Graphs and Linear Measurements

Sudipto Guha
University of Pennsylvania

(based on joint work with K. Ahn & A. McGregor)

Graphs

- One of the fundamental representation models in all Computer Science.
- A natural counterpoint to “Big - Vectors”.
- Structure is often more easily represented using graphs.
- And often ***defined*** using graphs.

Linear Measurements

- Inner products.
  - Mostly with (pseudo) random vectors.
  - Fingerprints. Coding Theory.
  - Compress(ed)(or)(ive) sensing.
  - Machine Learning.

- (Very) Easily parallelizable.

This Talk : Questions

- Is it feasible to devise graph algorithms using linear projections?
  - Construct witnesses, not just answering yes/no
  - Approximating the structure of the answer or the value of the answer can be very different

- Are there:
  - Fundamental problems?
  - Fundamental Algorithmic Techniques?
  - Fundamental Analysis avenues?

This Talk: Some answers

- Ahn, Guha, McGregor – SODA 2012, PODS 2012, manuscripts

- A Problem:
- A Technique:
- Analysis Themes:

- Many more exist. We need more.
- We will not focus on specific models too much.

This Talk: Some answers

- Ahn, Guha, McGregor – SODA 2012, PODS 2012, manuscripts

- A Problem: Sampling from a cut in a graph.
- A Technique: Parallel information gathering, sequential use
- Analysis Themes: Adaptivity of actions. Linearity.

- Semi-Streaming model
- Map-Reduce (with some central processing)

$m \rightarrow n$

The importance of being linear

- Order independent $\Rightarrow$ Deletions come free
  $\Rightarrow$ Obviously incremental
  $\Rightarrow$Obvious applications to dynamic graph algorithms

- Suppose a $\exists$ one pass streaming algorithm then
  - Sort the data (order independence)
  - Remove duplicates (deletions/affine-ness)
  - One way access to hash functions!

  $\Rightarrow$ We can assume perfect hash functions
  $\Rightarrow$ Algorithm designer only needs to focus on space
  $\Rightarrow$ Running times can be improved subsequently (possibly use historical data driven/derived features)

This Talk: Some answers

- Ahn, Guha, McGregor – SODA 2012, PODS 2012, manuscripts

- A Problem: Sampling from a cut in a graph.
- A Technique: Parallel information gathering, sequential use
- Analysis Themes: Adaptivity of actions. ~~Linearity.~~

- Semi-Streaming model
- Map-Reduce (with some central processing)

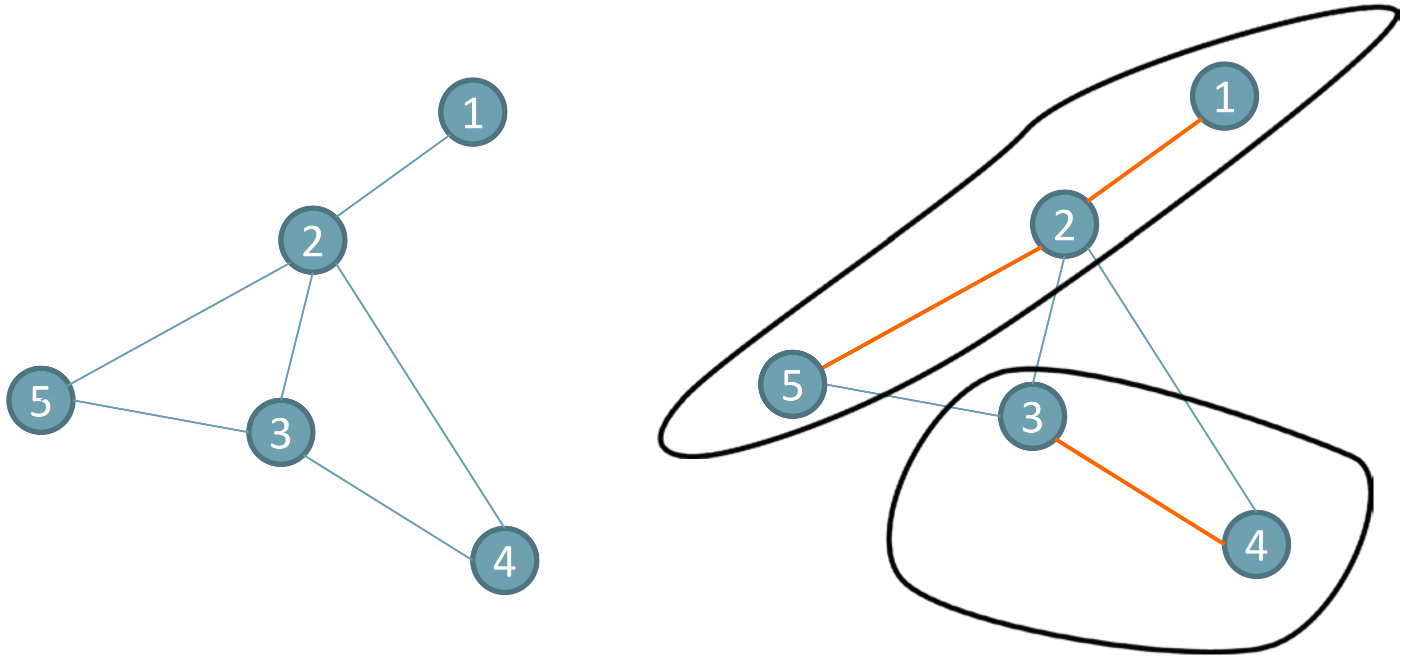
A Technique (and problem to go along)

- Parallel Information gathering
- Yet sequential use

- A graph presented one edge at a time
- Can we maintain connectivity?
- Can we maintain connectivity in O(n) space?
- What if edges are now deleted?

Connectivity in O(log n) rounds

- Every vertex chooses an edge UAR
- Collapse the connected components
- Number of surviving sub-components halves



Connectivity in O(log n) rounds

- Every vertex chooses an edge UAR
- Collapse the connected components
- Number of surviving sub-components halves

- Primitive: Given a vertex choose an edge UAR
- Primitive': Given a set of nodes choose an edge UAR
  - the edges have long sailed on by now
  - we are using the linear projections only

$\Rightarrow$ Given a cut choose an edge UAR
$\Rightarrow$ Note that the cuts are chosen adaptively!
$\Rightarrow$ But we can produce O(log n) data structures at once.

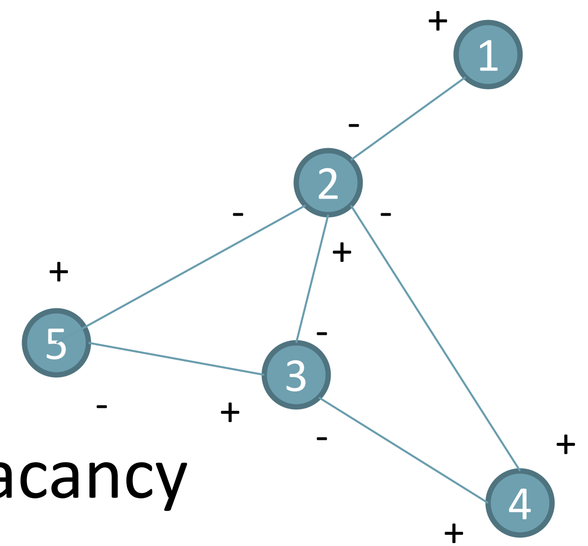
Connectivity in O(log n) rounds

- Every vertex chooses an edge UAR
- Collapse the connected components

- Choose O(log n) data structures
  - Given an arbitrary set, chooses an edge out of it.
  - Disjoint vertex sets "queried simultaneously"
  - This is the "sampling from a cut" problem.
  - We use $\tilde{O}(n)$ space.

Sampling from a Cut

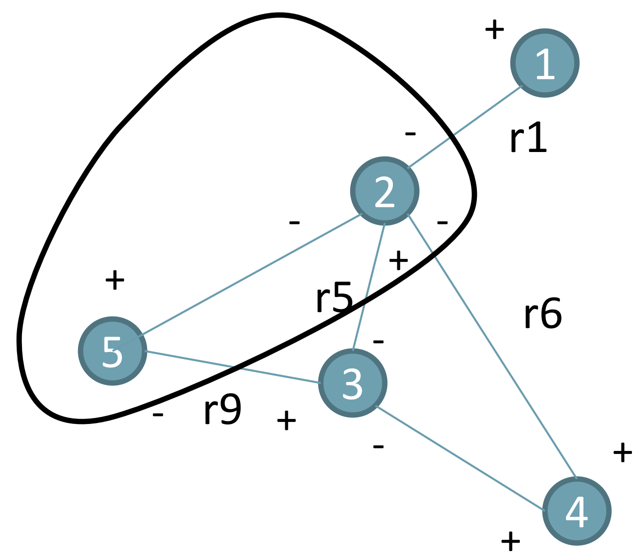
- Consider a graph
- Add orientations
  - Arbitrary but consistent
  - Number the edges
  - "Consider" the vertex-edge adjacency



$$
\begin{matrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0
\end{matrix}
$$

Sampling from a Cut

- Give arbitrary weights
- Add up weights for a vertex
- Given set, compute the sum



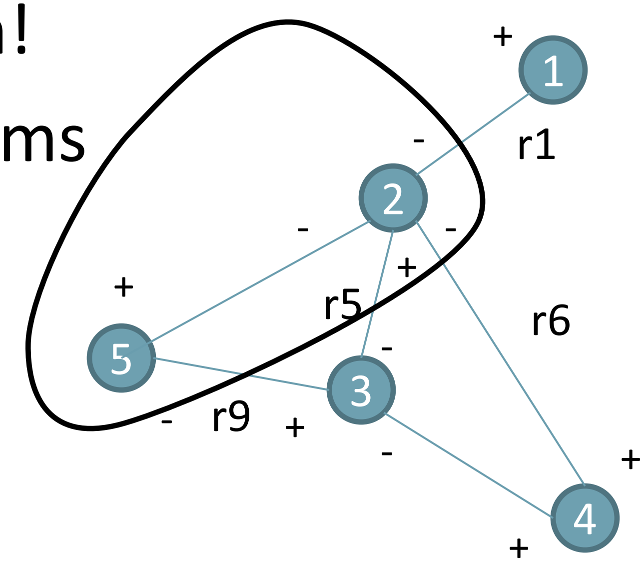
r1	r2	r3	r4	r5	r6	r7	r8	r9	r10
1	0	0	0	0	0	0	0	0	0
-1	0	0	0	1	-1	-1	0	0	0
0	0	0	0	-1	0	0	-1	1	0
0	0	0	0	0	1	0	1	0	0
0	0	0	0	0	0	1	0	-1	0

- r1 + r5 - r6 - r7

r7 - r9

Sampling from a Cut

- Reduces to a streaming problem!
- "Stream"= Union of vertex streams
- Solutions exist (ℓ_0 sampling)
- Space is $\tilde{O}(1)$ per vertex



r1	r2	r3	r4	r5	r6	r7	r8	r9	r10
1	0	0	0	0	0	0	0	0	0
-1	0	0	0	1	-1	-1	0	0	0
0	0	0	0	-1	0	0	-1	1	0
0	0	0	0	0	1	0	1	0	0
0	0	0	0	0	0	1	0	-1	0

- r1 + r5 - r6 - r7

r7 - r9

Recap: ℓ_0 Solutions

- Consider known universe [a] of positive integers
- Suppose we knew the number of distinct elements x up to powers of 2

- Hash [a] $\rightarrow$ [0,1] retain values [0,x/a]
- Of all v $\in$ [a] that hash to [0,x/a]

  – Maintain count
  – Sum
  – Sum of squares

  Sufficient to test if all items are equal

- Return average

Sampling from a Cut

- Why is this a fundamental problem?
- Lets consider some applications …

Minimum Spanning Trees

- Exact computation based on linear projections is provably hard.
- Kruskal's algorithm – add least weighted edge
- If the edge weights are integers; it suffices to count the number of components!
- $(1+\varepsilon)$ approximation in 1 pass and $\tilde{O}(n)$ space

Min Cut

- (In general) Connectivity answers
  - Is there an edge across this cut
- Suppose we asked how many? Say, find the MinCut.

- Karger's algorithm via uniform sampling.

- Easy in insertion model

- With deletions, remove k spanning trees
  - Sequentially (but compute them in 1 pass in parallel)
  - If cuts were small then we have all the edges!
  - If cuts are large then? "Layered graphs"

Cut Sparsification

- (In general) Connectivity answers
  - Is there an edge across this cut
- Suppose we asked how many?

- Goal: Store few edges and estimate each cut to $1 \pm \varepsilon$
- Benczur & Karger 1996: sampling
- Easy in insertion model

- With deletions, remove k spanning trees
  - Sequentially (but compute them in 1 pass in parallel)
  - If cuts were small then we have all the edges!
  - If cuts are large then? “Layered graphs”

Chain of results

- Sampling from a cut
  $\rightarrow$ Connectivity $\rightarrow$ MST
  $\rightarrow$ MinCut $\rightarrow$ Cut-Sparsification
  - Maximum Matching (Dual of Cut-Covering)
  - Multicut $\rightarrow$ Correlation Clustering
  - Spectral Sparsification

- Counting number of subgraphs
  - Replace vertex-edge incidence by subgraph-edge incidences. Otherwise similar idea applies.

Spectral Sparsification

- Spielman & Srivastava
- Conductance, mixing of random walks, clustering
- A generalization of cuts.

- A vector X with ± 1 entries represent a cut.
- $X^T LX$ = size of a cut where L=D-A and A is the vertex-vertex adjacency matrix; D=diagonal matrix of degrees

- Sparsification: preserve all $X^T LX$ where X is a vector with ± 1 entries

- Spectral sparsification : X is an arbitrary vector.

Spectral Sparsification

- Each edge is a 1 Ohm resistor
- Basic sub-problem:
  - Given s,t estimate the effective resistance.
- (small space, 1 pass, using linear sketches) ?

- Yes: sample e w.p. proportional to r_e and give weight $1/r_e$

- $1 \leq r_e c_e \leq n^{2/3}$ for simple unweighted graphs.
- And this is tight!

- Subquadratic space algorithm

Conclusion

- Examples of graph problems using linear projections
- Need more problems & connections
- Showed $\exists$ results; lots of places for improvements