{S})} c_{uv} \\
&= wt_G(S, \overline{S})
\end{aligned}
$$

Cut Sparsifiers [Benczur-Karger'96]

H approximates ***G*** if

for every cut ***S***⊂***V***

sum of weights of edges leaving ***S*** is preserved



Can find ***H*** with O(nlogn/ε²) edges in $\tilde{O}(m)$ time

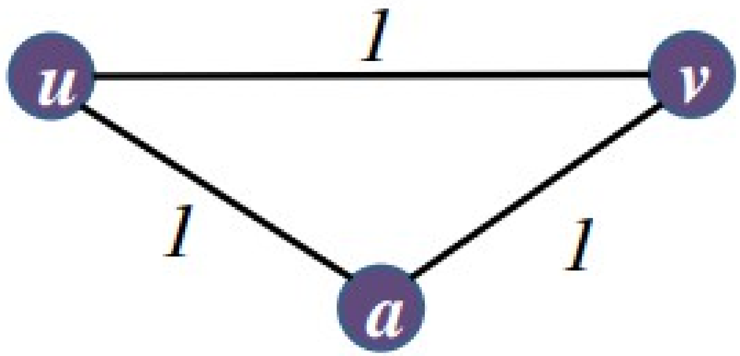
How?

Electrical Flows

Effective Resistance

Identify each edge of G with a unit resistor

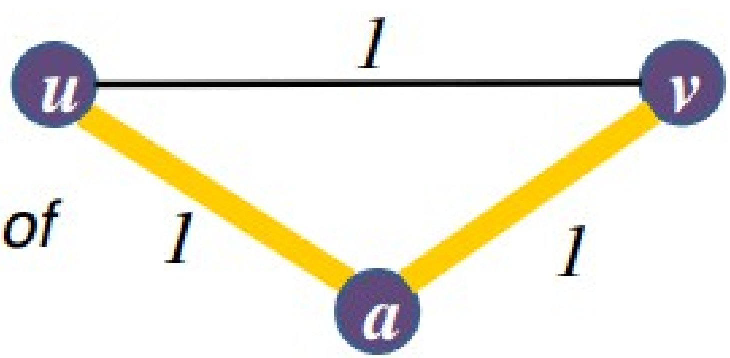
$R_{\mathrm{eff}}(e)$ is resistance between endpoints of e



Effective Resistance

Identify each edge of G with a unit resistor

$R_{\text{eff}}(e)$ is resistance between endpoints of e



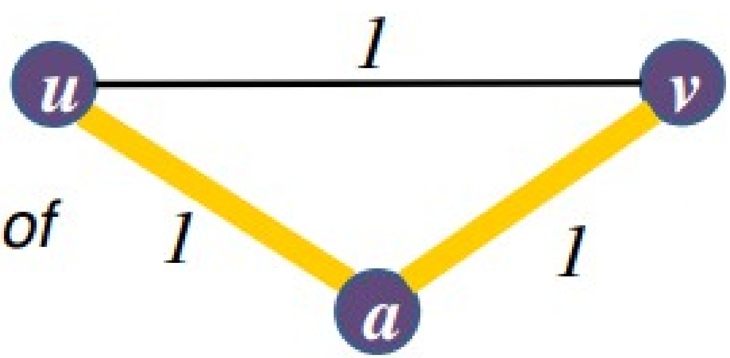
Resistance of path is 2

Effective Resistance

Identify each edge of G with a unit resistor

$R_{\text{eff}}(e)$ is resistance between endpoints of e

Resistance of path is 2



Resistance from u to v is

$$\frac{1}{1/2 + 1/1} = 2/3$$

The Algorithm

Sample edges of G with probability

$$p_e \propto R_{\text{eff}}(e)$$

If chosen, include in H with weight $\frac{1}{p_e}$

Take q=O(nlogn/ε²) samples with replacement

Divide all weights by *q*.

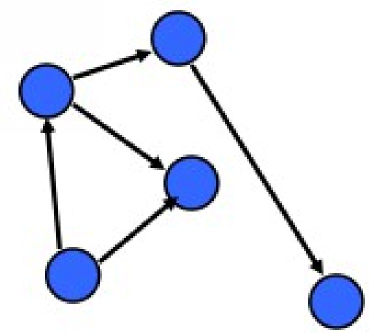
An algebraic expression for R_{eff}

Orient G arbitrarily.

Signed incidence matrix $B_{m \times n}$:

$$B(e,v) = \begin{cases} +1 & \text{if } v \text{ is head of } e \\ -1 & \text{if } v \text{ is tail of } e \\ 0 & \text{otherwise} \end{cases}$$

i.e., $B(uv, \cdot) = \chi_u - \chi_v$.



Write Laplacian as $L = B^T B$

$$R_{\text{eff}}(uv) = (\chi_u - \chi_v)^T L^{-1} (\chi_u - \chi_v)$$
$$= B(uv, \cdot) L^{-1} B(uv, \cdot)^T$$

Άρα μας νοιάζει ο χρόνος υπολογισμού των $R_{eff}(e)$

Nearly Linear Time

$$R_{\text{eff}}(uv) = \|BL^{-1}(\chi_u - \chi_v)\|_2^2$$

So care about distances between cols. of $\boldsymbol{BL^{-1}}$

Θα έπρεπε να λύσουμε m γραμμικά συστήματα (ένα για κάθε γραμμή του BL^{-1})

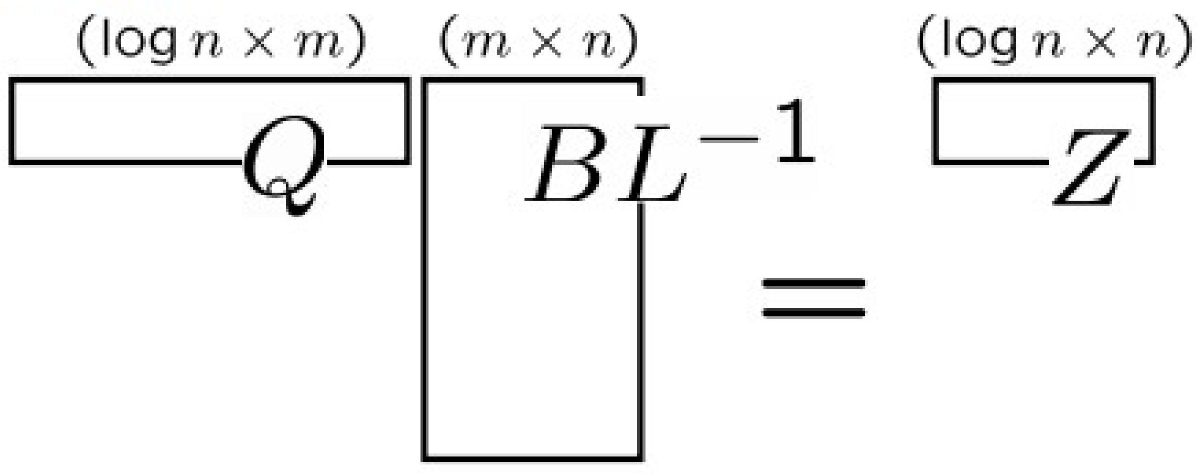
Nearly Linear Time

$$R_{\text{eff}}(uv) = \|BL^{-1}(\chi u - \chi v)\|_2^2$$

So care about distances between cols. of $\boldsymbol{BL^{-1}}$

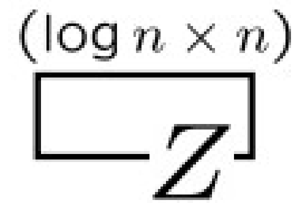
Johnson-Lindenstrauss! Take random $\boldsymbol{Q_{\log n \times m}}$

Set $\boldsymbol{Z = QBL^{-1}}$



Nearly Linear Time

Find **rows** of $Z_{\log n \times n}$ by



$Z = QBL^{-1}$

$ZL = QB$

$z_i L = (QB)_i$

$$R_{\text{eff}}(uv) \sim \|Z(\chi_u - \chi_v)\|^2$$

Solve $O(\log n)$ linear systems in L

Can show approximate R_{eff} suffice.

□

Koutis-Levin-Peng's faster algorithms Main idea

- Approximate $R_{eff}(e)$: **stretch** is a loose approximation to effective resistance

- Compensate for looseness by extra sampling

- re-sparsify

Stretch

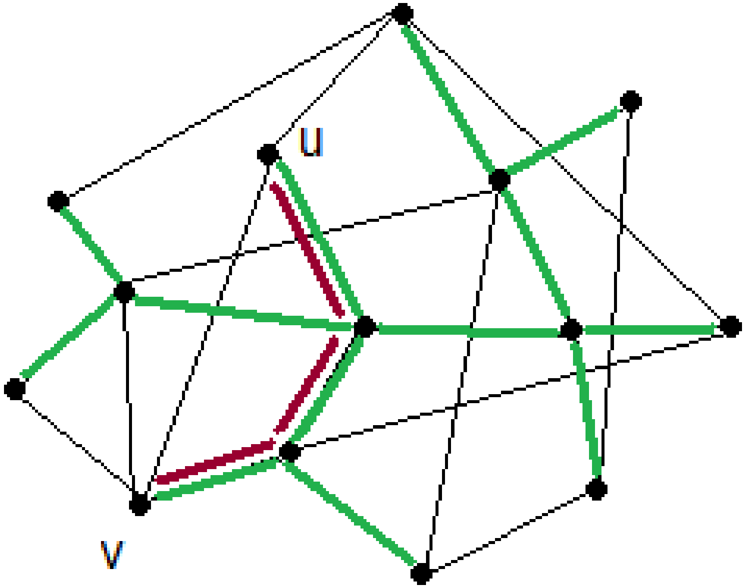
Let G=(V,E) and a **spanning tree** T

Let e=(u,v) and

p=e1,...,ek the **path** on T from u to v

$$stretch_T(e) = w_e \sum_{i=1}^{k} w_{ei}^{-1}$$

$$stretch_T(G) = \sum_{e \in E} stretch(e)$$



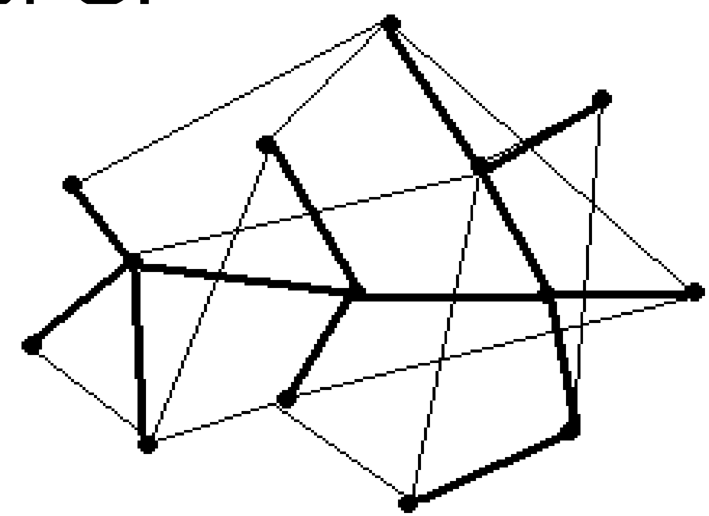
Easy to get a low-stretch spanning tree [AN'12]

Spine Heavy Graph

A graph with a **very low-stretch** spanning tree.

Spine heavy approximation of G:

- Find a low-stretch T
- Scale up tree edges



Solvers run faster on spine heavy graphs [KMP'11]

=>Fast approximation of Reff

Even faster algorithms

dence, poly-bounded weights

- Get an approximation H of G
- Sparsify H to H'
- Find a low stretch tree T of H'
- Approx Reff by stretch on T

- Oversampling
- re-sparsification

Even faster algorithms

dence, unweighted

Progressively sparsify a sequence

$$H = H_0, H_1, \ldots, H_t = G$$

where H_i is 2-approximation of H_{i+1}

Bibliography

- Ittai Abraham and Ofer Neiman. Using petal-decompositions to build a low stretch spanning tree. In Howard J. Karloff and Toniann Pitassi, editors, STOC, pages 395–406. ACM, 2012. 1, 3.4

- Ioannis Koutis, Gary L. Miller, and Richard Peng. A nearly-m log n solver for SDD linear systems. In FOCS ’11: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science, 2011. 1, 2.2, 2.4, 3.4

- Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In STOC’08:Proceedings of the 40th Annual ACM symposium on Theory of Computing, pages 563–568, 2008. 1,2.1, 3.5, 4, 4.1, 5

- Ioannis Koutis, Alex Levin, Richard Peng: Faster spectral sparsification and numerical algorithms for SDD matrices. CoRR abs/1209.5821 (2012)