

Fast Approximation of Betweenness Centrality through Sampling

M. Riondato and E. Kornaropoulos



Brown University
Computer Science

Boston University – October 18th, 2013



Motivation

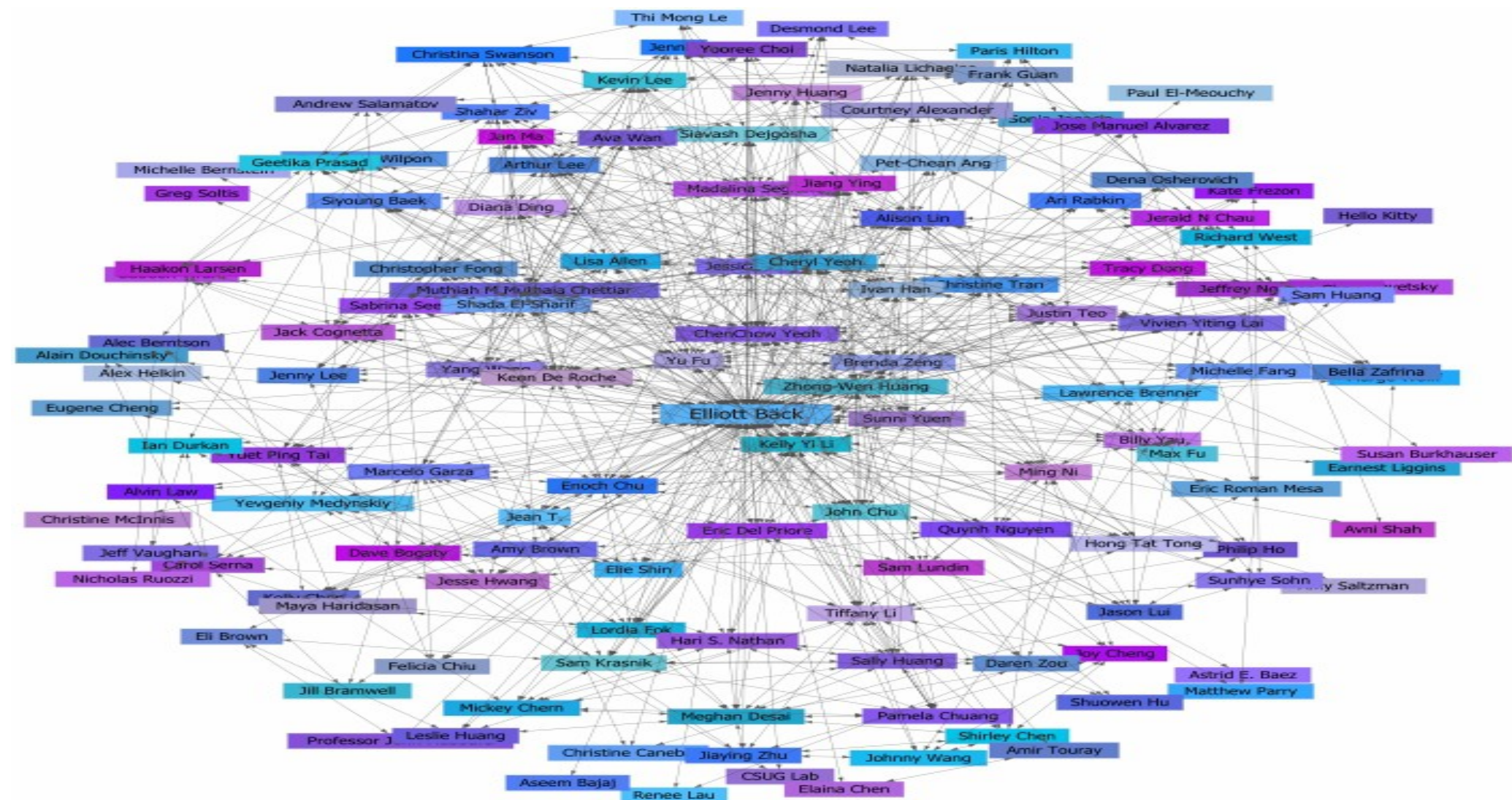
why is it interesting?







Relationship Graph





Motivation

- Proliflication of online social networks
- Social Network Analysis pre-existed them
- Core task:
Find the most important players



Motivation

- Proliferation of online social networks
- Social Network Analysis pre-existed them
- Core task:
Find the most important players
- why?
To target them (rumors, connectivity, marketing, promotions, ...)



Motivation

- Proliflication of online social networks
- Social Network Analysis pre-existed them
- Core task:
Find the most important players
- why?
To target them (rumors, connectivity, marketing, promotions, ...)
- Not just social networks: road networks (used in GPS navs), computer networks (Autonomous Systems), protein networks



Motivation

- Proliferation of online social networks
- Social Network Analysis pre-existed them
- Core task:
Find the most important players
- why?
To target them (rumor marketing, promotion)
Not well defined.
What is “important”?
- Not just social networks:
(used in GPS navs), computer networks
(Autonomous Systems), protein networks



Centrality Indices

- Centrality indices measure the **relative importance** of vertices in a graph
- Introduced in sociology literature, '70s



Centrality Indices

- Centrality indices measure the **relative importance** of vertices in a graph
- Introduced in sociology literature, '70s
- Different **flavours of importance**
 - Many definitions of centrality indices
- Most are **based on shortest paths**
 - **Intuition**: information “spreads” along shortest paths (probably not true)
- Can be extended to include **routing info**



Computing Centrality

- Today's networks have 10^8 vertices
- **Exact** computation requires many runs of **Single Source Shortest Paths (SSSP)** + **Aggregation**
- **Too expensive**, despite tricks to speed up the aggregation **[Brandes01]**



Computing Centrality

- Today's networks have 10^8 vertices
- **Exact** computation requires many runs of **Single Source Shortest Paths (SSSP)** + **Aggregation**
- **Too expensive**, despite tricks to speed up the aggregation **[Brandes01]**
- **Sampling** can help:
trade-off accuracy for speed



Outline

- Motivation
- Definition and settings
- Exact and simple sampling algorithms
- Our result: a fast sampling algorithm
- Better approximation for the top-K vertices
- Experiments
- Conclusions



Graph

- **Graph** $G = (V, E)$
 - $|V| = n$, $|E| = m$
 - Can be **weighted**: $w_e \geq 0, e \in E$
 - Can be **directed**
 - No self-loops
 - No multiple edges between a pair of vertices



(Shortest) Paths

- **Path** $p = (v_1, \dots, v_{|p|})$ ordered tuple of vertices
- End points of p : $\{v_1, v_{|p|}\}$
- **Internal vertices**: $\text{Int}(p) = p \setminus \{v_1, v_{|p|}\}$
- \mathcal{S}_{uv} : Set of shortest paths from u to v
- $\sigma_{uv} = |\mathcal{S}_{uv}|$: no. of shortest paths from u to v
- $\mathbb{S}_G = \bigcup_{(u,v) \in V \times V, u \neq v} \mathcal{S}_{uv}$: all shortest paths in G



Betweenness Centrality

- The **betweenness centrality** of a vertex $v \in V$ measures (roughly) the **fraction of shortest paths** going through v



Betweenness Centrality

- The **betweenness centrality** of a vertex $v \in V$ measures (roughly) the **fraction of shortest paths** going through v

- Let $\mathcal{T}_v = \{p \in \mathbb{S}_G : v \in \text{Int}(p)\}$

- **Betweenness centrality** of v

$$b(v) = \frac{1}{n(n-1)} \sum_{p_{uw} \in \mathbb{S}_G} \frac{1_{\mathcal{T}_v}(p_{uw})}{\sigma_{uw}} = \frac{1}{n(n-1)} \sum_{p_{uw} \in \mathcal{T}_v} \frac{1}{\sigma_{uw}}$$

- **k-betweenness**: local variant, consider only shortest **paths of length up to k**



Outline

- Motivation
- Definition and settings
- Exact and simple sampling algorithm
- Our result: a fast sampling algorithm
- Better approximation for the top-K vertices
- Experiments
- Conclusions



Exact Algorithm

- Naïve exact algorithm for betweenness:
 - All Pairs Shortest Paths + Aggregation
 - The aggregation part dominates
 - Complexity: $\Theta(n^3)$



Exact Algorithm

- **Naïve exact** algorithm for betweenness:
 - All Pairs Shortest Paths + Aggregation
 - The aggregation part dominates
 - **Complexity**: $\Theta(n^3)$
- **[Brandes01]**:
 - **split aggregation in smaller parts** by considering partial contributions
 - **Complexity**: $O(nm)$ or $O(nm + n^2 \log n)$



Exact Algorithm

- **Naïve exact** algorithm for betweenness:
 - All Pairs Shortest Paths + Aggregation
 - The aggregation part dominates
 - **Complexity**: $\Theta(n^3)$
- **[Brandes01]**:
 - **split aggregation in small** considering partial contributions
 - **Complexity**: $O(nm)$ or $O(nm + n^2 \log n)$

This is still too much for large networks



Sampling to the Rescue

- **Solution:** use **sampling**!
 - **Trade off accuracy for speed**
 - Can guarantee high quality approximations with high confidence
- ...especially when the analysis is tight!





Sampling to the Rescue

- Solution: use **sampling**!
 - Trade off accuracy for speed
 - Can guarantee high quality approximations with high confidence
- ...especially when the budget is tight!



Key questions:

- What should we sample?
- How much should we sample?



Guarantees

- We want (probabilistic) guarantees on the approximation



Guarantees

- We want (probabilistic) guarantees on the approximation
- An (ε, δ) -approximation for the betweenness is a set of estimations $\tilde{b}(v)$ for all $v \in V$ such that

$$\Pr \left(\exists v \in V : |\tilde{b}(v) - b(v)| > \varepsilon \right) < \delta$$

- ε controls the accuracy
- δ controls the confidence
- Trade-off: Higher accuracy and/or confidence requires more samples!



Simple Sampling Algorithm

- [BrandesPich07] inspired by [EppsteinWang01]
- The algorithm:

For r times do

 Sample random vertex v

 Compute SSSP from v

 Perform partial computation for $\tilde{b}(w), w \in V$



Simple Sampling Algorithm

- [BrandesPich07] inspired by [EppsteinWang01]
- The algorithm:

For r times do

Sample random vertex v

Compute SSSP from v

Perform partial computation for $\tilde{b}(w), w \in V$

Like exact algorithm
[Brandes01]



Simple Sampling Algorithm

- [BrandesPich07] inspired by [EppsteinWang01]

- The algorithm:

For r times do

Sample random vertex v

Compute SSSP from v

Perform partial computation for $\tilde{b}(w), w \in V$

Like exact algorithm
[Brandes01]

- At each step, the algorithm computes many shortest paths (at least $n-1$ for a connected, undirected graph)



Simple Sampling Algorithm

- [BrandesPich07] inspired by [EppsteinWang01]
- The algorithm:

For r times do

Sample

Sample size n
How to choose it?

Compu

Trade off between
accuracy/confidence
and speed

Like exact algorithm
[Brandes01]

Perform partial computation for $\tilde{b}(w), w \in V$

- At each step, the algorithm computes many shortest paths (at least $n-1$ for a connected, undirected graph)



Sample Size Derivation

- How to compute r ?



Sample Size Derivation

- How to compute r ?
- Use **Hoeffding bound** (method of bounded differences) to bound deviation of estimated betweenness from exact value **for a single vertex**:

$$\Pr(|\tilde{b}(v) - b(v)| > \varepsilon) < 2e^{-2r\varepsilon^2}$$



Sample Size Derivation

- How to compute r ?
- Use **Hoeffding bound** (method of bounded differences) to bound deviation of estimated betweenness from exact value **for a single vertex**:

$$\Pr(|\tilde{b}(v) - b(v)| > \varepsilon) < 2e^{-2r\varepsilon^2}$$

- Use **union bound** over all n vertices



Sample Size Derivation

- How to compute r ?
- Use **Hoeffding bound** (method of bounded differences) to bound deviation of estimated betweenness from exact value **for a single vertex**:

$$\Pr(|\tilde{b}(v) - b(v)| > \varepsilon) < 2e^{-2r\varepsilon^2}$$

- Use **union bound** over all n vertices
- Sample size r for (ε, δ) -approximation:

$$r \geq \frac{1}{2\varepsilon^2} \left(\ln n + \ln 2 + \ln \frac{1}{\delta} \right)$$



Drawbacks

- The sample size depends on $\ln n$
 - Obtained from the union bound: **loose!**



Drawbacks

- The sample size depends on $\ln n$
 - Obtained from the union bound: **loose!**
 - Is this **the right quantity?**
 - We believe not
 - It **should be a characteristic quantity of the graph**



Drawbacks

- The sample size depends on $\ln n$
 - Obtained from the union bound: **loose!**
 - Is this **the right quantity?**
 - We believe not
 - It **should be a characteristic quantity of the graph**
- At each sample, “heavy” computation (SSSP)
 - **Touch a lot of edges**, has low “**locality**”



Outline

- Motivation
- Definition and settings
- Exact and simple sampling algorithms
- Our result: a fast sampling algorithm
- Better approximation for the top-K vertices
- Experiments
- Conclusions



Fast Sampling Algorithm

- Our algorithm **solves the drawbacks**:
 - Does not use the union bound
 - use **VC-dimension**
 - Its sample size depends on a characteristic quantity of the graph (the **vertex-diameter**)
 - Not on number of vertices
 - Performs a **single s-t shortest path computation per sample**
 - Fewer edges touched, better locality



Fast Sampling Algorithm

- The algorithm:

For r times do

Sample a **random pair** of vertices (u, v)

Compute \mathcal{S}_{uv} , all shortest paths from u to v

Select a path p uniformly at random from \mathcal{S}_{uv}

For each $w \in \text{Int}(p)$

$$\tilde{b}(w) \leftarrow \tilde{b}(w) + 1/r$$



Vertex Diameter

- The **vertex diameter** $VD(G)$ of G is the **maximum size of a shortest path** between a pair of vertices of G :

$$VD(G) = \max\{|p| : p \in \mathbb{S}_G\}$$

- If the graph is not weighted: $VD(G) = \Delta_G + 1$
- No relationship in general if G is weighted



Analysis

- **Theorem:**

Given $\varepsilon, \delta \in (0, 1)$, if

$$r \geq \frac{1}{\varepsilon^2} \left(\lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1 + \ln \frac{1}{\delta} \right)$$

then $\Pr \left(\exists v \in V : |\tilde{\mathbf{b}}(v) - \mathbf{b}(v)| > \varepsilon \right) < \delta$

- we have an (ε, δ) -approximation



Analysis

- **Theorem:**

Given $\varepsilon, \delta \in (0, 1)$, if

$$r \geq \frac{1}{\varepsilon^2} \left(\lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1 + \ln \frac{1}{\delta} \right)$$

then $\Pr \left(\exists v \in V : |\tilde{b}(v) - b(v)| > \varepsilon \right) < \delta$

Always less than n

- We have an (ε, δ) -approximation



Analysis

- Theorem:

Given $\varepsilon, \delta \in (0, 1)$, if

$$r \geq \frac{1}{\varepsilon^2} \left(\lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1 + \ln \frac{1}{\delta} \right)$$

then $\Pr \left(\exists v \in V : |\tilde{b}(v) - b(v)| > \varepsilon \right) < \delta$

Always less than n

- We have an (ε, δ) -approximation
- For the proof, we use **VC-Dimension**



VC-Dimension

- [VapnikChervonenkis71]
- Combinatorial property of a collection of subsets from a domain
- Measures the “richness”, “expressivity” of the subsets



VC-Dimension

- [VapnikChervonenkis71]
- Combinatorial property of a collection of subsets from a domain
- Measures the “richness”, “expressivity” of the subsets
- Given a probability distribution on the domain, if we know the VC-dim of a collection of subsets, we can compute the sample size sufficient to approximate the probability mass of each subset using a sample and the empirical average



Range Sets

- VC-Dimension is defined on **range sets**
- B : Domain
- \mathcal{R} : collection of subsets from B (**ranges**)
- **No restrictions:**
 - B can be **infinite**
 - \mathcal{R} can be **infinite**
 - \mathcal{R} can contain **infinitely-large** subsets of B



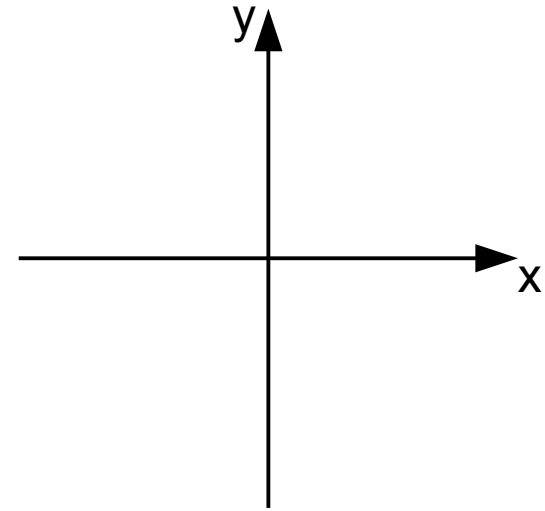
VC-Dimension

- Range set \mathcal{R} on domain B
- For any $C \subseteq B$, define $P_C = \{C \cap F : F \in \mathcal{R}\}$
- C is **shattered** if $P_C = 2^C$
- The VC-Dimension of \mathcal{R} is the **size of the largest shattered subset** of B



Example of VC-Dimension

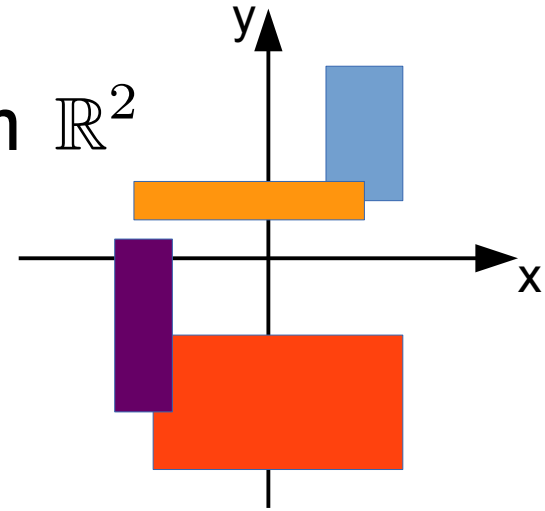
- $B = \mathbb{R}^2$





Example of VC-Dimension

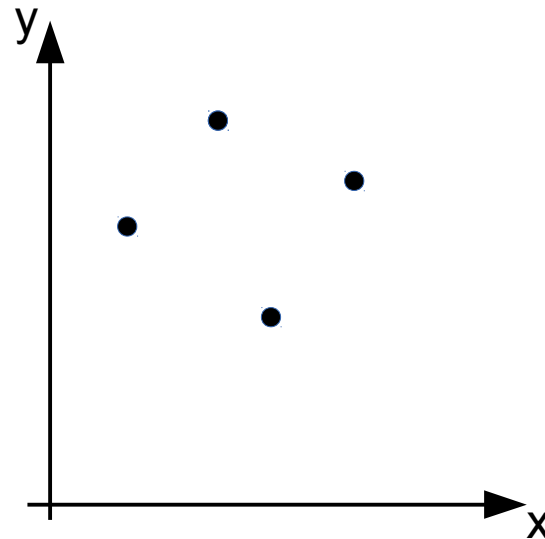
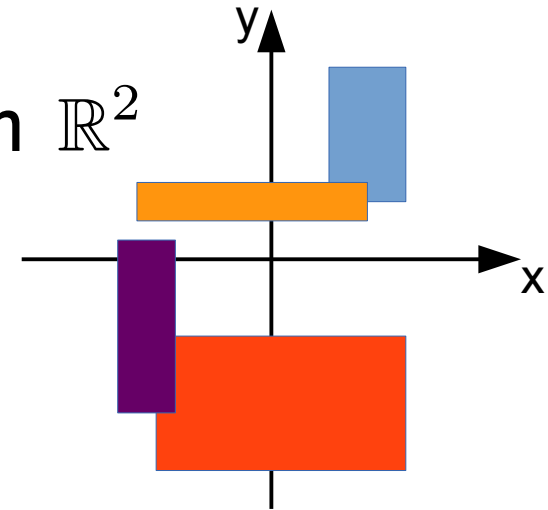
- $B = \mathbb{R}^2$
- $\mathcal{R} =$ all axis-aligned rectangles in \mathbb{R}^2





Example of VC-Dimension

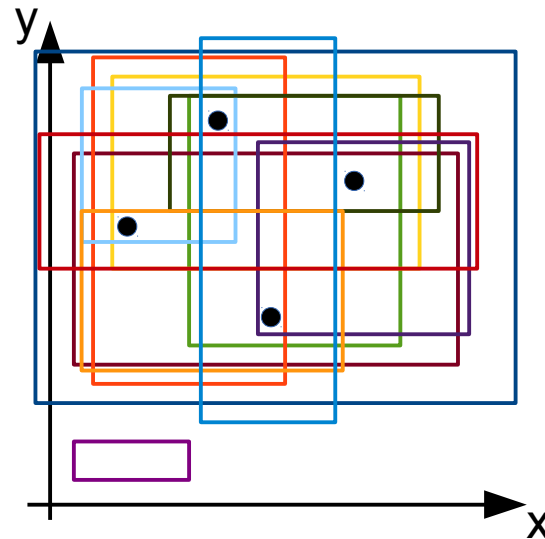
- $B = \mathbb{R}^2$
- \mathcal{R} = all axis-aligned rectangles in \mathbb{R}^2
- Shattering 4 points: **Easy**
 - Take any 4 points s.t. no 3 of them are aligned



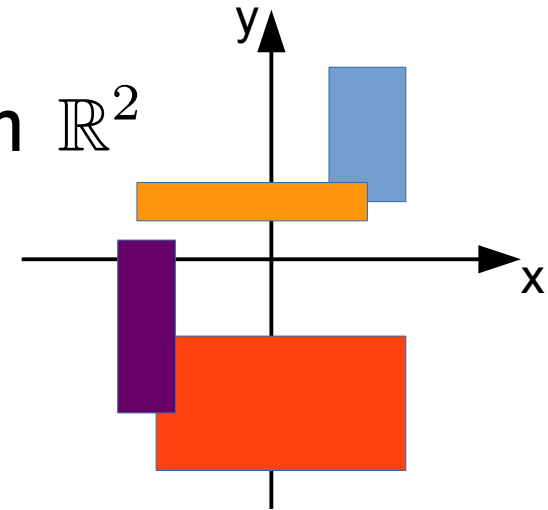


Example of VC-Dimension

- $B = \mathbb{R}^2$
- \mathcal{R} = all axis-aligned rectangles in \mathbb{R}^2
- Shattering 4 points: **Easy**
 - Take any 4 points s.t. no 3 of them are aligned



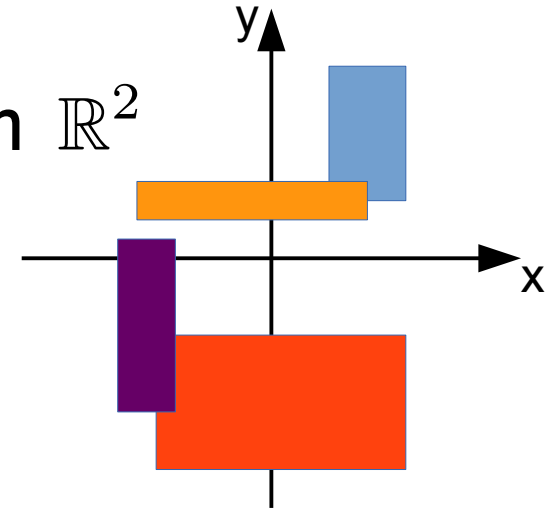
Need 16 rectangles
to shatter them





Example of VC-Dimension

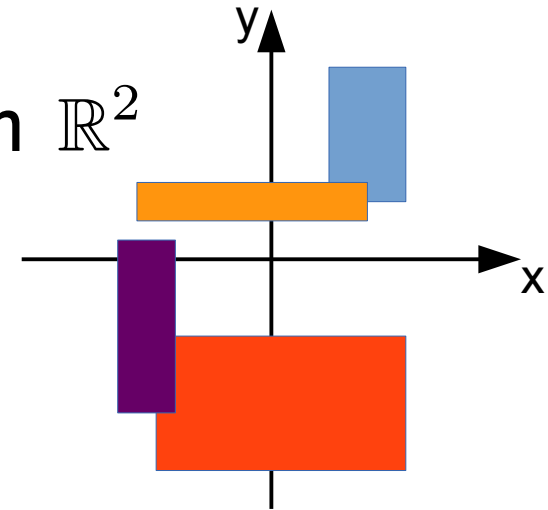
- $B = \mathbb{R}^2$
- $\mathcal{R} =$ all axis-aligned rectangles in \mathbb{R}^2
- Shattering 5 points?





Example of VC-Dimension

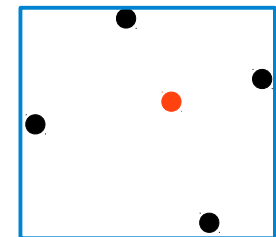
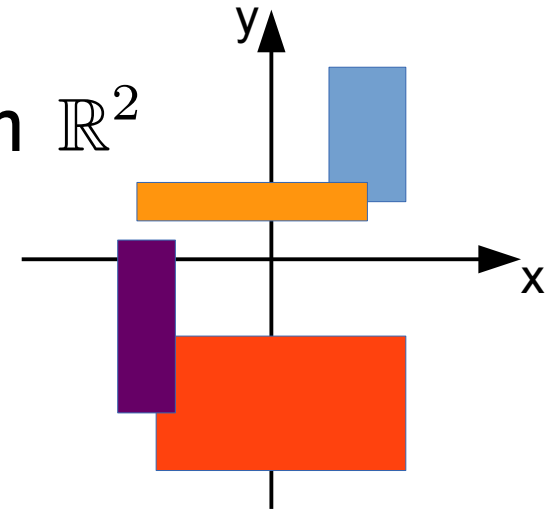
- $B = \mathbb{R}^2$
- \mathcal{R} = all axis-aligned rectangles in \mathbb{R}^2
- Shattering 5 points: impossible





Example of VC-Dimension

- $B = \mathbb{R}^2$
- \mathcal{R} = all axis-aligned rectangles in \mathbb{R}^2
- Shattering 5 points: **impossible**
 - Take any 5 points
 - One of them that is contained in all rectangles containing the other four
 - Impossible to find a rectangle containing only the other four
- $VC(\mathcal{R}) = 4$





VC-Dimension

- [VapnikChervonenkis71]
- Combinatorial property of a collection of subsets from a domain
- Measures the “richness”, “expressivity” of the subsets
- Given a probability distribution on the domain, if we know the VC-dim of a collection of subsets, we can compute the sample size sufficient to approximate the probability mass of each subsets using a sample and the empirical average



Sample Theorem

- Let \mathcal{R} have $VC(\mathcal{R}) \leq d$
- Let π be a probability distribution on B
- Let $\pi(A)$ be the probability mass of $A \subseteq B$
- Given $\varepsilon, \delta \in [0, 1]$, let S be a collection of samples from π

• If

$$|S| \geq \frac{1}{\varepsilon^2} \left(d + \ln \frac{1}{\delta} \right)$$

then,

$$\Pr \left(\exists A \in \mathcal{R} : \left| \pi(A) - \frac{1}{|S|} \sum_{s \in S} 1_A(s) \right| > \varepsilon \right) < \delta$$



Sample Theorem

- Let \mathcal{R} have $VC(\mathcal{R}) \leq d$
- Let π be a probability distribution on B
- Let $\pi(A)$ be the probability mass of $A \subseteq B$
- Given $\varepsilon, \delta \in [0, 1]$, let S be a collection of samples from π

• If

$$|S| \geq \frac{1}{\varepsilon^2} \left(d + \ln \frac{1}{\delta} \right)$$

Empirical Average

then,

$$\Pr \left(\exists A \in \mathcal{R} : \left| \pi(A) - \frac{1}{|S|} \sum_{s \in S} 1_A(s) \right| > \varepsilon \right) < \delta$$



Sample Theorem

- Let \mathcal{R} have $VC(\mathcal{R}) \leq d$
- Let π be a probability distribution on B
- Let $\pi(A)$ be the probability mass of $A \subseteq B$
- Given $\varepsilon, \delta \in [0, 1]$, let S be samples from π

• If

$$|S| \geq \frac{1}{\varepsilon^2} \left(d + \ln \frac{1}{\delta} \right)$$

If d does not depend on $|B|$,
then neither does $|S|$!!!

then,

$$\Pr \left(\exists A \in \mathcal{R} : \left| \pi(A) - \frac{1}{|S|} \sum_{s \in S} 1_A(s) \right| > \varepsilon \right) < \delta$$

Empirical Average



Roadmap

- We are going to build a range set for the problem and show an upper bound to its VC-dimension
- We define a probability distribution and use it to sample elements of the domain
- The sample theorem gives us the amount of samples we need to draw to obtain a good approximation of the betweenness of all vertices



Shortest Path Range Set

- Range set \mathcal{R}_G associated to shortest paths
- Domain: \mathbb{S}_G , all shortest paths in G
- $\mathcal{R}_G = \{\mathcal{T}_v, v \in V\}$
- Contains one range per vertex



Shortest Path Range Set

- Range set \mathcal{R}_G associated to shortest paths
- Domain: \mathbb{S}_G , all shortest paths in G
- $\mathcal{R}_G = \{\mathcal{T}_v, v \in V\}$
All shortest paths v is internal to
- Contains one range per vertex



Shortest Path Range Set

- Range set \mathcal{R}_G associated to shortest paths
- Domain: \mathbb{S}_G , all shortest paths in G
- $\mathcal{R}_G = \{\mathcal{T}_v, v \in V\}$
All shortest paths v is internal to
- Contains one range per vertex
- Sampling **probability distribution** on \mathbb{S}_G
$$\pi_G(p_{uv}) = \frac{1}{n(n-1)} \frac{1}{\sigma_{uv}}$$
- The algorithm samples paths according to π_G



Bounding the VC-Dimension

• **Theorem:** $VC(\mathcal{R}_G) \leq \lfloor \log_2 VD(G) - 2 \rfloor + 1$



Bounding the VC-Dimension

- **Theorem:** $VC(\mathcal{R}_G) \leq \lfloor \log_2 VD(G) - 2 \rfloor + 1$
- **Proof:**



Bounding the VC-Dimension

- **Theorem:** $VC(\mathcal{R}_G) \leq \lfloor \log_2 VD(G) - 2 \rfloor + 1$
- **Proof:** To shatter a set A of paths, $|A| = d$, we need 2^d different ranges $\mathcal{T}_v \in \mathcal{R}_G$



Bounding the VC-Dimension

- **Theorem:** $VC(\mathcal{R}_G) \leq \lfloor \log_2 VD(G) - 2 \rfloor + 1$
- **Proof:** To shatter a set A of paths, $|A| = d$, we need 2^d different ranges $\mathcal{T}_v \in \mathcal{R}_G$
 - Any $p \in A$ must appear in 2^{d-1} different ranges



Bounding the VC-Dimension

- **Theorem:** $VC(\mathcal{R}_G) \leq \lfloor \log_2 VD(G) - 2 \rfloor + 1$
- **Proof:** To shatter a set A of paths, $|A| = d$, we need 2^d different ranges $\mathcal{T}_v \in \mathcal{R}_G$
 - Any $p \in A$ must appear in 2^{d-1} different ranges
 - p appears only in the \mathcal{T}_v 's of the $v \in \text{Int}(p)$



Bounding the VC-Dimension

- **Theorem:** $VC(\mathcal{R}_G) \leq \lfloor \log_2 VD(G) - 2 \rfloor + 1$
- **Proof:** To shatter a set A of paths, $|A| = d$, we need 2^d different ranges $\mathcal{T}_v \in \mathcal{R}_G$
 - Any $p \in A$ must appear in 2^{d-1} different ranges
 - p appears only in the \mathcal{T}_v 's of the $v \in \text{Int}(p)$
 - p appears in $|\text{Int}(p)| \leq VD(G) - 2$ ranges \mathcal{T}_v



Bounding the VC-Dimension

- **Theorem:** $VC(\mathcal{R}_G) \leq \lfloor \log_2 VD(G) - 2 \rfloor + 1$
- **Proof:** To shatter a set A of paths, $|A| = d$, we need 2^d different ranges $\mathcal{T}_v \in \mathcal{R}_G$
 - Any $p \in A$ must appear in 2^{d-1} different ranges
 - p appears only in the \mathcal{T}_v 's of the $v \in \text{Int}(p)$
 - p appears in $|\text{Int}(p)| \leq VD(G) - 2$ ranges \mathcal{T}_v
 - For A to be shattered, must be $2^{d-1} \leq VD(G) - 2$
 - Implies the thesis



Back to the algorithm

- Recall the algorithm:

For r times do

Sample a **random pair** of vertices (u, v)

Compute \mathcal{S}_{uv} , all shortest paths from u to v

Select a path p uniformly at random from \mathcal{S}_{uv}

For each $w \in \text{Int}(p)$

$$\tilde{b}(w) \leftarrow \tilde{b}(w) + 1/r$$



Back to the algorithm

- Recall the algorithm:

For r times do

Sample a **random pair** of vertices (u, v)

Compute \mathcal{S}_{uv} , all shortest paths from u to v

select a path p uniformly at random from \mathcal{S}_{uv}

For each $w \in \text{Int}(p)$

$$\tilde{b}(w) \leftarrow \tilde{b}(w) + 1/r$$

Sampling from π



Back to the algorithm

- Recall the algorithm:

For r times do

Sample a **random pair** of vertices (u, v)

Compute \mathcal{S}_{uv} , all shortest paths from u to v

Select a path p uniformly at random from \mathcal{S}_{uv}

For each $w \in \text{Int}(p)$

$$\tilde{b}(w) \leftarrow \tilde{b}(w) + 1/r$$

Sampling from π

Empirical
Average



Correctness

- From the sample theorem we get that if

$$r \geq \frac{1}{\varepsilon^2} \left(\lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1 + \ln \frac{1}{\delta} \right)$$

then the returned collection of $\tilde{b}(v)$ is an (ε, δ) -approximation:

$$\Pr \left(\exists v \in V : |\tilde{b}(v) - b(v)| > \varepsilon \right) < \delta$$



Correctness

- From the sample theorem we get that if

$$r \geq \frac{1}{\varepsilon^2} \left(\lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1 + \ln \frac{1}{\delta} \right)$$

then the returned collection is an (ε, δ) -approximation:

$$\Pr \left(\exists v \in V : |\tilde{b}(v)| \geq \frac{1}{\varepsilon} \right) \leq \delta$$

In real world social networks,
the diameter is usually very small
(small world phenomenon)

Definitively smaller than n



Corollaries

- If there is a **unique shortest path** for each pair of vertices then $VC(\mathcal{R}_G) \leq 3$



Corollaries

- If there is a **unique shortest path** for each pair of vertices then $VC(\mathcal{R}_G) \leq 3$
- This “**collapse**” of the **VC-dimension** to a constant is somewhat **surprising**
- Suggests that there may be **other characteristic quantities** of the graph that **control the VC-Dimension**



Corollaries

- If there is a **unique shortest path** for each pair of vertices then $VC(\mathcal{R}_G) \leq 3$
 - This “**collapse**” of the **VC-dimension** to a constant is somewhat **surprising**
 - Suggests that there may be **other characteristic quantities** of the graph that **control the VC-Dimension**
- **k-betweenness**: if we only consider shortest paths of size up to k , then

$$VC(\mathcal{R}_G) \leq \lfloor \log_2 k - 1 \rfloor + 1$$



Diameter Approximation

- Computing $VD(G)$ exactly would require APSP
 - Defeat the purpose of sampling



Diameter Approximation

- Computing $VD(G)$ exactly would require APSP
 - Defeat the purpose of sampling
- We need an **approximation** of $VD(G)$
- Can be **constant approx**, we use **\log** anyway!



Diameter Approximation

- Computing $VD(G)$ exactly would require APSP
 - Defeat the purpose of sampling
- We need an **approximation** of $VD(G)$
- Can be **constant approx**, we use **log** anyway!
- **Undirected & unweighted**: **2-approx using BFS**
- All other cases: ???
 - Return the size of the largest WCC ($\leq n$)



Top-K Betweenness

- Often interest is **top-K players** in the network, and in their ranking
- Let $b^{(K)}$ be the K^{th} highest betweenness (ties broken arbitrarily)
- **Goal:**
find the set $T(K, G)$:

$$T(K, G) = \{v \in V : b(v) \geq b^{(K)}\}$$



Top-K Betweenness

- Often interest is **top-K players** in the network, and in their ranking
- Let $b^{(K)}$ be the K^{th} highest betweenness (ties broken arbitrarily)
- **Goal:**
find the set $T(K, G)$:

Similar to definition of
top-K Frequent Itemsets

$$T(K, G) = \{v \in V : b(v) \geq b^{(K)}\}$$



Top-K Betweenness

- “Two phases” sampling algorithm for high-quality approximation of the betweenness of the top-K vertices



Top-K Betweenness

- “Two phases” sampling algorithm for high-quality approximation of the betweenness of the top-K vertices
 - 1st phase: compute lower bound to $b^{(K)}$ using “standard” sample theorem
 - 2nd phase: Use relative variant of the sample theorem to compute superset of $T(K, G)$



Top-K Betweenness

- “Two phases” sampling algorithm for **high-quality approximation** of the betweenness of the top-K vertices
 - **1st phase**: compute **lower bound** to $b^{(K)}$ using “standard” sample theorem
 - **2nd phase**: Use **relative variant** of the sample theorem to compute superset of $T(K, G)$
- For all vertices v in output we have
$$(1 - \varepsilon)b(v) \leq \tilde{b}(v) \leq (1 + \varepsilon)b(v)$$
with probability at least $1 - \delta$



Top-K Betweenness

- “Two phases” sampling algorithm for **high-quality approximation** of the betweenness of the top-K vertices
 - **1st phase**: compute **lower bound** to $b^{(K)}$ using “standard” sample theorem
 - **2nd phase**: Use **relative variance** sample theorem to compute superset of $T(K, G)$
- For all vertices v in output we have
$$(1 - \varepsilon)b(v) \leq \tilde{b}(v) \leq (1 + \varepsilon)b(v)$$
with probability at least $1 - \delta$

Multiplicative factor
(rather than additive)



Tightness

- We proved the upper bound

$$VC(\mathcal{R}_G) \leq \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- Is it **tight**?



Tightness

- We proved the upper bound

$$VC(\mathcal{R}_G) \leq \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- Is it **tight**?

- **YES!**

- There are graphs with

$$VC(\mathcal{R}_G) = \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$



Tightness

- We proved the upper bound

$$VC(\mathcal{R}_G) \leq \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- Is it **tight**?

- **YES!**

- There are graphs with

$$VC(\mathcal{R}_G) = \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- **Example:**



Tightness

- We proved the upper bound

$$VC(\mathcal{R}_G) \leq \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- Is it **tight**?

- **YES!**

- There are graphs with

$$VC(\mathcal{R}_G) = \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- **Example:**

- $\text{VD}(G) = 9$



Tightness

- We proved the upper bound

$$VC(\mathcal{R}_G) \leq \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- Is it **tight**?

- **YES!**

- There are graphs with

$$VC(\mathcal{R}_G) = \lfloor \log_2(\text{VD}(G) - 2) \rfloor + 1$$

- **Example:**

- $\text{VD}(G) = 9$
- $VC(\mathcal{R}_G) = 3$



Outline

- Motivation
- Definition and settings
- Exact and simple sampling algorithms
- Our result: a fast sampling algorithm
- Better approximation for the top-K vertices
- Experiments
- Conclusions



Experimental Evaluation

- **Goals:**

- Evaluate **accuracy** of our algorithms
- Compare **running time, accuracy, and locality** with exact and simple sampling algorithms



Experimental Evaluation

- **Goals:**

- Evaluate **accuracy** of our algorithms
- Compare **running time, accuracy, and locality** with exact and simple sampling algorithms

- **Implementation:**

- **C** extension of **igraph** (igraph.sf.net)
- Exposed through **Python3** API
- Available from GitHub (not yet)



Experimental Evaluation

- **Goals:**

- Evaluate **accuracy** of our algorithms
- Compare **running time, accuracy, and locality** with exact and simple sampling algorithms

- **Implementation:**

- **C** extension of **igraph** (igraph.sf.net)
- Exposed through **Python3** API
- Available from **GitHub** (not yet)

- **Datasets:**

- **Real networks** (social, road, citation, ...) from SNAP (snap.stanford.edu)



Accuracy Results

- Always within ε from real value
- Accuracy even better than guaranteed



Speedup Results

- Almost 1 order of magnitude faster than simple sampling algorithm



Scalability Results

- Scales much better than simple sampling algorithm



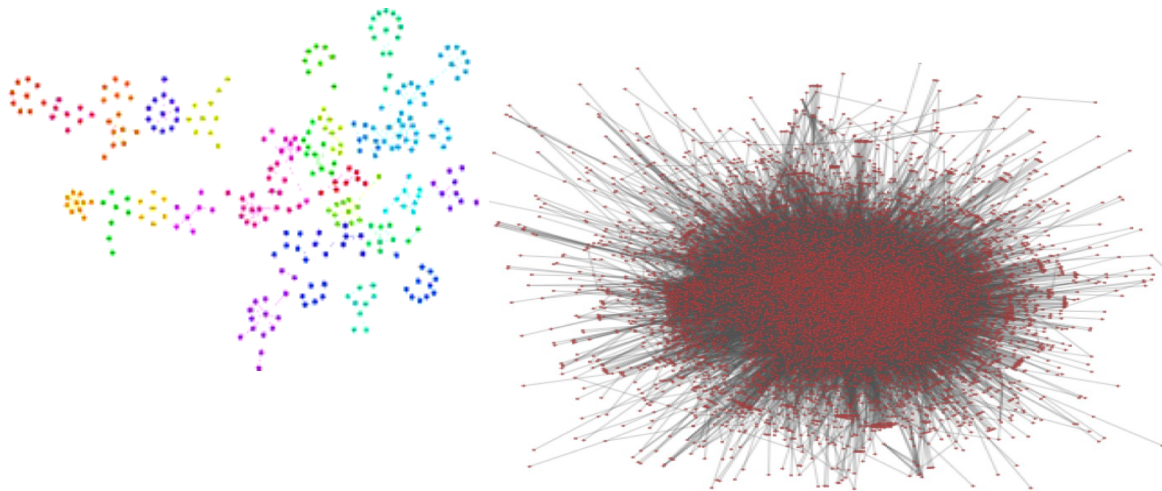
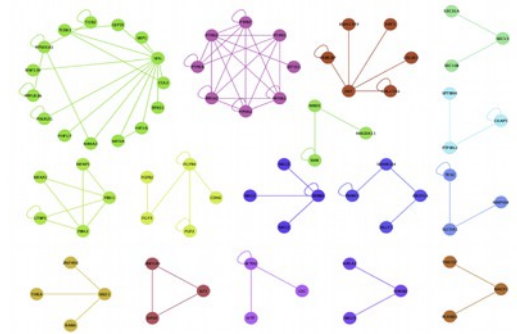
Conclusions

- We presented two **sampling-based randomized algorithms** to **approximate the betweenness of (top-K) vertices** in huge graphs
- The algorithms offer **probabilistic guarantees** on the accuracy of the approximations, using **much fewer samples** and performing **fewer computations** than previous available algorithms
- Experimental evaluation shows that the algorithm **outperforms previous works** in terms of **execution time and accuracy**



The End

- Questions or comments?
- matteo@cs.brown.edu
- <http://bigdata.cs.brown.edu>
- <http://database.cs.brown.edu>





Bibliography

- [Brandes01] *A faster algorithm for betweenness centrality*, J. Math. Sociol., 2001
- [BrandesPich07] *Centrality estimation in large networks*, Intl. J. Bifurc. Chaos, 2007
- [EppsteinWang01] *Fast approximation of centrality*, J. Graph Alg. and App., 2001
- [VapnikChervonenkis71] *On the uniform convergence of relative frequencies of events to their probabilities*, Th. Prob. and its App., 1971