# *Differentially Private Analysis of Graphs and Social Networks*

*Sofya Raskhodnikova*

*BU Computer Science*

BOSTON UNIVERSITY

# *Publishing information about graphs*

Many types of data can be represented as graphs where

- nodes correspond to individuals
- edges capture relationships
  - **“Friendships” in online social network**
  - **Financial transactions**
  - **Email communication**
  - **Health networks (of doctors and patients)**
  - **Romantic relationships**

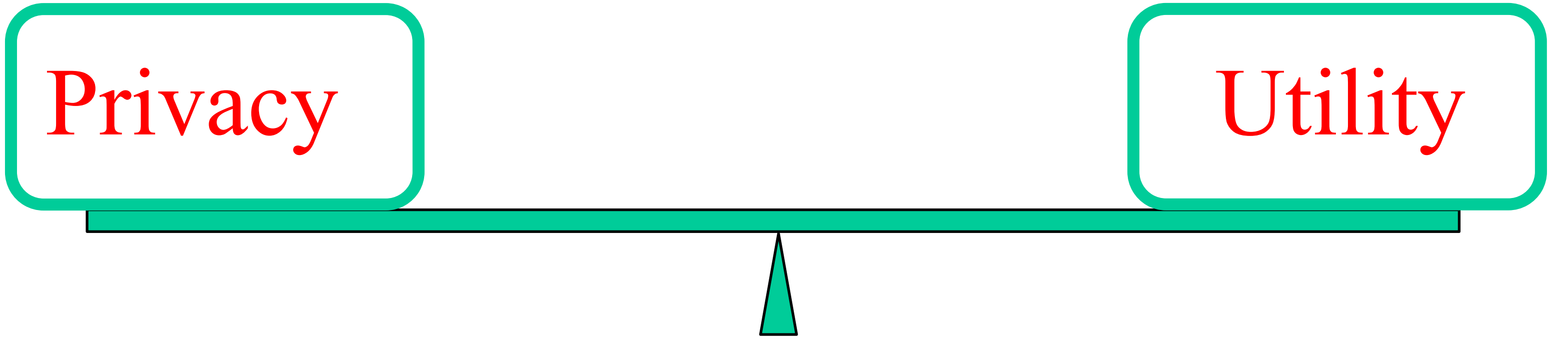
*image source http://community.expressor-software.com/blogs/mtarallo/36-extracting-data-facebook-social-graph-expressor-tutorial.html*


*image source http://www.queticointernetmarketing.com/new-amazing-facebook-photo-mapper/*

***Such graphs contain potentially sensitive information.***

# *Two conflicting goals*

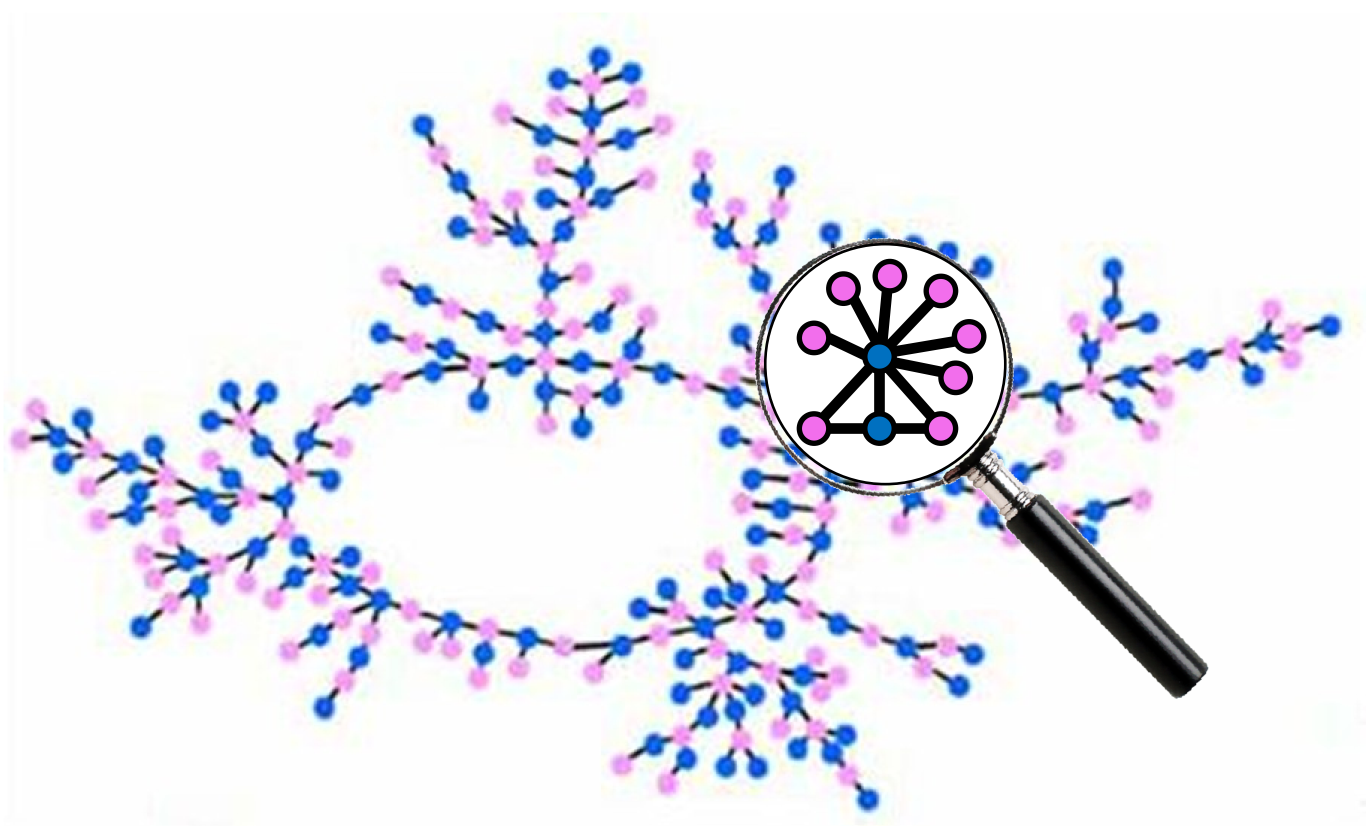
- Privacy: protecting information of individuals.
- Utility: drawing accurate conclusions about aggregate information.

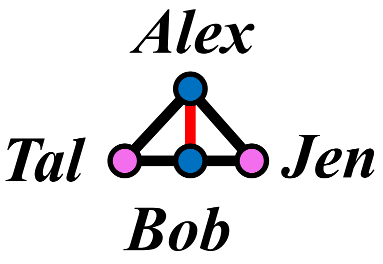


# ``Anonymized” graphs still pose privacy risk

- False dichotomy: personally identifying vs. non-personally identifying information.
- Links and any other information about individual can be used for de-anonymization.

In a typical real-life network, many nodes have unique neighborhoods.






Bearman, Moody, Stovel. *Chains of affection: The structure of adolescent romantic and sexual networks, American J. Sociology, 2008*

# *De-anonymization attacks*

– Movie ratings [Narayanan, Shmatikov 08]

– Social networks

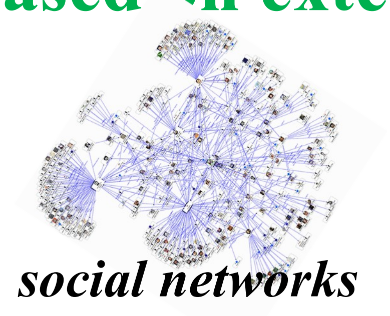
[Backstrom Dwork Kleinberg 07,
Narayanan Shmatikov 09, Narayanan Shi Rubinstein 12]

– Computer networks

[Coull Wright Monrose Collins Reiter 07,
Ribeiro Chen Miklau Townsley 08]

**Can reidentify individuals based on external sources.**

***internet***



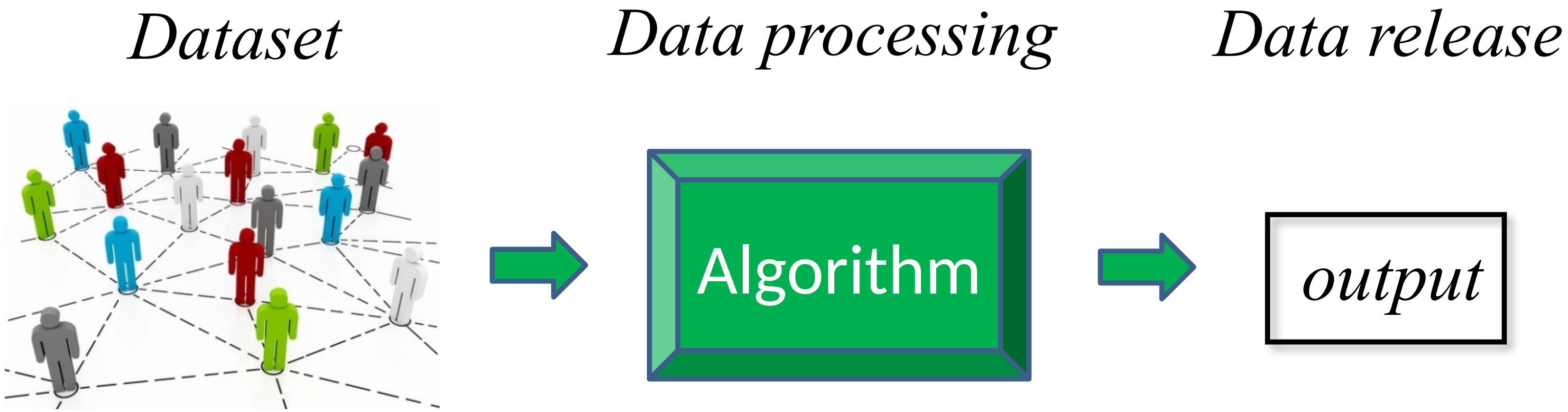
***social networks***

***anonymized datasets***

***What information can be released***

***without violating privacy?***

# *Differential privacy*




**Differential privacy** [Dwork McSherry Nissim Smith 06]

Intuition: Two datasets are ***neighbors*** if they differ in one individual's data.

An algorithm is **differentially private** if its output is roughly the same for all pairs of ***neighbors.***

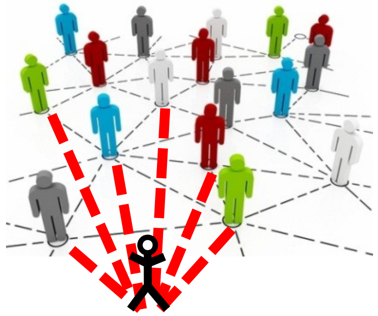
# *Two variants of differential privacy for graphs*

- **Edge** differential privacy

***G:***  ***G:*** 

Two graphs are **neighbors** if they differ in ***one edge***.

- **Node** differential privacy

***G:***  ***G:*** 

Two graphs are **neighbors** if one can be obtained from the other by deleting ***a node and its adjacent edges***.

# *Differential privacy (for graph data)*

*Dataset* — *Data processing* — *Data release*

→ Algorithm → *output*

**Differential privacy** [Dwork McSherry Nissim Smith 06]

An algorithm A is ***-differentially private*** if for all pairs of ***neighbors*** and all possible sets of outputs S:

$$\mathbf{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}\right) \in \boldsymbol{S}\right] \le \boldsymbol{e}^{\epsilon}\,\mathbf{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}'\right) \in \boldsymbol{S}\right]$$

# *Important properties of differential privacy*

- Robustness to side information
- Post-processing:

an algorithm accessing the data via an -differentially private subroutine is -differentially private.

- Composition:

an algorithm that runs  subroutines that are

-differentially private is -differentially private.

## *Is differential privacy too strong?*

- No weaker notion has been proposed that satisfies all three important properties.
- We can attain it for many useful statistics!

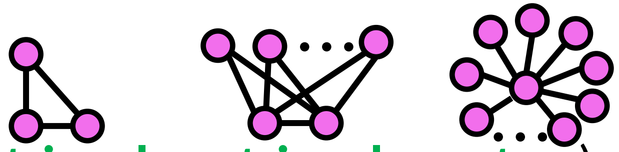
***What graph statistics can be computed accurately***

***with node differential privacy?***

Important: small error on sparse graphs.

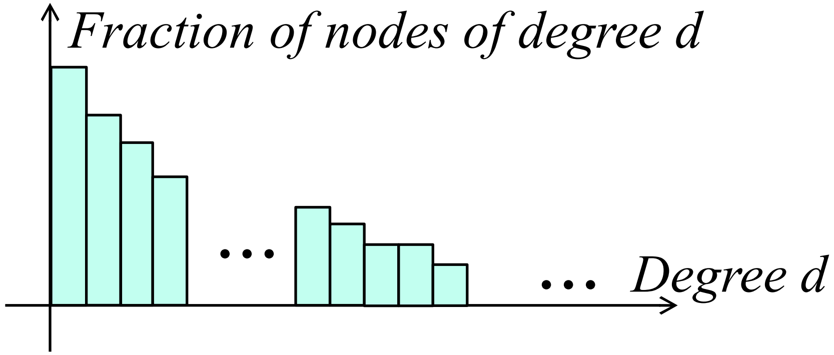
# *Graph statistics*

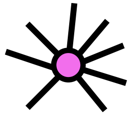
- Number of edges
- Counts of small subgraphs



(e.g., **triangles**, **-triangles**, **-stars**)

- Degree distribution



- Number of connected components, -core decomposition
- How different our graph is from a graph with a certain property
- Correlation between node attributes (e.g., body fat index) and network attributes (e.g., number of obese neighbors).
- Sizes of graph cuts.
- Fitting stochastic block models

…

# *Tools used in differentially private graph algorithms*

- Smooth sensitivity
  - A more nuanced notion of sensitivity than the one mentioned in Katrina's talk
- Sample and aggregate
- Maximum flow
- Linear and convex programming
- Random projections
- Iterative updates
- Postprocessing
- Noisy stochastic gradient descent

# *Differentially private graph analysis*

## *A taste of techniques*

## ***Basic question: how to compute a statistic f***

*Dataset*

*Data processing*

Algorithm

*Data release*

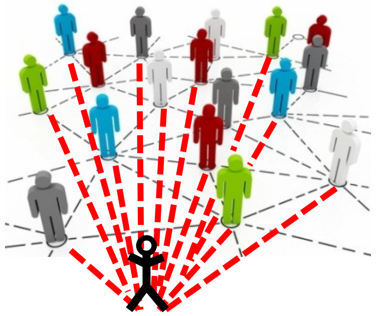
*Approximation to*

**How accurately**

**can an -differentially private algorithm compute f(G)?**

*image source http://www.queticointernetmarketing.com/new-amazing-facebook-photo-mapper/*

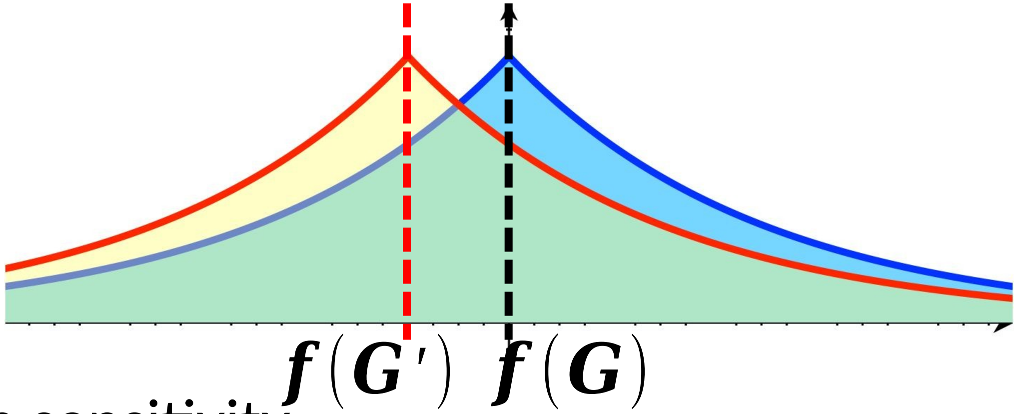
# *Challenge for node privacy: high sensitivity*



- **Sensitivity** of a function  is

$$\partial f = \max_{(\text{node})\,\text{neighbors}\,G,G'} \left| f(G) - f(G') \right|$$

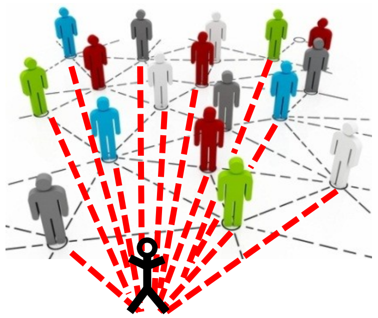
- Katrina’s talk: if  has low sensitivity, it can be computed accurately with differential privacy (by Laplace mechanism).
- Intuition: Adding noise  makes ,  hard to distinguish




- Challenge: Even simple graph statistics have high sensitivity.

# *Challenge for node privacy: high sensitivity*

- **Sensitivity** of a function is



- **Examples:**

for graphs on nodes:

➢ (G) is the number of edges in G.

= .

➢ (G) is the number of triangles in G.

=.

*Idea: project onto graphs with low sensitivity.*

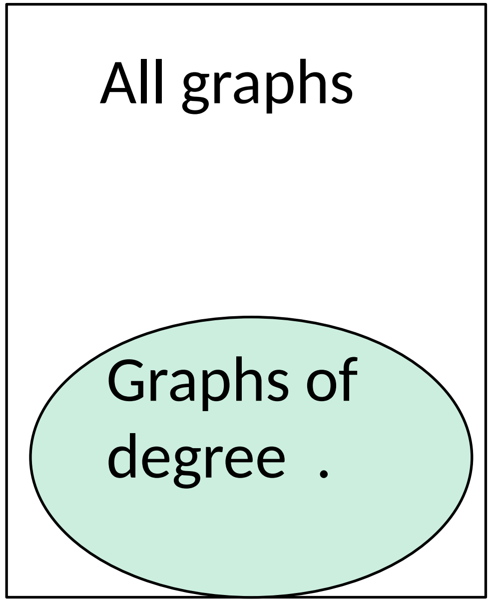
[Kasiviswanathan Nissim Raskhodnikova Smith 13]

See also [Blocki Blum Datta Sheffet 13, Chen Zhou 13]

# *“Projections” on graphs of small degree*

Consider graphs of degree ,

where




- **Examples:**

➢ (G) is the number of edges in G.

➢ (G) is the number of triangles in G.

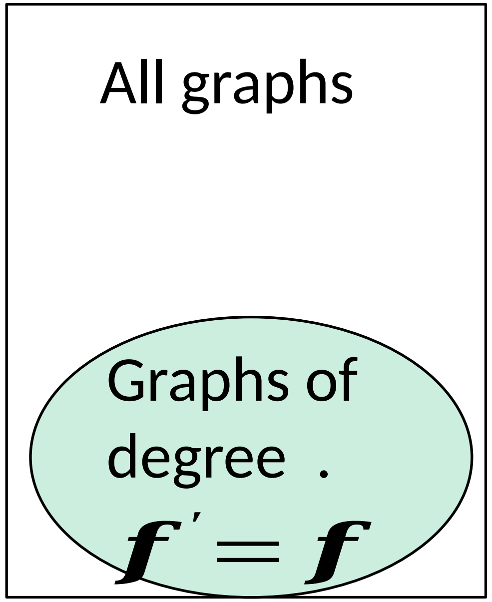
= .

=.

# *Lipschitz extensions*

A function  is a **Lipschitz extension** of  if

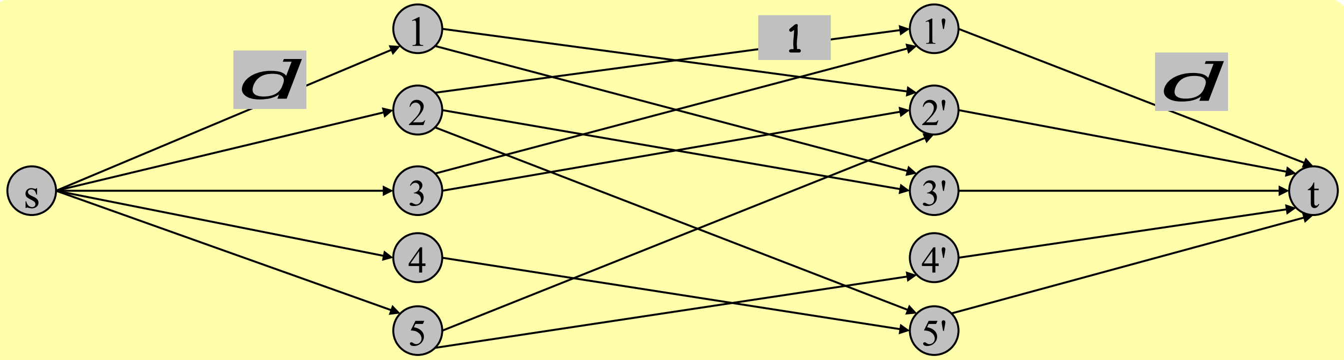
- agrees with  on graphs of degree
- Sensitivity of  is low




- Release  using Laplace mechanism
- There exist Lipschitz extensions for all real-valued functions [McShane 34]
- We design Lipschitz extensions that can be computed efficiently using linear and convex programming for
  - subgraph counts [Kasiviswanathan Nissim Raskhodnikova Smith 13]
  - degree distribution [Raskhodnikova Smith 16, Day Li Lyu 16]
  - number of connected components [Kalemaj Raskhodnikova Smith Tsourakakis]

# *Lipschitz extension of : flow graph*

For a graph G=(V, E), define **flow graph of G**:



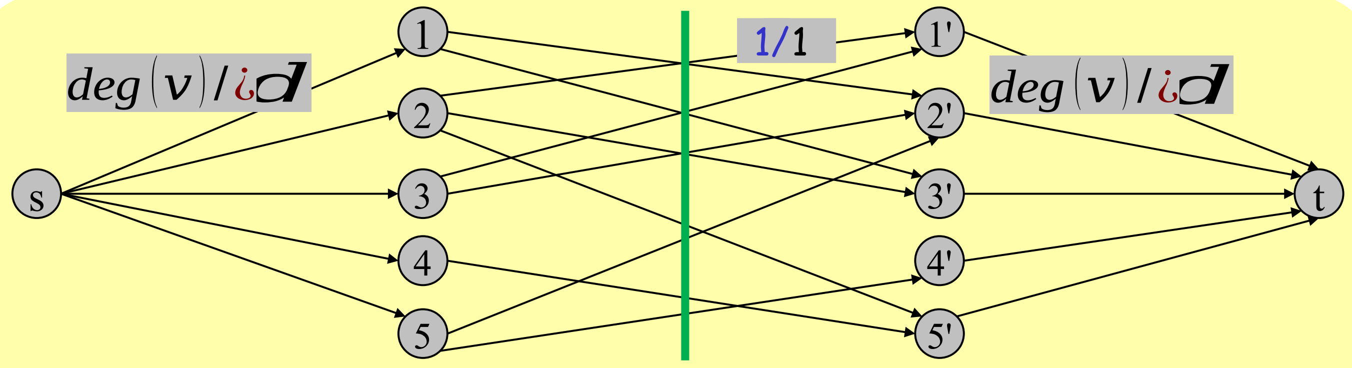

Add edges  and  for all .

**(G)** is the value of the maximum flow in this graph.

**Lemma.** **(G)/2** is a Lipschitz extension of .

# *Lipschitz extension of : flow graph*

For a graph G=(V, E), define **flow graph of G**:




Add edges  and  for all .

**(G)** is the value of the maximum flow in this graph.

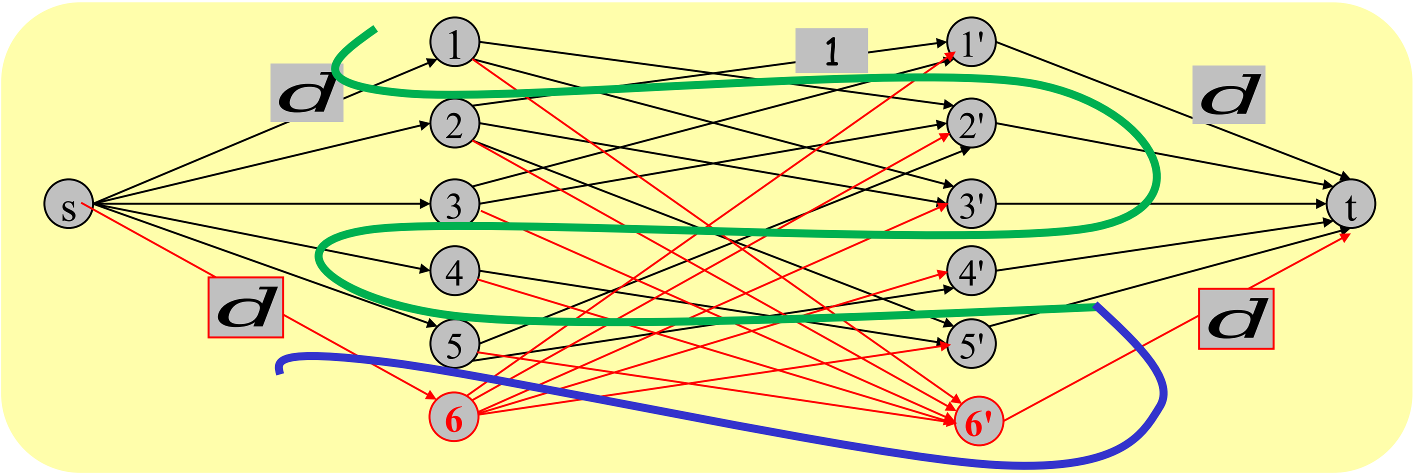
**Lemma.** **(G)/2** is a Lipschitz extension of .

**Proof: (1)  (G) =**  for all **G** of degree at most

(2)   is small

# *Lipschitz extension of : flow graph*

For a graph G=(V, E), define **flow graph of G**:




**(G)** is the value of the maximum flow in this graph.

**Lemma. (G)/2** is a Lipschitz extension of .

**Proof: (1)  (G) =**  for all **G** of degree at most

**(2)**   **= 2**

# *Lipschitz extensions via linear programs*

For a graph G=([n], E), define **LP** with variables  for all triangles :

Maximize $\sum_{T = \triangle \, of \, G} x_T$

for all  triangles

for all nodes

(G) is the value of **LP**.

**Lemma.** (G) is a Lipschitz extension of .

- Computable efficiently.
- Can be generalized to other counting queries.

# *Summary*

- Accurate subgraph counts for realistic graphs can be computed by node-private algorithms
  - Use Lipschitz extensions and linear programming
  - One can choose a “good” value of cut-off degree  privately
  - It is one example of many graph statistics that node-private algorithms do well on.