ABRA: Approximating Betweenness Centrality in Static and Dynamic Graphs with Rademacher Averages

MATTEO RIONDATO, Two Sigma Investments, LP
ELI UPFAL, Brown University

ΑΒΡΑΞΑΣ (ABRAXAS): Gnostic word of mystic meaning.

We present ABRA, a suite of algorithms to compute and maintain probabilistically-guaranteed, high-quality, approximations of the betweenness centrality of all nodes (or edges) on both static and fully dynamic graphs. Our algorithms use progressive random sampling and their analysis rely on Rademacher averages and pseudodimension, fundamental concepts from statistical learning theory. To our knowledge, this is the first application of these concepts to the field of graph analysis. Our experimental results show that ABRA is much faster than exact methods, and vastly outperforms, in both runtime and number of samples, state-of-the-art algorithms with the same quality guarantees.

CCS Concepts: •**Mathematics of computing** → **Probabilistic algorithms;** •**Human-centered computing** → **Social networks;** •**Theory of computation** → **Shortest paths; Dynamic graph algorithms; Sketching and sampling; Sample complexity and generalization bounds;**

Additional Key Words and Phrases: Progressive sampling; Pseudodimension; VC-Dimension

ACM Reference Format:
Matteo Riondato, and Eli Upfal. 2016. ABRA: Approximating Betweenness Centrality in Static and Dynamic Graphs with Rademacher Averages. *ACM* XX, X, Article XX (October 2016), 36 pages.
DOI: 0000001.0000001

1. INTRODUCTION

Centrality measures are fundamental concepts in graph analysis: they assign to each node or edge in the network a score that quantifies some notion of importance of the node/edge in the network [Newman 2010]. Betweenness Centrality (BC) is a very popular centrality measure that, informally, defines the importance of a node or edge z in the network as proportional to the fraction of shortest paths in the network that go through z [Anthonisse 1971; Freeman 1977] (see Sect. 3 for formal definitions).

Brandes [2001] presented an algorithm (denoted BA) to compute the exact BC values for all nodes or edges in a graph $G = (V, E)$ in time $O(|V||E|)$ if the graph is unweighted, or time $O(|V||E| + |V|^2 \log |V|)$ if the graph has positive weights. The cost of BA is excessive on modern networks with millions of nodes and tens of millions of edges. Moreover, having the exact BC values may often not be needed, given the exploratory nature of the task, and a high-quality approximation of the values is usually sufficient, provided it comes with stringent guarantees.

Today's networks are not only large, but also *dynamic*: edges are added and removed continuously. Keeping the BC values up-to-date after edge insertions and removals is a challenging task, and proposed algorithms [Green et al. 2012; Kas et al. 2013; Kourtellis et al.

A preliminary version of this work appeared in the proceedings of ACM KDD'16 as [Riondato and Upfal 2016].
This work was supported in part by NSF grant IIS-1247581 and NIH grant R01-CA180776.
Authors' addresses: Eli Upfal, Department of Computer Science, Brown University, email: eli@cs.brown.edu; Matteo Riondato, Labs, Two Sigma Investments LP, email: matteo@twosigma.com.

DOI: 0000001.0000001

2015; Lee et al. 2012; Nasre et al. 2014a,b; Pontecorvi and Ramachandran 2015] may improve the running time for some specific class of input graphs and update models, but in general do not offer worst-case time and space complexities better than from-scratch-recomputation using BA. Maintaining a high-quality approximation up-to-date is more feasible and more *sensible*: there is little informational gain in keeping track of exact BC values that change continuously.

Contributions. We focus on developing algorithms for approximating the BC of all nodes and edges in static and dynamic graphs. Our contributions are the following.

— We present ABRA (for "Approximating Betweenness with Rademacher Averages"), the first family of algorithms based on *progressive sampling* for approximating the BC of all nodes in static and dynamic graphs, where node and edge insertions and deletions are allowed. The BC approximations computed by ABRA are *probabilistically guaranteed* to be within an user-specified additive error ε from their exact values. We also present variants with relative error (i.e., within a multiplicative factor ε of the true value) for the top-k nodes with highest BC, and variants that use refined estimators to give better approximations with a slightly larger sample size.
— Our analysis relies on Rademacher averages [Koltchinskii 2001; Shalev-Shwartz and Ben-David 2014] and pseudodimension [Pollard 1984], fundamental concepts from the field of statistical learning theory [Vapnik 1999]. Building on known and novel results using these concepts, ABRA computes the approximations without having to keep track of any global property of the graph, in contrast with existing algorithms [Bergamini and Meyerhenke 2015; Bergamini et al. 2015; Riondato and Kornaropoulos 2016]. ABRA performs only "real work" towards the computation of the approximations, without having to obtain such global properties or update them after modifications of the graph. To the best of our knowledge, ours is the first application of Rademacher averages and pseudodimension to graph analysis problems, and the first to use *progressive* random sampling for BC computation. Using pseudodimension, we derive new analytical results on the sample complexity of the BC computation task, generalizing previous contributions [Riondato and Kornaropoulos 2016], and formulating a conjecture on the connection between pseudodimension and the distribution of shortest path lengths. Our work hence also showcases the usefulness of these highly theoretical concepts developed in the setting of supervised learning to develop practical algorithms for important problems in unsupervised settings.
— The results of our experimental evaluation on real networks show that ABRA outperforms, in both speed and number of samples, the state-of-the-art methods offering the same guarantees [Riondato and Kornaropoulos 2016].

The present paper extends and improves the conference version [Riondato and Upfal 2016] along multiple directions. The two most relevant additions are *1)* a stricter bound to the maximum approximation error, which allows ABRA's stopping condition to be satisfied at smaller sample sizes than before; *2)* an upper bound to the number of samples needed by ABRA to compute an approximation of the desired quality, which allows ABRA to deterministically stop after the number of samples suggested by the upper bound. We also present all the proofs of our theoretical results, and additional experimental results, which give insights to the betweenness estimation problem and to the behavior of our algorithms. We also added examples throughout the text, with the goal of improving the clarity of the presentation and to make the paper more self-contained.

Outline. We discuss related works in Sect. 2. The formal definitions of the concepts we use in the work can be found in Sect. 3. Our algorithms for approximating BC on static graphs are presented in Sect. 4, while the dynamic case is discussed in Sect. 5. The results of our extensive experimental evaluation are presented in Sect. 6. We draw conclusions in Sect. 7. Additional details can be found in the Appendices.

Table I: Comparison of sample-based algorithms for BC estimation on graphs.

Works	Sample Space	Sample Size for (ε, δ)-approximation *	Analysis Techniques
[Jacob et al. 2005], [Brandes and Pich 2007] [Hayashi et al. 2015]	nodes	$O\left(\frac{1}{\varepsilon^2}\left(\ln\lvert V\rvert + \ln\frac{1}{\delta}\right)\right)$	Hoeffding's ineq., Union bound
[Riondato and Kornaropoulos 2016] [Bergamini and Meyerhenke 2016]	shortest paths	$O\left(\frac{1}{\varepsilon^2}\left(\log_2 \mathsf{VD}(G) + \ln\frac{1}{\delta}\right)\right)$ †	VC-Dimension
This work	pairs of nodes	Variable, at most $O\left(\frac{1}{\varepsilon^2}\left(\log_2 \mathsf{L}(G) + \ln\frac{1}{\delta}\right)\right)$ ‡	Rademacher Avg., Pseudodimension

* See Def. 3.2 for the formal definition.
† $\mathsf{VD}(G)$ is the vertex diameter of the graph G.
‡ $\mathsf{L}(G)$ is the size of the largest weakly connected component of G. See Sect. 4.2 for tighter bounds.

2. RELATED WORK

The definition of Betweenness Centrality comes from the sociology literature [Anthonisse 1971; Freeman 1977], but the study of efficient algorithms to compute it started only when graphs of substantial size became available to the analysts, following the emergence of the Web. The BA algorithm by Brandes [2001] is currently the asymptotically fastest algorithm for computing the exact BC values for all nodes in the network. A number of works also explored heuristics to improve BA [Erdős et al. 2015; Sarıyüce et al. 2013], but retained the same worst-case time complexity.

The use of random sampling to approximate the BC values in static graphs was proposed independently by Jacob et al. [2005] and Brandes and Pich [2007], and successive works explored the tradeoff space of sampling-based algorithms [Bergamini and Meyerhenke 2015, 2016; Bergamini et al. 2015; Riondato and Kornaropoulos 2016]. Other works focused on estimating the betweenness centrality of a single target node, rather than on obtaining uniform guarantees for all the nodes [Bader et al. 2007; Ji and Yan 2016]. We focus here on related works that offer approximation guarantees similar to ours. For an in-depth discussion of previous contributions approximating BC on static graphs, we refer the reader to [Riondato and Kornaropoulos 2016, Sect. 2]. Table I shows a comparison of the sample space, sample size, and analysis techniques for the different works discussed in this section.

Riondato and Kornaropoulos [2016] present algorithms that employ the Vapnik-Chervonenkis (VC) dimension [Vapnik 1999] to compute what is currently the tightest upper bound on the sample size sufficient to obtain guaranteed approximations of the BC of all nodes in a static graph. Their algorithms offer the same guarantees as ABRA but, to compute the sample size, they need to compute an upper bound on a characteristic quantity of the graph (the vertex diameter, namely the maximum number of nodes on any shortest path). Thanks to our use of Rademacher averages in a progressive random sampling setting, ABRA does not need to compute any characteristic quantity of the graph, and instead uses an efficient-to-evaluate stopping condition to determine when the approximated BC values are close to the exact ones. This allows ABRA to use smaller samples and be much faster than the algorithms by Riondato and Kornaropoulos [2016].

A number of works [Green et al. 2012; Kas et al. 2013; Kourtellis et al. 2015; Lee et al. 2012; Nasre et al. 2014a,b; Pontecorvi and Ramachandran 2015] focused on computing the *exact* BC for all nodes in a dynamic graph, taking into consideration different update models. None of these algorithm is provably asymptotically faster than a complete computation from scratch using Brandes' algorithm [Brandes 2001] on general graphs (some of them are faster than BA on some specific classes of input and under some specific update models), and they all require significant amount of space (more details about these works can be found in [Bergamini and Meyerhenke 2015, Sect. 2]). In contrast, Bergamini and Meyerhenke [2015, 2016] built on the work by Riondato and Kornaropoulos [2016] to derive an algorithm for maintaining high-quality approximations of the BC of all nodes when the graph is dynamic and both additions and deletions of edges are allowed. Due to the use of the algorithm by Riondato and Kornaropoulos [2016] as a building block, the algorithm must keep track of the vertex diameter after an update to the graph. Our algorithm for dynamic graphs, instead, does not need this piece of information, and therefore can spend more time in computing the approximations, rather than in keeping track of global properties of the graph. Moreover, our algorithm can handle directed graphs, which is not the case for the algorithms by Bergamini and Meyerhenke [2015, 2016].

Hayashi et al. [2015] recently proposed a data structure called *Hypergraph Sketch* to maintain the shortest path DAGs between pairs of nodes following updates to the graph. Their algorithm uses random sampling and this novel data structure allows them to maintain a high-quality, probabilistically guaranteed approximation of the BC of all nodes in a dynamic graph. Their guarantees come from an application of the simple uniform deviation bounds (i.e., the union bound) to determine the sample size, as previously done by Jacob et al. [2005] and Brandes and Pich [2007]. As a result, the resulting sample size is excessively large, as it depends on the *number of nodes in the graph*. Our improved analysis using the Rademacher averages allows us to develop an algorithm that uses the Hypergraph Sketch with a much smaller number of samples, and is therefore faster.

Progressive random sampling with Rademacher Averages has been used by Elomaa and Kääriäinen [2002] and Riondato and Upfal [2015] in completely different settings, i.e., to train classification trees and to mine frequent itemsets respectively.

3. PRELIMINARIES

We now introduce the formal definitions and basic results that we use throughout the paper.

3.1. Graphs and Betweenness Centrality

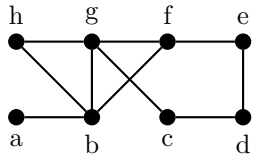
Let $G = (V, E)$ be a graph. G may be directed or undirected and may have non-negative weights on the edges. For any ordered pair (u, v) of different nodes $u \neq v$, let $\mathcal{S}_{uv}$ be the set of *Shortest Paths* (SPs) from u to v, and let $\sigma_{uv} = |\mathcal{S}_{uv}|$. Given a path p between two nodes $u, v \in V$, a node $w \in V$ is *internal to* p if and only if $w \neq u$, $w \neq u$, and p goes through w. We denote as $\sigma_{uv}(w)$ the number of SPs from u to v that w is internal to.

Definition 3.1 (*Betweenness Centrality (*BC*) [Anthonisse 1971; Freeman 1977]*). Given a graph $G = (V, E)$, the *Betweenness Centrality (*BC*) of a node* $w \in V$ is defined as

$$\mathsf{b}(w) = \frac{1}{|V|(|V|-1)} \sum_{\substack{(u,v) \in V \times V \\ u \neq v}} \frac{\sigma_{uv}(w)}{\sigma_{uv}} \quad (\in [0, 1]) \ .$$

An example of a graph and the associated values, taken from [Riondato and Kornaropoulos 2016, Sect. 3] is shown in Fig. 1.

Many variants of BC have been proposed in the literature, including, e.g., one for edges [Newman 2010] and one limited to random walks of a fixed length [Kourtellis et al.



(a) Example graph

(b) Betweenness values

Vertex v	a	b	c	d	e	f	g	h
$\mathsf{b}(v)$	0	0.250	0.125	0.036	0.054	0.080	0.268	0

Fig. 1: Example of betweenness values

2012]. Our results can be extended to many of these variants, following the same discussion as in [Riondato and Kornaropoulos 2016, Sect. 6].

In this work we focus on computing an (ε, δ)-*approximation* of the collection $B = \{\mathsf{b}(w), w \in V\}$.

Definition 3.2 ((ε, δ)-*approximation*). Given $\varepsilon, \delta \in (0, 1)$, an (ε, δ)-*approximation to* B is a collection $\tilde{B} = \{\tilde{\mathsf{b}}(w), w \in V\}$ such that

$$\Pr(\exists w \in v \text{ s.t. } |\tilde{\mathsf{b}}(w) - \mathsf{b}(w)| > \varepsilon) < \delta \ .$$

In other words, with probability at least $1 - \delta$, all BC approximations are within ε from their real value.

In Sect. 4.4 we discuss a relative (i.e., multiplicative) error variant for the top-k nodes with highest BC.

3.2. Rademacher Averages

Rademacher Averages are fundamental concepts to study the rate of convergence of a set of sample averages to their expectations. They are at the core of statistical learning theory [Vapnik 1999] but their usefulness extends way beyond the learning framework [Riondato and Upfal 2015]. We present here only the definitions and results that we use in our work and we refer the readers to, e.g., the book by Shalev-Shwartz and Ben-David [2014] for in-depth presentation and discussion.

While the Rademacher complexity can be defined on an arbitrary measure space, we restrict our discussion here to a sample space that consists of a finite domain $\mathcal{D}$ and a uniform distribution over that domain. Let $\mathcal{F}$ be a family of functions from $\mathcal{D}$ to $[0, 1]$,[1] and let $\mathcal{S} = \{s_1, \ldots, s_\ell\}$ be a collection of ℓ elements from $\mathcal{D}$. For each $f \in \mathcal{F}$, the *true average* and the *sample average* of f on a sample $\mathcal{S}$ are, respectively,

$$\mathsf{m}_{\mathcal{D}}(f) = \frac{1}{|\mathcal{D}|} \sum_{c \in \mathcal{D}} f(c) \quad (= \mathbb{E}[f]) \qquad \text{and} \qquad \mathsf{m}_{\mathcal{S}}(f) = \frac{1}{\ell} \sum_{i=1}^{\ell} f(s_i) \ . \tag{1}$$

Given $\mathcal{S}$, we are interested in bounding the *maximum deviation of* $\mathsf{m}_{\mathcal{S}}(f)$ *from* $\mathsf{m}_{\mathcal{D}}(f)$ *among all* $f \in \mathcal{F}$, i.e., the quantity

$$\sup_{f \in \mathcal{F}} |\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)| \ . \tag{2}$$

[1]The *non-negativity* of the functions in $\mathcal{F}$ is of crucial importance, as many of the results presented in this section are valid *only* for non-negative functions. We show how to extend the results to the case of functions to $[0, b]$ in Appendix A. Extension to intervals of the reals are also possible.

For $1 \leq i \leq \ell$, let λ_i be a Rademacher r.v., i.e., a r.v. that takes value 1 with probability $1/2$ and -1 with probability $1/2$. The r.v.'s λ_i are independent. Consider the quantity

$$\mathsf{R}_{\mathcal{F}}(\mathcal{S}) = \mathbb{E}_{\lambda}\left[\sup_{f\in\mathcal{F}} \frac{1}{\ell}\sum_{i=1}^{\ell} \lambda_i f(s_i)\right], \tag{3}$$

where the expectation is taken only w.r.t. the Rademacher r.v.'s, i.e., conditioning on $\mathcal{S}$. The quantity $\mathsf{R}_{\mathcal{F}}(\mathcal{S})$ is known as the *(conditional) Rademacher average of $\mathcal{F}$ on $\mathcal{S}$*.[2]

The connection between $\mathsf{R}_{\mathcal{F}}(\mathcal{S})$ and the maximum deviation (2) is a key result in statistical learning theory. Classically, e.g., in textbooks and surveys, the connection has been presented using suboptimal bounds that are useful for conveying the intuition behind the connection, but inappropriate for practical use (see, e.g., [Shalev-Shwartz and Ben-David 2014, Thm. 26.5], and compare the bounds presented therein with the ones presented in the following.) Better (i.e., tighter) although more complex bounds are available [Oneto et al. 2013, 2016]. Specifically, we use Thms. 3.3 and 3.4, whose proofs are presented in Appendix A. They extend [Oneto et al. 2013, Thms. 3.10 and 3.11] to a probabilistic tail bound for the supremum of the *absolute value* of the deviation, and specialize them for functions with co-domain $[0, 1]$.

For the rest of this section, let $\mathcal{S}$ be a collection of ℓ elements of $\mathcal{D}$ sampled independently. Let also, for $x \in \mathbb{R}$,

$$\phi(x) = (1+x)\ln(1+x) - x,$$

and, for any $x \in (0, 1)$ and $y \in (2\mathsf{R}_{\mathcal{F}}(\mathcal{S}), 1]$ let

$$g_{\mathcal{S}}(x, y) = 2\exp\left[-2\ell x^2 (y - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))^2\right] + \exp\left[-\ell\left((1-x)y + 2x\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right)\phi\left(\frac{2\mathsf{R}_{\mathcal{F}}(\mathcal{S})}{(1-x)y + 2x\mathsf{R}_{\mathcal{F}}(\mathcal{S})} - 1\right)\right] . \tag{4}$$

THEOREM 3.3. *Let $\eta \in (0, 1)$ and let ξ^* be the optimal value (if it exists) of the minimization problem*

$$\begin{aligned} &\min_{x,\xi} \ \xi \\ &s.t.\ g_{\mathcal{S}}(x, \xi) \leq \eta \\ &\quad \xi \in (2\mathsf{R}_{\mathcal{F}}(\mathcal{S}), 1] \\ &\quad x \in (0, 1) \end{aligned} \tag{5}$$

Then

$$\Pr\left(\sup_{f\in\mathcal{F}} |\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)| \geq \xi^*\right) \leq \eta \ . \tag{6}$$

The bound in (6) is *implicit*, in the sense that the upper bound to the maximum absolute deviation depends on solving a minimization problem. The minimum can be found quite straightforwardly by using the method of Lagrangian multipliers to include the constraint involving $g_{\mathcal{S}}$ in the objective function. The resulting objective function has first and second derivative everywhere in the feasible region, hence, we can efficiently compute the minimum using a root-finding procedure such as Newton's method or higher order methods.

It is also possible to derive an *explicit* bound that does not need any additional numerical procedure (proof in Appendix A).

[2]In this work, we deal, for the most part, with the conditional Rademacher average, rather than with its expectation over the possible samples (which is known as the "Rademacher average", without specializing adjectives). Hence we usually omit the specification "conditional", unless it is needed to avoid confusion.

THEOREM 3.4. *Let $\eta \in (0,1)$. Then, with probability at least $1-\eta$,*

$$\sup_{f\in\mathcal{F}} |\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)| \leq 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}) + 2\frac{\ln\frac{3}{\eta} + \sqrt{\left(\ln\frac{3}{\eta} + 2\ell\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right)\ln\frac{3}{\eta}}}{\ell} + \sqrt{\frac{\ln\frac{3}{\eta}}{2\ell}} \ . \quad (7)$$

Even more refined bounds than the ones presented above are available [Oneto et al. 2016], both implicit and explicit, but, as observed by Oneto et al., in practice they do not seem perform better than the one presented in (6).

Computing, or even estimating, the expectation in (3) w.r.t. the Rademacher r.v.'s is not straightforward and can be computationally expensive, requiring a time-consuming Monte Carlo simulation [Boucheron et al. 2005]. For this reason, *upper bounds to the Rademacher average* are usually employed in (6) and (7) in place of $\mathsf{R}_{\mathcal{F}}(\mathcal{S})$. A powerful and efficient-to-compute bound is presented in Thm. 3.5. Given $\mathcal{S}$, consider, for each $f \in \mathcal{F}$, the vector $\mathbf{v}_{f,\mathcal{S}} = (f(s_1), \ldots, f(s_\ell))$, and let $\mathcal{V}_{\mathcal{S}} = \{\mathbf{v}_{f,\mathcal{S}}, f \in \mathcal{F}\}$ be the *set* of such vectors ($|\mathcal{V}_{\mathcal{S}}| \leq |\mathcal{F}|$, as there may be distinct functions of $\mathcal{F}$ with identical vectors).

THEOREM 3.5. *Let $\mathsf{w} : \mathbb{R}^+ \to \mathbb{R}^+$ be the function*

$$\mathsf{w}(r) = \frac{1}{r}\ln\left(\sum_{\mathbf{v}\in\mathcal{V}_{\mathcal{S}}} \exp\left[\frac{r^2\|\mathbf{v}\|_2^2}{2\ell^2}\right]\right), \quad (8)$$

where $\|\cdot\|_2$ denotes the ℓ_2-norm (Euclidean norm). Then

$$\mathsf{R}_{\mathcal{F}}(\mathcal{S}) \leq \min_{r\in\mathbb{R}^+} \mathsf{w}(r) \ . \quad (9)$$

The function w is convex, continuous in $\mathbb{R}^+$, and has first and second derivatives w.r.t. r everywhere in its domain, so it is possible to minimize it efficiently using standard convex optimization methods [Boyd and Vandenberghe 2004]. More refined bounds can be derived but are more computationally expensive to compute [Anguita et al. 2014].

4. APPROXIMATING BETWEENNESS CENTRALITY IN STATIC GRAPHS

We now present and analyze ABRA-s, our *progressive sampling algorithm* for computing an (ε,δ)-approximation to the collection of exact BC values in a *static* graph. Many of the details and properties of ABRA-s are shared with the other ABRA algorithms we present in later sections.

Progressive Sampling. Progressive sampling algorithms are intrinsically *iterative*. At a high level, they work as follows. At iteration i, the algorithm extracts an approximation of the values of interest (in our case, of the BC of all nodes) from a collection $\mathcal{S}_i$ of $S_i = |\mathcal{S}_i|$ random samples from a suitable domain $\mathcal{D}$ (in our case, the samples are pairs of different nodes). Then, the algorithm checks a specific *stopping condition* that uses information obtained from the sample $\mathcal{S}_i$ and from the computed approximation. If the stopping condition is satisfied, then the approximation has, with the required probability, the desired quality (in our case, it is an (ε,δ)-approximation). The approximation is then returned in output and the algorithm terminates. If the stopping condition is not satisfied, the algorithm builds a collection $\mathcal{S}_{i+1}$ by adding random samples to $\mathcal{S}_i$ until it has size S_{i+1}. Then it computes a new approximation from the so-created collection $\mathcal{S}_{i+1}$, and checks the stopping condition again and so on.

There are two main challenges for the designer of progressive sampling algorithm: deriving a "good" stopping condition and determining good choices for the initial sample size S_1 and the subsequent sample sizes S_{i+1}, $i \geq 1$.

An ideal stopping condition is such that:

(1) when satisfied, it guarantees that the computed approximation has the desired quality properties (in our case, it is an (ε, δ)-approximation); and
(2) it can be evaluated efficiently; and
(3) it is "weak", in the sense that is satisfied at small sample sizes.

The stopping condition for ABRA-s (presented in the following) is based on Thm. 3.4 and Thm. 3.5, and has all the above desirable properties.

The second challenge is determining the *sample schedule* $(S_i)_{i>0}$. Any monotonically increasing sequence of positive numbers can act as sample schedule, but the goal in designing a good sample schedule is to minimize the number of iterations that are needed before the stopping condition is satisfied, while minimizing the sample size S_i at the iteration i at which this happens. The sample schedule may be fixed in advance, but an *adaptive approach* that ties the sample schedule to the stopping condition can give better results, as the sample size S_{i+1} for iteration $i+1$ can be computed using information obtained in (or up-to) iteration i. We developed a general adaptive approach to compute the sample schedule which can be used also in other progressive sampling algorithms and is not specific to ABRA (see Sect. 4.1.1.)

4.1. Algorithm Description and Analysis

ABRA-s takes as input a graph $G = (V, E)$, which may be directed or undirected, and may have non-negative weights on the edges, and two parameters $\varepsilon, \delta \in (0, 1)$. It outputs a collection $\widetilde{B} = \{\widetilde{\mathsf{b}}(w), w \in V\}$ that is an (ε, δ)-approximation of the betweenness centralities $B = \{\mathsf{b}(w), w \in V\}$. Let$\mathcal{D} = \{(u, v) \in V \times V, u \neq v\}$. For each node $w \in V$, let $f_w : \mathcal{D} \to [0, 1]$ be the function

$$f_w(u, v) = \frac{\sigma_{uv}(w)}{\sigma_{uv}}, \tag{10}$$

i.e., $f_w(u, v)$ is the fraction of shortest paths (SPs) from u to v that go through w (i.e., that w is internal to.) Let $\mathcal{F} = \{f_w, w \in V\}$ be the set of these functions. Given this definition, we have that

$$\mathsf{m}_{\mathcal{D}}(f_w) = \frac{1}{|\mathcal{D}|} \sum_{(u,v)\in\mathcal{D}} f_w(u, v) = \frac{1}{|V|(|V|-1)} \sum_{\substack{(u,v)\in V\times V \\ u\neq v}} \frac{\sigma_{uv}(w)}{\sigma_{uv}} = \mathsf{b}(w) \ .$$

The intuition behind ABRA-s is the following. Let $\mathcal{S} = \{(u_i, v_i), 1 \leq i \leq \ell\}$ be a collection of ℓ pairs (u, v) sampled independently and uniformly from $\mathcal{D}$. For the sake of clarity, we define

$$\widetilde{\mathsf{b}}(w) = \mathsf{m}_{\mathcal{S}}(f_w) = \frac{1}{\ell} \sum_{i=1}^{\ell} f_w(u_i, v_i) = \frac{1}{\ell} \sum_{i=1}^{\ell} \frac{\sigma_{u_i v_i}(w)}{\sigma_{u_i v_i}} \ .$$

For each $w \in V$ consider the vector

$$\mathbf{v}_w = (f_w(u_1, v_1), \ldots, f_w(u_\ell, v_\ell)) \ .$$

It is easy to see that $\widetilde{\mathsf{b}}(w) = \|\mathbf{v}_w\|_1/\ell$. Let now $\mathcal{V}_{\mathcal{S}}$ be the *set* of these vectors:

$$\mathcal{V}_{\mathcal{S}} = \{\mathbf{v}_w, w \in V\} \ .$$

It is possible, if not likely, that $|\mathcal{V}_{\mathcal{S}}| \leq |V|$ as there may be two different nodes u and v with $\mathbf{v}_u = \mathbf{v}_v$. If we have complete knowledge of the set $\mathcal{V}_{\mathcal{S}}$ (i.e., of its elements), then we can compute the quantity

$$\omega^* = \min_{r\in\mathbb{R}^+} \frac{1}{r} \ln\left(\sum_{\mathbf{v}\in\mathcal{V}_{\mathcal{S}}} \exp\left[\frac{r^2 \|\mathbf{v}\|_2^2}{2\ell^2} \right] \right),$$

which, from Thm. 3.5, is an *upper bound* to $\mathsf{R}_{\mathcal{F}}(\mathcal{S})$. We can use ω^* to obtain an upper bound $\xi_{\mathcal{S}}$ to the supremum of the absolute deviation by either plugging ω^* in (4) and solve the minimization problem (5), or by just plugging ω^* in (7). It follows from Thm. 3.3 or Thm. 3.4 that the collection $\widetilde{B} = \{\widetilde{\mathsf{b}}(w) = \|\mathbf{v}_w\|_1/\ell, w \in V\}$ is an $(\xi_{\mathcal{S}}, \eta)$-approximation to the exact betweenness values.

ABRA-s builds on this intuition and works as follows. Suppose for now that we fix a sample schedule a priori, i.e., fix a monotonically increasing sequence $(S_i)_{i>0}$ of sample sizes (we show in Sect. 4.1.1 how to compute the sample schedule adaptively on the fly). The algorithm builds a collection $\mathcal{S}$ by sampling pairs (u, v) independently and uniformly at random from $\mathcal{D}$ until $\mathcal{S}$ has size S_1. After each pair of nodes has been sampled, ABRA-s performs an $s - t$ SP computation from u to v and then backtracks from v to u along the SPs just computed, to keep track of the set $\mathcal{V}_{\mathcal{S}}$ of vectors (details given below). For clarity of presentation, let $\mathcal{S}_1$ denote $\mathcal{S}$ when it has size exactly S_1, and analogously for $\mathcal{S}_i$ and S_i, $i > 1$. Once $\mathcal{S}_i$ has been "built", ABRA-s computes $\xi_{\mathcal{S}_i}$ as described earlier, using $\eta = \delta/(3 \cdot 2^i)$. It then checks whether $\xi_{\mathcal{S}_i} \leq \varepsilon$. This is ABRA-s *stopping condition*: when it holds, ABRA-s returns

$$\widetilde{B} = \{\widetilde{\mathsf{b}}(w) = \|\mathbf{v}_w\|_1/S_i, w \in V\} \ .$$

Otherwise, ABRA-s iterates and continues adding samples from $\mathcal{D}$ to $\mathcal{S}$ until it has size S_2, and so on until $\eta_{\mathcal{S}_i} \leq \varepsilon$ holds. The pseudocode for ABRA-s is presented in Alg. 1, including the steps to update $\mathcal{V}_{\mathcal{S}}$, described in the following, and to adaptively choose the sample schedule (see Sect. 4.1.1). For clarity of the presentation, the pseudocode computes $\xi_{\mathcal{S}_i}$ by plugging ω_i^* in (7).

We now prove the correctness of the algorithm.

THEOREM 4.1 (CORRECTNESS). *The collection $\widetilde{B}$ returned by **ABRA-s** is a (ε, δ)-approximation.*

PROOF. The claim follows from the definitions of $\mathcal{S}$, $\mathcal{V}_{\mathcal{S}}$, $\mathcal{F}$, f_w for $w \in V$, $\widetilde{\mathsf{b}}(w)$, $\xi_{\mathcal{S}_i}$, Thm. 3.5, and from the fact that, at the iteration i, it holds, from Thm. 3.3 or Thm. 3.4, that

$$\Pr\left(\sup_{f \in \mathcal{F}} |\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)| \geq \xi_{\mathcal{S}_i}\right) \leq \frac{\delta}{2^i},$$

hence by taking an union bound over all iterations, we obtain the thesis. □

Computing and maintaining the set $\mathcal{V}_{\mathcal{S}}$. We now discuss in details how ABRA-s efficiently maintain the set $\mathcal{V}_{\mathcal{S}}$ of vectors, which is used to compute the value $\xi_{\mathcal{S}}$ and the values $\widetilde{\mathsf{b}}(w) = \|\mathbf{v}_w\|_1/|\mathcal{S}|$ in $\widetilde{B}$. In addition to $\mathcal{V}_{\mathcal{S}}$, ABRA-s also maintains a map M from V to $\mathcal{V}_{\mathcal{S}}$ (i.e., $M[w]$ is a vector $\mathbf{v}_w \in \mathcal{V}_{\mathcal{S}}$), and a counter $\mathsf{c}_{\mathbf{v}}$ for each $\mathbf{v} \in \mathcal{V}_{\mathcal{S}}$, denoting how many nodes $w \in V$ have $M[w] = \mathbf{v}$.

At the beginning of the execution of the algorithm, $\mathcal{S} = \emptyset$ and $\mathcal{V}_{\mathcal{S}} = \emptyset$. Nevertheless, ABRA-s initializes $\mathcal{V}_{\mathcal{S}}$ to contain one special empty vector $\mathbf{0}$, with no components, and M so that $M[w] = \mathbf{0}$ for all $w \in V$, and $\mathsf{c}_{\mathbf{0}} = |V|$ (lines 3 and following in Alg. 1).

After having sampled a pair (u, v) from $\mathcal{D}$, ABRA-s updates $\mathcal{V}_{\mathcal{S}}$, M and the counters as follows. First, it performs (line 11) a $s - t$ SP computation from u to v using any SP algorithm (e.g., BFS, Dijkstra, or even any bidirectional search SP algorithm) modified, as discussed by Brandes [2001, Lemma 3], to keep track, for each node w encountered during the computation, of the SP distance $\mathsf{d}(u, w)$ from u to w, of the number σ_{uw} of SPs from u to w, and of the set $\mathsf{P}_u(w)$ of (immediate) predecessors of w along the SPs from u.[3] Once

[3]Storing the set of immediate predecessors is not necessary. By not storing it, we can reduce the space complexity from $O(|E|)$ to $O(|V|)$, at the expense of some additional computation at runtime.

ALGORITHM 1: ABRA-s: absolute error approximation of BC on static graphs

input : Graph $G = (V, E)$, accuracy parameter $\varepsilon \in (0, 1)$, confidence parameter $\delta \in (0, 1)$
output: Set $\widetilde{B}$ of BC approximations for all nodes in V

1 $\mathcal{D} \leftarrow \{(u, v) \in V \times V, u \neq v\}$
2 $S_0 \leftarrow 0$, $S_1 \leftarrow \frac{(1+16\varepsilon+\sqrt{1+32\varepsilon})\ln(3/\delta)}{4\varepsilon^2}$
3 $\mathbf{0} = (0)$
4 $\mathcal{V} = \{\mathbf{0}\}$
5 **foreach** $w \in V$ **do** $M[w] = \mathbf{0}$
6 $\mathtt{c_0} \leftarrow |V|$
7 $i \leftarrow 1$, $j \leftarrow 1$
8 **while** *True* **do**
9 **for** $\ell \leftarrow 1$ **to** $S_i - S_{i-1}$ **do**
10 $(u, v) \leftarrow$ `uniform_random_sample(`$\mathcal{D}$`)`
11 `compute_SPs(`u, v`)` `//Truncated SP computation`
12 **if** *reached* v **then**
13 **foreach** $z \in \mathsf{P}_u[v]$ **do** $\sigma_{zv} \leftarrow 1$
14 **foreach** *node* w *on a SP from* u *to* v*, in reverse order by* $\mathtt{d}(u, w)$ **do**
15 $\sigma_{uv}(w) \leftarrow \sigma_{uw}\sigma_{wv}$
16 $\mathbf{v} \leftarrow M[w]$
17 $\mathbf{v}' \leftarrow \mathbf{v} \cup \{(j, \sigma_{uv}(w))\}$
18 **if** $\mathbf{v}' \notin \mathcal{V}$ **then**
19 $c_{\mathbf{v}'} \leftarrow 1$
20 $\mathcal{V} \leftarrow \mathcal{V} \cup \{\mathbf{v}'\}$
21 **else** $\mathtt{c}_{\mathbf{v}'} \leftarrow \mathtt{c}_{\mathbf{v}'} + 1$
22 $M[w] \leftarrow \mathbf{v}'$
23 **if** $\mathtt{c}_{\mathbf{v}} > 1$ **then** $\mathtt{c}_{\mathbf{v}} \leftarrow \mathtt{c}_{\mathbf{v}} - 1$
24 **else** $\mathcal{V} \leftarrow \mathcal{V} \setminus \{\mathbf{v}\}$
25 **foreach** $z \in \mathsf{P}_u[w]$ **do** $\sigma_{zv} \leftarrow \sigma_{zv} + \sigma_{wv}$
26 **end**
27 **end**
28 $j \leftarrow j + 1$
29 **end**
30 $\omega_i^* \leftarrow \min_{r \in \mathbb{R}^+} \frac{1}{r} \ln\left(\sum_{\mathbf{v} \in \mathcal{V}_\mathcal{S}} \exp\left[r^2\|\mathbf{v}\|^2/(2S_i^2)\right]\right)$
31 $\gamma_i \leftarrow \ln(3/\delta) + i\ln 2$
32 $\xi_{\mathcal{S}_i} \leftarrow 2\omega_i^* + 2\frac{\gamma_i + \sqrt{\gamma_i(\gamma_i + 2S_i\omega_i^*)}}{S_i} + \sqrt{\frac{\gamma_i}{2S_i}}$
33 **if** $\xi_{\mathcal{S}_i} \leq \varepsilon$ **then**
34 **break**
35 **else**
36 $S_{i+1} \leftarrow$ `nextSampleSize()`
37 $i \leftarrow i + 1$
38 **end**
39 **end**
40 **return** $\widetilde{B} \leftarrow \{\widetilde{\mathsf{b}}(w) \leftarrow \|M[w]\|_1/S_i, w \in V\}$

v has been reached (and only if it has been reached), the algorithm starts backtracking from v towards u along the SPs it just computed (line 14). During this backtracking, the algorithm visits the nodes along the SPs in *inverse* order of SP distance from u, ties broken arbitrarily. For each visited node w different from u and v, ABRA-s computes the value

$f_w(u,v) = \sigma_{uv}(w)$ of SPs from u to v that go through w, which is obtained as

$$\sigma_{uv}(w) = \sigma_{uw} \times \sum_{z \,:\, w \in \mathsf{P}_u(z)} \sigma_{zv}$$

where the value σ_{uw} is obtained during the $s-t$ SP computation, and the values σ_{zw} are computed recursively during the backtracking (line 25), as described by Brandes [2001]. After computing $\sigma_{uv}(w)$, the algorithm takes the vector $\mathbf{v} \in \mathcal{V}_\mathcal{S}$ such that $M[w] = \mathbf{v}$ and creates a new vector $\mathbf{v}'$ by appending $\sigma_{uv}(w)$ to the end of $\mathbf{v}$.[4] Then it adds $\mathbf{v}'$ to the *set* $\mathcal{V}_\mathcal{S}$, updates $M[w]$ to $\mathbf{v}'$, and increments the counter $\mathsf{c}_{\mathbf{v}'}$ by one (lines 16 to 22). Finally, the algorithm decrements the counter $\mathsf{c}_\mathbf{v}$ by one, and if $\mathsf{c}_\mathbf{v}$ becomes equal to zero, ABRA-s removes $\mathbf{v}$ from $\mathcal{V}_\mathcal{S}$ (line 24). At this point, the algorithm moves to analyzing another node w' with distance from u less or equal to the distance of w from u. It is easy to see that when the backtracking reaches u, the set $\mathcal{V}_\mathcal{S}$, the map M, and the counters, have been correctly updated.

We remark that to compute $\xi_{\mathcal{S}_i}$ and $\widetilde{B}$ and to keep the map M up to date, ABRA-s does not actually need to store the vectors in $\mathcal{V}_\mathcal{S}$ (even in sparse form), but it is sufficient to maintain their ℓ_1- and ℓ_2 (i.e., Euclidean) norms, which require much less space.

4.1.1. Computing the sample schedule. We now discuss how to compute the initial sample size S_1 at the beginning of ABRA-s (line 2 of Alg. 1) and the sample size S_{i+1} at the end of iteration i of the main loop (line 36). We remark that any sample schedule $(S_i)_{i>0}$ can be used, and our method is an *heuristic* that nevertheless aims at making use of all available information at the end of each iteration to the most possible extent, with the goal of increasing the chances that the stopping condition is satisfied at the next iteration.

A reasonable initial sample size S_1 is

$$S_1 \geq \frac{(1 + 16\varepsilon + \sqrt{1 + 32\varepsilon})\ln(6/\delta)}{4\varepsilon^2} \ . \tag{11}$$

To understand the intuition behind this choice, recall (7), and consider that, at the beginning of the algorithm, there is obviously no information available about $\mathsf{R}_\mathcal{F}(\mathcal{S}_1)$, except that it is *non-negative*. It follows that, for the r.h.s. of (7) to be at most ε at the end of the first iteration (i.e., for the stopping condition to be satisfied at this time), it is necessary that

$$\frac{4\ln(6/\delta)}{S_1} + \sqrt{\frac{\ln(6/\delta)}{2S_1}} \leq \varepsilon \ .$$

Solving for S_1 under the domain constraints $S_1 \geq 1$, $\delta \in (0,1)$, $\varepsilon \in (0,1)$, gives the unique solution in (11). This choice for the initial sample size is an heuristic because it is actually possible that the stopping condition is satisfied at smaller sample condition when using (5) to compute an upper bound to the Rademacher averages.

Computing the next sample size S_{i+1} at the end of iteration i (in the pseudocode in Alg. 1, this is done by calling `nextSampleSize()` on line 36) is slightly more involved. The intuition is to assume that ω_i^*, which is an upper bound on $\mathsf{R}_\mathcal{F}(\mathcal{S}_i)$, is also an upper bound on $\mathsf{R}_\mathcal{F}(\mathcal{S}_{i+1})$, whatever $\mathcal{S}_{i+1}$ will be, and whatever size it may have. At this point, we can ask what is the minimum size $S_{i+1} = |\mathcal{S}_{i+1}|$ for which $\xi_{\mathcal{S}_{i+1}}$ would be at most ε, under the assumption that $\mathsf{R}_\mathcal{F}(\mathcal{S}_{i+1}) \leq \omega_i^*$. More formally, let

$$\gamma_{i+1} = \ln\frac{3}{\delta} + (i+1)\ln 2,$$

[4] ABRA-s uses a sparse representation for the vectors $\mathbf{v} \in \mathcal{V}_\mathcal{S}$, storing only the non-zero components of each $\mathbf{v}$ as pairs (j,g), where j is the component index and g is the value of that component.

we want to solve the inequality

$$\sqrt{\frac{\gamma_{i+1}}{2S_{i+1}}} + 2\gamma_{i+1}\frac{\sqrt{\gamma_{i+1}(\gamma_{i+1} + 2S_{i+1}\omega_i^*}}{S_{i+1}} + 2\omega_i^* \leq \varepsilon \tag{12}$$

where S_{i+1} acts as the unknown. The l.h.s. of this inequality is obtained by using, in (7) ω_i^* in place of $\mathsf{R}_{\mathcal{F}}(\mathcal{S})$, S_{i+1} in place of ℓ, $\delta/2^{i+1}$ in place of η , and slightly reorganize the terms for readability. As can be verified using a symbolic mathematical computation program, finding the solution to the above inequality requires computing the roots of the fourth grade polynomial (in x)

$$\begin{aligned}
&64\gamma_{i+1}^8 + (-16\gamma_{i+1}^5 + 256\omega_i^*\gamma_{i+1}^7)x + \\
&\left(\gamma_{i+1}^2 - 32\omega_i^*\gamma_{i+1}^4 - 128(\omega_i^*)^2\gamma_{i+1}^4 + 128\omega_i^*\varepsilon\gamma_{i+1}^4 - 32\varepsilon^2\gamma_{i+1}^4 + 256(\omega_i^*)^2\gamma_{i+1}^6\right)x^2 + \\
&\left(-16(\omega_i^*)^2\gamma_{i+1} + 16\omega_i^*\varepsilon\gamma_{i+1} - 4\varepsilon^2\gamma_{i+1} - 256(\omega_i^*)^3\gamma_{i+1}^3 + 256(\omega_i^*)^2\varepsilon\gamma_{i+1}^3 - 64\omega_i^*\varepsilon^2\gamma_{i+1}^3\right)x^3 + \\
&\left(64(\omega_i^*)^4 - 128(\omega_i^*)^3\varepsilon + 96(\omega_i^*)^2\varepsilon^2 - 32\omega_i^*\varepsilon^3 + 4\varepsilon^4\right)x^4 \ .
\end{aligned}$$

ABRA-s computes the four roots using a zeroes-finding algorithm such as the Jenkins-Traub algorithm [Jenkins and Traub 1972]), and sets S_{i+1} equal to the smallest root larger than S_i.

The assumption $\mathsf{R}_{\mathcal{F}}(\mathcal{S}_{i+1}) \leq \omega_i^*$, which is not guaranteed to be true, is what makes this procedure for selecting the next sample size an *heuristics*. Nevertheless, using information available at the current iteration to compute the sample size for the next iteration is more sensible than having a fixed sample schedule, as it tunes the growth of the sample size to the quality of the current sample. Moreover, it removes from the user the burden of choosing a sample schedule, effectively eliminating one parameter of the algorithm.

4.2. Upper bounds on the number of samples

It is natural to ask whether, given a graph $G = (V, E)$, there exists an integer s such that ABRA-s can stop and output $\tilde{B}$ after having sampled s pair of nodes and $\tilde{B}$ will be an (ε, δ)-approximation, independently from whether the stopping condition is satisfied or not at that point in the execution. If such a sample size s exists, we can modify the stopping condition of ABRA-s to just stop after having examined a sample of that size, as we describe in Sect. 4.2.1. Such a sample size exists and it is a function of a characteristic quantity of the graph G and of ε and δ. Its derivation and correctness analysis use *pseudodimension* [Pollard 1984], an extension of the Vapnik-Chervonenkis dimension to real-valued functions. A short introduction on pseudodimension can be found in Appendix B. The fundamental result that we use is that having an upper bound on the pseudodimension allows to bound the supremum of the deviations from (2), as stated in the following theorem.

THEOREM 4.2 ([LI ET AL. 2001]). *Let D be a domain and $\mathcal{F}$ be a family of functions from D to $[0, 1]$. Let $\mathsf{PD}(\mathcal{F}) \leq d$. Given $\varepsilon, \eta \in (0, 1)$, let $\mathcal{S}$ be a collection of elements sampled independently and uniformly at random from D, with size*

$$|\mathcal{S}| = \frac{c}{\varepsilon^2}\left(d + \log\frac{1}{\eta}\right) \ . \tag{13}$$

Then

$$\Pr\left(\exists f \in \mathcal{F} \ s.t. \ |\mathsf{m}_D(f) - \mathsf{m}_{\mathcal{S}}(f)| > \varepsilon\right) < \eta \ .$$

The constant c is universal.

4.2.1. Using the upper bounds in the stopping condition. There are two ways of modifying the stopping rule of ABRA-s to use Thm. 4.2 and the upper bounds to the pseudodimension presented in the following subsections:

Fixed size. Before running ABRA-s, we compute an upper bound d to the pseudodimension, by finding the weakly connected components of the graph G' in time $O(|E|+|V|)$. Let then δ_1 and δ_2 be such that $\delta_1 + \delta_2 = \delta$. We compute a sample size S_p using (13) with the computed pseudodimension upper bound d and $\eta = \delta_1$. Then we run ABRA-s using δ_2 instead of δ, and with the additional stopping condition to return the current BC approximation when the sample reaches size S_p. The fact that returned collection is an (ε, δ)-approximation comes from Thms. 4.1 and 4.2 and an application of the union bound.
Iteration-dependent size. After having computed an upper bound d for the pseudodimension as before, we can use it at the end of each iteration i by computing the sample size to be used at the next iteration as the minimum between the sample size obtained from the sample schedule (either automatic or not) and the sample sized obtained by plugging d and $\eta = \delta/(3 \cdot 2^i)$ into (13). The correctness follows easily from Thms. 4.1 and 4.2.

4.2.2. General Cases. We now show a general upper bound on the pseudodimension of $\mathcal{F}$. The derivation of this upper bound follows the one for VC-Dimension in [Riondato and Kornaropoulos 2016, Sect. 4], adapted to our settings.

Let $G = (V, E)$ be a graph, and consider the family

$$\mathcal{F} = \{f_w, w \in V\}$$

where f_w goes from $\mathcal{D} = \{(u, v) \in V \times V, u \neq v\}$ to $[0, 1]$ and is defined in (10). The rangeset $\mathcal{F}^+$ contains one range R_w for each node $w \in V$. The set $R_w \subseteq \mathcal{D} \times [0, 1]$ contains pairs in the form $((u, v), x)$, with $(u, v) \in \mathcal{D}$ and $x \in [0, 1]$. The pairs $((u, v), x) \in R_w$ with $x > 0$ are all and only the pairs in this form such that

(1) w is on a SP from u to v; and
(2) $x \leq \sigma_{uv}(w)/\sigma_{uv}$.

For any SP p let $\mathsf{Int}(p)$ be the *set* of nodes that are internal to p, i.e., not including the extremes of p. For any pair (u, v) of distinct nodes, let

$$N_{uv} = \bigcup_{p \in \mathcal{S}_{uv}} \mathsf{Int}(p)$$

be the set of nodes in the SP DAG from u to v, *excluding u and v*, and let $s_{uv} = |N_{uv}|$. Let $\mathsf{H}(G)$ be the maximum integer h such that there are at least $\lfloor \log_2 h \rfloor + 1$ pairs (u, v) such that is $s_{uv} \geq h$. Except in trivial cases, $\mathsf{H}(G) > 0$.

THEOREM 4.3. *We have* $\mathsf{PD}(\mathcal{F}) \leq \lfloor \log_2 \mathsf{H}(G) \rfloor + 1$.

PROOF. Let $k > \lfloor \log_2 \mathsf{H}(G) \rfloor + 1$ and assume for the sake of contradiction that $\mathsf{PD}(\mathcal{F}) = k$. From the definition of pseudodimension, we have that there is a set Q of k elements of the domain of $\mathcal{F}^+$ that is shattered.

From the definition of $\mathsf{H}(G)$ and from Lemma B.1, we have that Q must contain an element $a = ((u, v), x)$, $x > 0$, of the domain of $\mathcal{F}^+$ such that $s_{uv} < \mathsf{H}(G)$.

There are 2^{k-1} non-empty subsets of Q containing a. Let us label these non-empty subsets of Q containing a as $S_1, \ldots, S_{2^{k-1}}$, where the labelling is arbitrary. Given that Q is shattered, for each set S_i there must be a range R_i in $\mathcal{F}^+$ such that $S_i = Q \cap R_i$. Since all the S_i's are different from each other, then all the R_i's must be different from each other. Given that a is a member of every S_i, a must also belong to each R_i, that is, there are 2^{k-1} distinct ranges in $\mathcal{F}^+$ containing a. But a belongs only to (not necessarily all) the ranges corresponding to nodes in N_{uv}. This means that a belongs to at most s_{uv} ranges in $\mathcal{F}^+$.

But $s_{uv} < \mathsf{H}(G)$ by definition of $\mathsf{H}(G)$, so p can belong to at most $\mathsf{H}(G)$ ranges from $\mathcal{R}_G$. Given that $2^{k-1} > \mathsf{H}(G)$, we reached a contradiction and there cannot be 2^{k-1} distinct ranges containing a, hence not all the sets S_i can be expressed as $Q \cap R_i$ for some $R_i \in \mathcal{F}^+$.

Then Q cannot be shattered and we have

$$\mathsf{PD}(\mathcal{F}) = \mathsf{VC}(\mathcal{F}^+) \leq \lfloor \log_2 \mathsf{H}(G) \rfloor + 1 \ .$$

□

Computing $\mathsf{H}(G)$ exactly is not practical as it would defeat the purpose of using sampling. Instead, we now present looser but efficient-to-compute upper bounds on the pseudodimension of $\mathcal{F}$ which can be used in practice.

Let $G = (V, E)$ be a graph and let $G' = (V', E')$ be the graph obtained by removing from V some nodes and from E the edges incident to any of the removed nodes. Specifically:

— If G is undirected, we obtain V' by removing all nodes of degree 1 from V.
— If G is directed, we obtain V' by removing all nodes u such that the elements of E involving u are either all in the form (u, v) or are all in the form (v, u).

Consider now the largest (in terms of number of nodes) *Weakly Connected Component* (WCC) of G', and let L be its size (number of nodes in it).

LEMMA 4.4. *We have:*

$$\mathsf{PD}(\mathcal{F}) \leq \lfloor \log_2 L \rfloor + 1 \ .$$

PROOF. Let's consider *undirected* graphs first. Each WCC of G' is a subset (potentially improper) of one and only one WCC of G. Let W be a WCC of G (W is a set of nodes, $W \subseteq V$) and let W' be the corresponding WCC of G' ($W' \subseteq V'$). Let (u, v) be a pair of nodes in W. It holds $N_{uw} \subseteq W$, i.e., $W \cap N_{uw} = N_{uw}$. We want to show that $N_{uw} \subseteq W'$.

Let v be any node in $W \setminus W'$ (if such a node exists, otherwise it must be $W' = W$ and therefore it must be $N_{uw} \subseteq W'$, since $N_{uw} \subseteq W$). It must be that $v \in V \setminus V'$, i.e., v is one of the removed nodes, which must have had degree 1 in G. The node v is not *internal* to any SP between any two nodes in G, i.e., $v \notin N_{zy}$ for any pair of nodes $(z, y) \in V \times V$, and particularly $v \notin N_{uw}$. This is true for any $v \in W \setminus W'$, hence $(W \setminus W') \cap N_{uw} = \emptyset$. We have:

$$\begin{aligned} W' \cap N_{uw} &= (W \cap N_{uw}) \setminus ((W \setminus W') \cap N_{uw}) \\ &= N_{uw} \setminus \emptyset = N_{uw}, \end{aligned}$$

i.e., $N_{uw} \subseteq W'$. Thus, $|N_{uw}| \leq |W'|$, and therefore $\mathsf{H}(G) \leq L$, from which we obtain the thesis, given Thm. 4.3.

Let's now consider *directed* graphs. It is no longer true that each WCC of G' is a subset (potentially improper) of one and only one WCC of G: there may be multiple WCCs of G' that are subsets of a WCC of G, hence we cannot proceed as in the case of undirected graphs.

Let $\{u, v, w, z\}$ be a set of nodes in V', such that at least three of them are distinct (if two of them are the same, we can assume w.l.o.g. that they are neither u and v nor w and z), and such that there is a path (and hence a SP) in G from u to v and from w to z, and that all these nodes belong to the same WCC of G but to two or more different WCCs of G'. We want to show that no set containing both $((u, v), x)$ and $((w, z), y)$ for some $x, y \in (0, 1)$ could have been shattered by $\mathcal{F}^+$.

Let $S = \{((u, v), x), ((w, z), y)\}$, for u, v, w, z as above. If $\mathcal{F}^+$ cannot shatter S than it cannot shatter any superset of S, so we can focus on S. We assumed that there is a SP from u to v and a SP from w to z in G. Any SP from u to v and from w to z still exists in G', as the removed nodes are not internal to any SPs in G. Hence u and v belong to the same WCC A in G' and w and z belong to the same WCC B in G'. We have, by construction of u, v, w, z that $A \neq B$.

Assume that S is shattered by $\mathcal{F}^+$. Then there must be a node h that is internal to both a SP from u to v and a SP from w to z. If there was not such a node h then S could not be shattered, as there would not be a node ℓ such that the intersection between S and the range R_ℓ associated to ℓ is S. Since h exists and it is internal to two SPs, then it must belong to V'. Since all SPs from u to v and from w to z still exist in G' then so do those that go through h. This means that there is a path from each of u, v, w, z to the others (e.g., from u to each of v, w, and z), hence they should all belong to the same WCC of G', but this is a contradiction. Hence S cannot be shattered by $\mathcal{F}^+$.

This implies that sets that can be shattered by $\mathcal{F}^+$ are only sets in the form $\{((u_i, v_i), x_i), i = k\}$ such that all nodes u_i and v_i (for all i) belong to the same WCC of G'. Hence, we can proceed as in the undirected graphs case and obtain the thesis. □

We comment that the upper bound derived in Lemma 4.4 is somewhat disappointing, and sometimes non-informative: if G is undirected and has a single connected component, then the same bound to the sample size that can be obtained using the pseudodimension (see Thm. 4.2 below) could be easily obtained using the union bound. We conjecture that it should be possible to obtain better bounds (see Conjecture 4.9.)

4.2.3. Special Cases. In this section we consider some special restricted settings that make computing an high-quality approximation of the BC of all nodes easier. One example of such restricted settings is when the graph is *undirected* and every pair of distinct nodes is either connected with a *single* SP or there is no path between the two nodes (because they belong to different connected components). Examples of these settings are many road networks, where the unique SP condition is often enforced [Geisberger et al. 2008]. Riondato and Kornaropoulos [2016, Lemma 2] showed that, in this case, the number of samples needed to compute a high-quality approximation of the BC of all nodes is *independent* of any property of the graph, and only depends on the quality controlling parameters ε and δ. The algorithm by Riondato and Kornaropoulos [2016] works differently from ABRA-s, as it samples one SP at a time and only updates the BC estimation of nodes along this path, rather than sampling a pair of nodes and updating the estimation of all nodes on any SPs between the sampled nodes. Nevertheless we can actually even generalize the result by Riondato and Kornaropoulos [2016], as shown in Thm. 4.5.

THEOREM 4.5. *Let $G = (V, E)$ be a graph such that it is possible to partition the set $\mathcal{D} = \{(u, v) \in V \times V, u \neq v\}$ in two classes: a class $A = \{(u^*, v^*)\}$ containing a single pair of different nodes (u^*, v^*) such that $\sigma_{u^*v^*} \leq 2$ (i.e., connected by either at most two SPs or not connected), and a class $B = \mathcal{D} \setminus A$ of pairs (u, v) of nodes with $\sigma_{uv} \leq 1$ (i.e., either connected by a single SP or not connected). Then the pseudodimension of the family of functions*

$$\{f_w \ : \ \mathcal{D} \to [0, 1], w \in V\},$$

where f_w is defined as in (10)*, is at most* 3.

Befor we present the proof of this theorem, let us complete the discussion, showing how this bound can be used to automent the stopping condition of ABRA-s.

COROLLARY 4.6. *Suppose to augment ABRA-s with the additional stopping condition instructing to return the set $\tilde{B} = \{\widetilde{\mathsf{b}}(w), w \in V\}$ after a total of*

$$r = \frac{c}{\varepsilon^2}\left(3 + \ln\frac{1}{\delta}\right)$$

pairs of nodes have been sampled from $\mathcal{D}$. The set $\tilde{B}$ is an (ε, δ)-approximation.

To prove Thm. 4.5 we show, in Lemma 4.7, that some subsets of $\mathcal{D} \times [0,1]$ can not be shattered by $\mathcal{F}^+$, on any graph G. Thm. 4.5 follows immediately from this result, and Corollary 4.6 then follows from Thms. 4.2 and 4.5.

LEMMA 4.7. *There exists no undirected graph $G = (V, E)$ such that it is possible to shatter a set*

$$B = \{((u_i, v_i), x_i), 1 \leq i \leq 4\} \subseteq \mathcal{D} \times [0,1]$$

if there are at least three distinct values $j', j'', j''' \in [1,4]$ for which

$$\sigma_{u_{j'}v_{j'}} = \sigma_{u_{j''}v_{j''}} = \sigma_{u_{j'''}v_{j'''}} = 1 \ .$$

PROOF. First of all, according to Lemmas B.1 and B.2, for B to be shattered it must be

$$(u_i, v_i) \neq (u_j, v_j) \text{ for } i \neq j$$

and $x_i \in (0,1]$, $1 \leq i \leq 4$.

Riondato and Kornaropoulos [2016, Lemma 2] showed that there exists no undirected graph $G = (V, E)$ such that it is possible to shatter B if

$$\sigma_{u_1v_1} = \sigma_{u_2v_2} = \sigma_{u_3v_3} = \sigma_{u_4v_4} = 1 \ .$$

Hence, what we need to show to prove the thesis is that it is impossible to build an undirected graph $G = (V, E)$ such that $\mathcal{F}^+$ can shatter B when the elements of B are such that

$$\sigma_{u_1v_1} = \sigma_{u_2v_2} = \sigma_{u_3v_3} = 1 \quad \text{and} \quad \sigma_{u_4v_4} = 2 \ .$$

Assume now that such a graph G exists and therefore B is shattered by $\mathcal{F}^+$.

For $1 \leq i \leq 3$, let p_i be the *unique* SP from u_i to v_i, and let p'_4 and p''_4 be the two SPs from u_4 to v_4.

First of all, notice that if any two of p_1, p_2, p_3 meet at a node a and separate at a node b, then they can not meet again at any node before a or after b, as otherwise there would be multiple SPs between their extreme nodes, contradicting the hypothesis. Let this fact be denoted as F_1.

Since B is shattered, its subset

$$A = \{((u_i, v_i), x_i), 1 \leq i \leq 3\} \subset B$$

is also shattered, and in particular it can be shattered by a collection of ranges that is a subset of a collection of ranges that shatters B. We now show some facts about the properties of this shattering which we will use later in the proof.

Define

$$i^+ = \begin{cases} i+1 & \text{if } i = 1,2 \\ 1 & \text{if } i = 3 \end{cases}$$

and

$$i^- = \begin{cases} 3 & \text{if } i = 1 \\ i-1 & \text{if } i = 2,3 \end{cases} \ .$$

Let w_A be a node such that $R_{w_A} \cap A = A$. For any set $L = \{k_1, k_2, \dots\} \subseteq \{1,2,3,4\}$ of indices, let $w_L = w_{k_1,k_2,\dots}$ be the node such that

$$R_L \cap A = \{((u_{k_\ell}, v_{k_\ell}), x_{k_\ell}), k_\ell \in L\} \ .$$

For example, for $i \in \{1,2,3\}$, w_{i,i^+} is the node such that

$$R_{w_{i,i^+}} \cap A = \{((u_i, v_i), x_i), ((u_{i^+}, v_{i^+}), x_{i^+})\} \ .$$

Analogously, w_{i,i^-} is the node such that

$$R_{w_{i,i^-}} \cap A = \{((u_i, v_i), x_i), ((u_{i^-}, v_{i^-}), x_{i^-})\} \ .$$

We want to show that w_A is on the SP connecting w_{i,i^+} to w_{i,i^-} (such a SP must exist because the graph is undirected and w_{i,i^+} and w_{i,i^-} must be on the same connected component, as otherwise they could not be use to shatter A.) Assume w_A was not on the SP connecting w_{i,i^+} to w_{i,i^-}. Then we would have that either w_{i,i^+} is "between" w_A and w_{i,i^-} ((i.e., along the SP connecting these nodes) or w_{i,i^-} is between w_A and w_{i,i^+}. Assume it was the former (the latter follows by symmetry). Then

(1) there must be a SP p' from u_{i^-} to v_{i^+} that goes through w_{i,i^-};
(2) there must be a SP p'' from u_{i^-} to v_{i^+} that goes through w_A;
(3) there is no SP from u_{i^-} to v_{i^+} that goes through w_{i,i^+}.

Since there is only one SP from u_{i^-} to v_{i^-}, it must be that $p' = p''$. But then p' is a SP that goes through $w_{i,i-}$ and through w_A but not through w_{i,i^+}, and p_i is a SP that goes through w_{i,i^-}, through w_{i,i^+}, and through w_A (either in this order or in the opposite). This means that there are at least two SPs between w_{i,i^-} and w_A, and therefore there would be two SPs between u_i and v_i, contradicting the hypothesis that there is only one SP between these nodes. Hence it must be that w_A is between w_{i,i^-} and w_{i,i^+}. This is true for all i, $1 \leq i \leq 3$. Denote this fact as F_2.

Consider now the nodes $w_{i,4}$ and $w_{j,4}$, for $i, j \in \{1, 2, 3\}$, $i \neq j$. We now show that they can not belong to the same SP from u_4 and v_4.

— Assume that $w_{i,4}$ and $w_{j,4}$ are on the same SP p from u_4 to v_4 and assume that $w_{i,j,4}$ is also on p. Consider the possible orderings of $w_{i,4}$, $w_{j,4}$ and $w_{i,j,4}$ along p.
  — If the ordering is $w_{i,4}$, then $w_{j,4}$, then $w_{i,j,4}$ or $w_{j,4}$, then $w_{j,4}$, then $w_{i,j,4}$, or the reverses of these orderings (for a total of four orderings), then it is easy to see that fact F_1 would be contradicted, as there are two different SPs from the first of these nodes to the last, one that goes through the middle one, and one that does not, but then there would be two SPs between the pair of nodes (u_k, v_k) where k is the index in $\{1, 2, 3\}$ different than 4 that is in common between the first and the last nodes in this ordering, and this would contradict the hypothesis, so these orderings are not possible.
  — Assume instead the ordering is such that $w_{i,j,4}$ is between $w_{i,4}$ and $w_{j,4}$ (two such ordering exist). Consider the paths p_i and p_j. They must meet at some node $w_{f_{i,j}}$ and separate at some node $w_{l_{i,j}}$. From the ordering, and fact F_1, $w_{i,j,4}$ must be between these two nodes. From fact F_2 we have that also w_A must be between these two nodes. Moreover, neither $w_{i,4}$ nor $w_{j,4}$ can be between these two nodes. But then consider the SP p. This path must go together with p_i (resp. p_j) from at least $p_{i,4}$ (resp. $p_{j,4}$) to the farthest between $w_{f_{i,j}}$ and $w_{l_{i,j}}$ from $p_{i,4}$ (resp. $p_{j,4}$). Then in particular p goes through all nodes between $w_{f_{i,j}}$ and $w_{l_{i,j}}$ that p_i and p_j go through. But since w_A is among these nodes,and w_A can not belong to p, this is impossible, so these orderings of the nodes $w_{i,4}$, $w_{j,4}$, and $w_{i,j,4}$ are not possible.

  Hence we showed that $w_{i,4}$, $w_{j,4}$, and $w_{i,j,4}$ can not be on the same SP from u_4 to v_4.

— Assume now that $w_{i,4}$ and $w_{j,4}$ are on the same SP from u_4 to v_4 but $w_{i,j,4}$ is on the other SP from u_4 to v_4 (by hypothesis there are only two SPs from u_4 to v_4). Since what we show in the previous point must be true for all choices of i and j, we have that all nodes $w_{h,4}$, $1 \leq h \leq 3$, must be on the same SP from u_4 to v_4, and all nodes in the form $w_{i,j,4}$, $1 \leq i < j \leq 3$ must be on the other SP from u_4 to v_4. Consider now these three nodes, $w_{1,2,4}$, $w_{1,3,4}$, and $w_{2,3,4}$ and consider their ordering along the SP from u_4 to v_4 that they lay on. No matter what the ordering is, there is an index $h \in \{1, 2, 3\}$ such that the SP p_h must go through the extreme two nodes in the ordering but not through the middle one. But this would contradict fact F_1, so it is impossible that we have $w_{i,4}$ and $w_{j,4}$ on the same SP from u_4 to v_4 but $w_{i,j,4}$ is on the other SP, for any choice of i and j.

We showed that the nodes $w_{i,4}$ and $w_{j,4}$ can not be on the same SP from u_4 to v_4. But this is true for any choice of the unordered pair (i, j) and there are three such choices, but only two SPs from u_4 to v_4, so it is impossible to accommodate all the constraints requiring $w_{i,4}$ and $w_{j,4}$ to be on different SPs from u_4 to v_4. Hence we reach a contradiction and B can not be shattered. □

The bound in Thm. 4.5 is tight, i.e., there exists a graph for which the pseudodimension is exactly 3 [Riondato and Kornaropoulos 2016, Lemma 4]. Moreover, as soon as we relax the requirement in Thm. 4.5 and allow two pairs of nodes to be connected by two SPs, there are graphs with pseudodimension 4, as shown in the following Lemma.

LEMMA 4.8. *There is an undirected graph $G = (V, E)$ such that there is a set $\{(u_i, v_i), u_i, v_i \in V, u_i \neq v_i, 1 \leq i \leq 4\}$ with $|\mathsf{S}_{u_1,v_1}| = |\mathsf{S}_{u_2,v_2}| = 2$ and $|\mathsf{S}_{u_3,v_3}| = |\mathsf{S}_{u_4,v_4}| = 1$ that is shattered.*

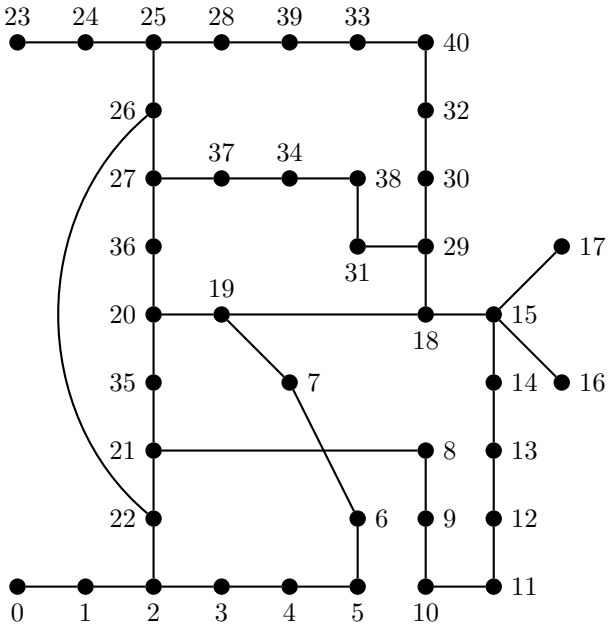


Fig. 2: Graph for Lemma 4.8

PROOF. Consider the undirected graph $G = (V, E)$ in Fig. 2. There is a single SP from 0 to 16:

$$0, 1, 2, 22, 21, 35, 20, 19, 18, 15, 16 \ .$$

There is a single SP from 23 to 17:

$$23, 24, 25, 26, 27, 36, 20, 19, 18, 15, 17 \ .$$

There are exactly two SPs from 5 to 33:

$$5, 4, 3, 2, 22, 26, 25, 28, 39, 33 \text{ and}$$
$$5, 6, 7, 19, 18, 29, 30, 32, 40, 33 \ .$$

There are exactly two SPs from 11 to 34:

$$11, 10, 9, 8, 21, 22, 26, 27, 37, 34 \text{ and}$$
$$11, 12, 13, 14, 15, 18, 29, 31, 38, 34 \ .$$

Let $a = ((0, 16), 1)$, $b = ((23, 17), 1)$, $c = ((5, 33), 1/2)$, and $d = ((11, 34), 1/2)$. We can shatter the set $Q = \{a, b, c, d\}$, as shown in Table II. □

Table II: How to shatter $Q = \{a, b, c, d\}$ from Lemma 4.8.

$P \subseteq Q$	Node v such that $P = Q \cap R_v$
$\emptyset$	0
$\{a\}$	1
$\{b\}$	24
$\{c\}$	40
$\{d\}$	38
$\{a, b\}$	20
$\{a, c\}$	2
$\{a, d\}$	21
$\{b, c\}$	25
$\{b, d\}$	27
$\{c, d\}$	29
$\{a, b, c\}$	19
$\{a, b, d\}$	15
$\{a, c, d\}$	22
$\{b, c, d\}$	26
$\{a, b, c, d\}$	18

For the case of *directed* networks, it is currently an open question whether a high-quality (i.e., within ε) approximation of the BC of all nodes can be computed from a sample whose size is independent of properties of the graph, but it is known that, even if possible, the constant would not be the same as for the undirected case [Riondato and Kornaropoulos 2016, Sect. 4.1].

We conjecture that, given some information on how many pair of nodes are connected by x shortest paths, for $x \geq 0$, it should be possible to derive a strict bound on the pseudodimension associated to the graph. Formally, we pose the following conjecture, which would allow us to generalize Lemma 4.7, and develop an additional stopping rule for ABRA-s based on the empirical pseudodimension.

CONJECTURE 4.9. *Let $G = (V, E)$ be a graph and let ℓ be the maximum positive integer for which there exists a set $L = \{(u_1, v_i), \ldots, (u_\ell, v_\ell)\}$ of ℓ distinct pairs of distinct vertices such that*

$$\sum_{i=1}^{\ell} \sigma_{u_i v_i} \geq \binom{\ell}{\lfloor \ell/2 \rfloor} \ .$$

then $\mathsf{PD}(\mathcal{F}) \leq \ell$.

The conjecture is tight in the sense that, e.g., for the graph in Fig. 2, we have that $\ell = 4$ and the pseudodimension is exactly ℓ.

4.3. Improved Estimators

Geisberger et al. [2008] present an improved estimator for BC using random sampling. Their experimental results show that the quality of the approximation is significantly improved, but they do not present any theoretical analysis. Their algorithm, which follows the work

of Brandes and Pich [2007] differs from ours as it samples nodes and performs a Single-Source-Shortest-Paths (SSSP) computation from each of the sampled nodes. We can use an adaptation of their estimator in a variant of our algorithm, and we can prove that this variant is still probabilistically guaranteed to compute an (ε, δ)-approximation of the BC of all nodes, therefore removing the main limitation of the original work, which offered no quality guarantees. We now present this variant considering, for ease of discussion, the special case of the linear scaling estimator by Geisberger et al. [2008]. This technique can be extended to the generic parameterized estimators they present. We also limit our discussion to the case of using Thm. 3.4 to compute a bound to the supremum of the deviations, but the discussion can be extended to the case for Thm. 3.3.

The intuition behind the improved estimator is to increase the estimation of the BC for a node w proportionally to the ratio between the SP distance $\mathsf{d}(u, w)$ from the first component u of the pair (u, v) to w and the SP distance $\mathsf{d}(u, v)$ from u to v. Rather than sampling pairs of nodes, the algorithm samples triples (u, v, d), where d is a *direction*, (either $\leftarrow$ or $\rightarrow$), and updates the betweenness estimation differently depending on d, as follows. Let $\mathcal{D}' = \mathcal{D} \times \{\leftarrow, \rightarrow\}$ and for each $w \in V$, define the function g_w from $\mathcal{D}'$ to $[0, 1]$ as:

$$g_w(u, v, d) = \begin{cases} \frac{\sigma_{uv}(w)}{\sigma_{uv}} \frac{\mathsf{d}(u,w)}{\mathsf{d}(u,v)} & \text{if } d = \rightarrow \\ \frac{\sigma_{uv}(w)}{\sigma_{uv}} \left(1 - \frac{\mathsf{d}(u,w)}{\mathsf{d}(u,v)}\right) & \text{if } d = \leftarrow \end{cases}$$

Let $\mathcal{S}$ be a collection of ℓ elements of $\mathcal{D}'$ sampled uniformly and independently at random with replacement. Our estimation $\widetilde{\mathsf{b}}(w)$ of the BC of a node w is

$$\widetilde{\mathsf{b}}(w) = \frac{2}{\ell} \sum_{(u,v,d) \in \mathcal{S}} g_w(u, v, d) = 2\mathsf{m}_{\mathcal{S}}(f_w) \ .$$

The presence of the factor 2 in the estimator calls for some minor adjustments in the definition of $\xi_{\mathcal{S}_i}$ which, for this variant of ABRA-s, becomes

$$\xi_{\mathcal{S}_i} = 2\omega_i^* + 2\frac{2\gamma_i + \sqrt{b\gamma_i(2\gamma_i + 2\ell\omega_i^*)}}{\ell} + 2\sqrt{\frac{\gamma_i}{2\ell}},$$

This change ensures that the output of this variant of ABRA-s is still a high-quality approximation of the BC of all nodes, i.e., that Thm. 4.1 still holds with this new definition of $\xi_{\mathcal{S}_i}$. This is due to the fact that the results on the Rademacher averages presented in Sect. 3.2 can be extended to families of functions whose co-domain is an interval $[0, b]$, as discussed in Appendix A.

4.4. Relative-error Top-k Approximation

In practical applications it is usually necessary (and sufficient) to identify the nodes with highest BC, as they act, in some sense, as the "primary information gateways" of the network. In this section we present a variant ABRA-k of ABRA-s to compute a high-quality approximation of the set $\mathsf{TOP}(k, G)$ of the top-k nodes with highest BC in a graph G. The approximation $\widetilde{\mathsf{b}}(w)$ returned by ABRA-k for a node w is within a *multiplicative* factor ε from its exact value $\mathsf{b}(w)$, rather than an additive factor ε as in ABRA-s. This higher accuracy has a cost in terms of the number of samples needed to compute the approximations.

Formally, assume to order the nodes in the graph in decreasing order by BC, ties broken arbitrarily, and let b_k be the BC of the k-th node in this ordering. Then the set $\mathsf{TOP}(k, G)$ is defined as the set of nodes with BC at least b_k, and can contain more than k nodes:

$$\mathsf{TOP}(k, G) = \{(w, \mathsf{b}(w)) \ : \ v \in V \text{ and } \mathsf{b}(w) \geq b_k\} \ .$$

The algorithm ABRA-k follows the same approach as the algorithm for the same task by Riondato and Kornaropoulos [2016, Sect. 5.2] and works in two phases. Let δ_1 and δ_2 be

such that $(1-\delta_1)(1-\delta_2) \geq (1-\delta)$. In the first phase, we run ABRA-s with parameters ε and δ_1. Let ℓ' be the k-th highest value $\widetilde{\mathsf{b}}(w)$ returned by ABRA-s, ties broken arbitrarily, and let $\tilde{b}' = \ell' - \varepsilon$.

In the second phase, we use a variant ABRA-r of ABRA-s with a modified stopping condition based on relative-error versions of Thms. 3.3 and 3.4 (Thms. C.1 and C.2 from Appendix C), which take ε, δ_2, and $\theta = \tilde{b}'$ as parameters. The parameter θ plays a role in the stopping condition. Indeed, ABRA-r is the same as ABRA-s, with the only crucial difference in the definition of the quantity $\xi\mathcal{S}_i$, which, when using the explicit bound from Thm. C.2, now becomes:

$$\xi_{\mathcal{S}_i} = 2\omega_i^* + 2\frac{\theta^{-1}\ln(3/\eta) + \sqrt{(\theta^{-1}\ln(3/\eta) + 2\ell\omega_i^*)\theta^{-1}\ln(3/\eta)}}{\ell} + \theta^{-1}\sqrt{\frac{\ln(3/\eta)}{2\ell}} \ . \quad (14)$$

One can analogously redefine $\xi_{\mathcal{S}_i}$ using the implicit bound from Thm. C.1.

The pseudocode for ABRA-k is presented in Alg. 2.

ALGORITHM 2: ABRA-k: relative-error approximation of top-k BC nodes on static graph

input : Graph $G = (V, E)$, accuracy parameter $\varepsilon \in (0,1)$, confidence parameter $\delta \in (0,1)$, value $k \geq 1$

output: Set $\widetilde{B}$ of approximations of the BC of the top-k nodes in V with highest BC

1 $\delta', \delta'' \leftarrow$ reals such that $(1-\delta_1)(1-\delta_2) \geq 1-\delta$

2 $\widetilde{B}' \leftarrow$ output of ABRA-s run with input G, ε, δ'

3 $\ell' \leftarrow k$-th highest value in $\widetilde{B}'$

4 $\tilde{b}' = \ell' - \varepsilon$

5 $\widetilde{B} \leftarrow$ output of a variant of ABRA-s using the definition of $\xi_{\mathcal{S}_i}$ from (14), and input $G, \varepsilon, \delta'', \tilde{b}'$

6 **return** $\widetilde{B}$

THEOREM 4.10. *Let $\widetilde{B} = \{\widetilde{\mathsf{b}}(w), w \in V\}$ be the output of ABRA-r. Then $\widetilde{B}$ is such that*

$$\Pr\left(\exists w \in V \ : \ \frac{|\widetilde{\mathsf{b}}(v) - \mathsf{b}(v)|}{\max\{\theta, \mathsf{b}(v)\}} > \varepsilon\right) < \delta \ .$$

The proof follows the same steps as the proof for Thm. 4.1, using the above definition of $\xi_{\mathcal{S}_i}$ and applying Thms. 3.3 and 3.4 from Appendix C instead of Thms. 3.3 and 3.4.

Let ℓ'' be the k^{th} highest value $\widetilde{\mathsf{b}}(w)$ returned by ABRA-r (ties broken arbitrarily) and let $\tilde{b}'' = \ell''/(1+\varepsilon)$. ABRA-k then returns the set

$$\widetilde{\mathsf{TOP}}(k, G) = \{(w, \widetilde{\mathsf{b}}(w)) \ : \ w \in V \text{ and } \widetilde{\mathsf{b}}(w) \geq \tilde{b}''\} \ .$$

We have the following result showing the properties of the collection $\widetilde{\mathsf{TOP}}(k, G)$.

THEOREM 4.11. *With probability at least $1-\delta$, the set $\widetilde{\mathsf{TOP}}(k, G)$ is such that:*

(1) for any pair $(v, \mathsf{b}(v)) \in \mathsf{TOP}(k, G)$, there is one pair $(v, \widetilde{\mathsf{b}}(v)) \in \widetilde{\mathsf{TOP}}(k, G)$ (i.e., we return a superset of the top-k nodes with highest betweenness) and this pair is such that $|\widetilde{\mathsf{b}}(w) - \mathsf{b}(w)| \leq \varepsilon\mathsf{b}(w)$;

(2) for any pair $(w, \widetilde{\mathsf{b}}(w)) \in \widetilde{\mathsf{TOP}}(k, G)$ such that $(w, \mathsf{b}(w)) \notin \mathsf{TOP}(k, G)$ (i.e., any false positive) we have that $\widetilde{\mathsf{b}}(w) \leq (1+\varepsilon)b_k$ (i.e., the false positives, if any, are among the nodes returned by ABRA-k with lower BC estimation).

PROOF. With probability at least $1-\delta'$, the set $\tilde{B}'$ computed during the first phase (execution of ABRA-s) has the properties from Thm. 4.1. With probability at least $1-\delta''$, the set $\tilde{B}''$ computed during the second phase (execution of ABRA-s) has the properties from Thm. 4.10. Suppose both these events occur, which happens with probability at least $1-\delta$. Consider the value ℓ'. It is straightforward to check that ℓ' is a lower bound on b_k: indeed there must be at least k nodes with exact BC at least ℓ'. For the same reasons, and considering the fact that we run ABRA-r with parameters ε, δ'', and $\theta = \ell'$, we have that $\ell'' \leq b_k$. From this and the definition of $\widetilde{\mathsf{TOP}}(k,G)$, it follows that the elements of $\widetilde{\mathsf{TOP}}(k,G)$ are such that their exact BC may be greater than ℓ'', and therefore of b_k. This means that $\mathsf{TOP}(k,G) \subseteq \widetilde{TOP}(k,G)$. The other properties of $\widetilde{TOP}(k,G)$ follow from the properties of the output of ABRA-r. □

5. DYNAMIC GRAPH BC APPROXIMATION

In this section we present an algorithm, named ABRA-d, that computes and keeps up to date an high-quality approximation of the BC of all nodes in a *fully dynamic graph*, i.e., in a graph where nodes and edges can be added or removed over time. Our algorithm builds on the recent work by Hayashi et al. [2015], who introduced two fast data structures called the Hypergraph Sketch and the Two-Ball Index: the Hypergraph Sketch stores the BC estimations for all nodes, while the Two-Ball Index is used to store the SP DAGs and to understand which parts of the Hypergraph Sketch needs to be modified after an update to the graph (i.e., an edge or node insertion or deletion). Hayashi et al. [2015] show how to populate and update these data structures to maintain an (ε,δ)-approximation of the BC of all nodes in a fully dynamic graph. Using the novel data structures results in orders-of-magnitude speedups w.r.t. previous contributions [Bergamini and Meyerhenke 2015, 2016]. The algorithm by Hayashi et al. [2015] is based on a static random sampling approach which is identical to the one described for ABRA-s, i.e., pairs of nodes are sampled and the BC estimation of the nodes along the SPs between the two nodes are updated as necessary. Their analysis on the number of samples necessary to obtain an (ε,δ)-approximation of the BC of all nodes uses the union bound, resulting in a number of samples that depends on the logarithm of the number of nodes in the graph, i.e., $O(\varepsilon^{-}2(\log(|V|/\delta)))$ pairs of nodes must be sampled.

ABRA-d builds and improves over the algorithm presented by Hayashi et al. [2015] as follows. Instead of using a static random sampling approach with a fixed sample size, we use the progressive sampling approach and the stopping condition that we use in ABRA-s to understand when we sampled enough to first populate the Hypegraph Sketch and the Two-Ball Index. Then, after each update to the graph, we perform the same operations as in the algorithm by Hayashi et al. [2015], with the crucial addition, after these operation have been performed, of keeping the set $\mathcal{V}_{\mathcal{S}}$ of vectors and the map M (already used in ABRA-s) up to date, and checking whether the stopping condition is still satisfied. If it is not, additional pairs of nodes are sampled and the Hypergraph Sketch and the Two-Ball Index are updated with the estimations resulting from these additional samples. The sampling of additional pairs continues until the stopping condition is satisfied, potentially according to a sample schedule either automatic, or specified by the user. As we show in Sect. 6, the overhead of additional checks of the stopping condition is minimal. On the other hand, the use of the progressive sampling scheme based on the Rademacher averages allows us to sample much fewer pairs of nodes than in the static sampling case based on the union bound: Riondato and Kornaropoulos [2016] already showed that it is possible to sample much less than $O(\log|V|)$ nodes, and, as we show in our experiments, our sample sizes are even smaller than the ones by Riondato and Kornaropoulos [2016]. The saving in the number of samples results in a huge speedup, as the running time of the algorithms are, in a first approximation, linear in the number of samples, and in a reduction in the amount

of space required to store the data structures, as they now store information about fewer SP DAGs.

THEOREM 5.1. *The set* $\widetilde{B} = \{\widetilde{\mathsf{b}}(w), w \in V\}$ *returned by* ***ABRA-d*** *after each update has been processed is such that*

$$\Pr(\exists w \in V \text{ s.t. } |\widetilde{\mathsf{b}}(w) - \mathsf{b}(w)| > \varepsilon) < \delta \ .$$

The proof follows from the correctness of the algorithm by Hayashi et al. [2015] and of ABRA-s (Thm. 4.1).

6. EXPERIMENTAL EVALUATION

In this section we presents the results of our experimental evaluation.[5] We measure and analyze the performances of ABRA-s in terms of its runtime and sample size and accuracy, and compared them with those of the exact algorithm BA [Brandes 2001] and the approximation algorithm RK [Riondato and Kornaropoulos 2016], which offers the same guarantees as ABRA-s (computes an (ε, δ)-approximation the BC of all nodes).

Implementation and Environment. We implement ABRA-s and ABRA-d in C++, as an extension of the NetworKit library [Staudt et al. 2016]. We use NLopt [Johnson 2014] for the optimization steps . The code is available from http://matteo.rionda.to/software/ABRA-radebetw.tbz2 We performed the experiments on a machine with a AMD Phenom$^{\text{TM}}$ II X4 955 processor and 16GB of RAM, running FreeBSD 11.

Datasets and Parameters. We use graphs of various nature (communication, citations, P2P, and social networks) from the SNAP repository [Leskovec and Krevl 2014]. The characteristics of the graphs are reported in the leftmost column of Table III.

In our experiments we varied ε in the range $[0.005, 0.3]$, and we also evaluate a number of different sampling schedules (see Sect. 6.2). In all the results we report, δ is fixed to 0.1. We experimented with different values for this parameter, and, as expected, it has a very limited impact on the nature of the results, given the logarithmic dependence of the sample size on δ. We performed five runs for each combination of parameters. The variance between the different runs was essentially insignificant, so we report, unless otherwise specified, the results for a random run.

6.1. Runtime and Speedup

Our main goal was to develop an algorithm that can compute an (ε, δ)-approximation of the BC of all nodes as fast as possible. Hence we evaluate the runtime and the speedup of ABRA-s w.r.t. BA and RK. The results are reported in columns 3 to 5 (from the left) of Table III. The values for $\varepsilon = 0.005$ are missing for Email-Enron and Cit-HepPh because in these cases both RK and ABRA-s were slower than BA, a phenomena that, limited to RK, can be seen also in the first line of the results for soc-Epinions: in this case, RK is slower than BA, but ABRA-s is faster.

As expected, the runtime is a perfect linear function of the sample size (column 9), which in turns grows as ε^{-2}. The speedup w.r.t. the exact algorithm BA is significant and naturally decreases quadratically with ε. More interestingly ABRA-s is always faster than RK, sometimes by a significant factor. At first, one may think that this is due to the reduction in the sample size (column 10), but a deeper analysis shows that this is only one component of the speedup, which is almost always greater than the reduction in sample size. The other component can be explained by the fact that RK must perform

[5]The results presented here used a previous version of ABRA-s. We are currently in the process of updating our implementation and running the experiments again. We expect the same results from a qualitative point of view.

an expensive computation (computing the vertex diameter [Riondato and Kornaropoulos 2016] of the graph) to determine the sample size before it can start sampling, while ABRA-s can immediately start sampling and rely on the stopping condition, whose computation is inexpensive, as we will discuss. The different speedups for different graphs are due to different characteristics of the graphs: when the SP DAG between two nodes has many paths, ABRA-s does more work per sample than RK, which only backtracks along a single SP of the DAG, hence the speedup is smaller.

Table III: Runtime, speedup, breakdown of runtime, sample size, reduction, and absolute error

Graph	ε	Runtime (sec.)	Speedup w.r.t. BA	Speedup w.r.t. RK	Runtime Breakdown (%) Sampling	Stop Cond.	Other	Sample Size	Reduction w.r.t. RK	Absolute Error ($\times 10^5$) max	avg	stddev
Soc-Epinions1 Directed $\lvert V\rvert = 75,879$ $\lvert E\rvert = 508,837$	0.005	483.06	1.36	2.90	99.983	0.014	0.002	110,705	2.64	70.84	0.35	1.14
	0.010	124.60	5.28	3.31	99.956	0.035	0.009	28,601	2.55	129.60	0.69	2.22
	0.015	57.16	11.50	4.04	99.927	0.054	0.018	13,114	2.47	198.90	0.97	3.17
	0.020	32.90	19.98	5.07	99.895	0.074	0.031	7,614	2.40	303.86	1.22	4.31
	0.025	21.88	30.05	6.27	99.862	0.092	0.046	5,034	2.32	223.63	1.41	5.24
	0.030	16.05	40.95	7.52	99.827	0.111	0.062	3,668	2.21	382.24	1.58	6.37
P2p-Gnutella31 Directed $\lvert V\rvert = 62,586$ $\lvert E\rvert = 147,892$	0.005	100.06	1.78	4.27	99.949	0.041	0.010	81,507	4.07	38.43	0.58	1.60
	0.010	26.05	6.85	4.13	99.861	0.103	0.036	21,315	3.90	65.76	1.15	3.13
	0.015	11.91	14.98	4.03	99.772	0.154	0.074	9,975	3.70	109.10	1.63	4.51
	0.020	7.11	25.09	3.87	99.688	0.191	0.121	5,840	3.55	130.33	2.15	6.12
	0.025	4.84	36.85	3.62	99.607	0.220	0.174	3,905	3.40	171.93	2.52	7.43
	0.030	3.41	52.38	3.66	99.495	0.262	0.243	2,810	3.28	236.36	2.86	8.70
Email-Enron Undirected $\lvert V\rvert = 36,682$ $\lvert E\rvert = 183,831$	0.010	202.43	1.18	1.10	99.984	0.013	0.003	66,882	1.09	145.51	0.48	2.46
	0.015	91.36	2.63	1.09	99.970	0.024	0.006	30,236	1.07	253.06	0.71	3.62
	0.020	53.50	4.48	1.05	99.955	0.035	0.010	17,676	1.03	290.30	0.93	4.83
	0.025	31.99	7.50	1.11	99.932	0.052	0.016	10,589	1.10	548.22	1.21	6.48
	0.030	24.06	9.97	1.03	99.918	0.061	0.021	7,923	1.02	477.32	1.38	7.34
Cit-HepPh Undirected $\lvert V\rvert = 34,546$ $\lvert E\rvert = 421,578$	0.010	215.98	2.36	2.21	99.966	0.030	0.004	32,469	2.25	129.08	1.72	3.40
	0.015	98.27	5.19	2.16	99.938	0.054	0.008	14,747	2.20	226.18	2.49	5.00
	0.020	58.38	8.74	2.05	99.914	0.073	0.013	8,760	2.08	246.14	3.17	6.39
	0.025	37.79	13.50	2.02	99.891	0.091	0.018	5,672	2.06	289.21	3.89	7.97
	0.030	27.13	18.80	1.95	99.869	0.108	0.023	4,076	1.99	359.45	4.45	9.53

Runtime breakdown. The main challenge in designing a stopping condition for progressive sampling algorithm is striking the right balance between the strictness of the condition (i.e., it should stop early) and the efficiency in evaluating it. We now comment on the efficiency, and will report about the strictness in Sect. 6.2 and 6.3. In columns 6 to 8 of Table III we report the breakdown of the runtime into the main components. It is evident that evaluating the stopping condition amounts to an insignificant fraction of the runtime, and most of the time is spent in computing the samples (selection of nodes, execution of SP algorithm, update of the BC estimations). The amount in the "Other" column corresponds to time spent in logging and checking invariants. We can then say that our stopping condition is extremely efficient to evaluate, and ABRA-s is almost always doing "real" work to improve the estimation.

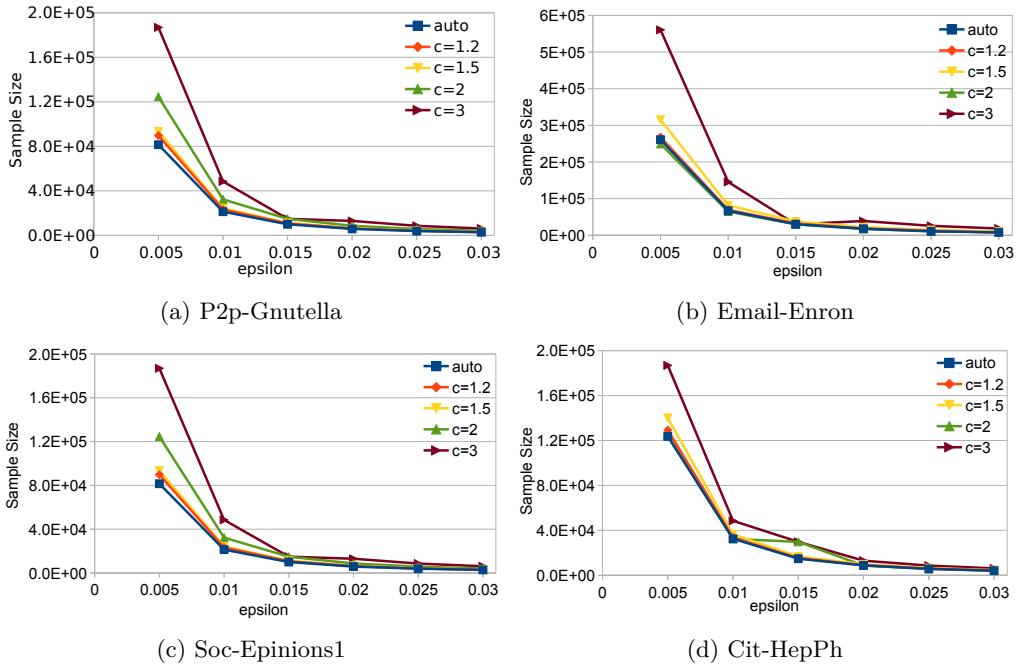
6.2. Sample Size and Sample Schedule

We evaluate the final sample size of ABRA-s and the performances of the "automatic" sample schedule (Sect. 4.1.1). The results are reported in columns 9 and 10 of Table III. As expected, the sample size grows with ε^{-2}. We already commented on the fact that ABRA-s uses a sample size that is consistently (up to $4\times$) smaller than the one used by RK and how this is part of the reason why ABRA-s is much faster than RK. In Fig. 3 we show the behavior of the final sample size chosen by the automatic sample schedule in comparison with *static geometric sample schedules*, i.e., schedules for which the sample size at iteration $i+1$ is c times the size of the sample size at iteration i. We can see that the *automatic sample schedule is always better than the geometric ones*, sometimes significantly depending on the value of c (e.g., more than $2\times$ decrease w.r.t. using $c = 3$ for $\varepsilon = 0.05$). Effectively this means that the automatic sample schedule really frees the user from having to selecting a parameter whose impact on the performances of the algorithm may be devastating (larger final sample size implies higher runtime). Moreover, thanks to the automatic sample schedule, ABRA-s always terminates after just two iterations, while this was not the case for the geometric sample schedules (taking even 5 iterations in some cases): this means that effectively the automatic sample schedules "jumps" directly to a sample size for which the stopping condition will be verified. We can sum up the results and say that the stopping condition of ABRA-s stops at small sample sizes, smaller than those used in RK, and the automatic sample schedule we designed is extremely efficient at choosing the right successive sample size, to the point that ABRA-s only needs two iterations.

6.3. Accuracy

We evaluate the accuracy of ABRA-s by measuring the absolute error $|\widetilde{\mathsf{b}}(v) - \mathsf{b}(v)|$. The theoretical analysis guarantees that this quantity should be at most ε for all nodes, with probability at least $1-\delta$. A first important result is that in *all* the thousands of runs of ABRA-s, the maximum error was *always* smaller than ε (not just with probability $> 1-\delta$). We report statistics about the absolute error in the three rightmost columns of Table III and in Fig. 4. The minimum error (not reported) was always 0, so we do not report it in the table. The maximum error is *an order of magnitude smaller than* ε, and the average error is around *three orders of magnitude* smaller than ε, with a very small standard deviation. As expected, the error grows as ε^{-2}. In Fig. 4 we show the behavior of the maximum, average, and average plus three standard deviations (approximately corresponding to the 95% percentile), to appreciate how most of the errors are almost two orders of magnitude smaller than ε.

All these results show that *ABRA-s is very accurate, more than what is guaranteed by the theoretical analysis*. This can be explained by the fact that the bounds to the sampling size, the stopping condition, and the sample schedule are *conservative*, in the sense that ABRA-s



(a) P2p-Gnutella

(b) Email-Enron

(c) Soc-Epinions1

(d) Cit-HepPh

Fig. 3: Final sample size for different sample schedules

may be sampling more than necessary to obtain an (ε, δ)-approximation. Tightening any of these components would result in a less conservative algorithm that offers the same approximation quality guarantees, and is an interesting research direction.

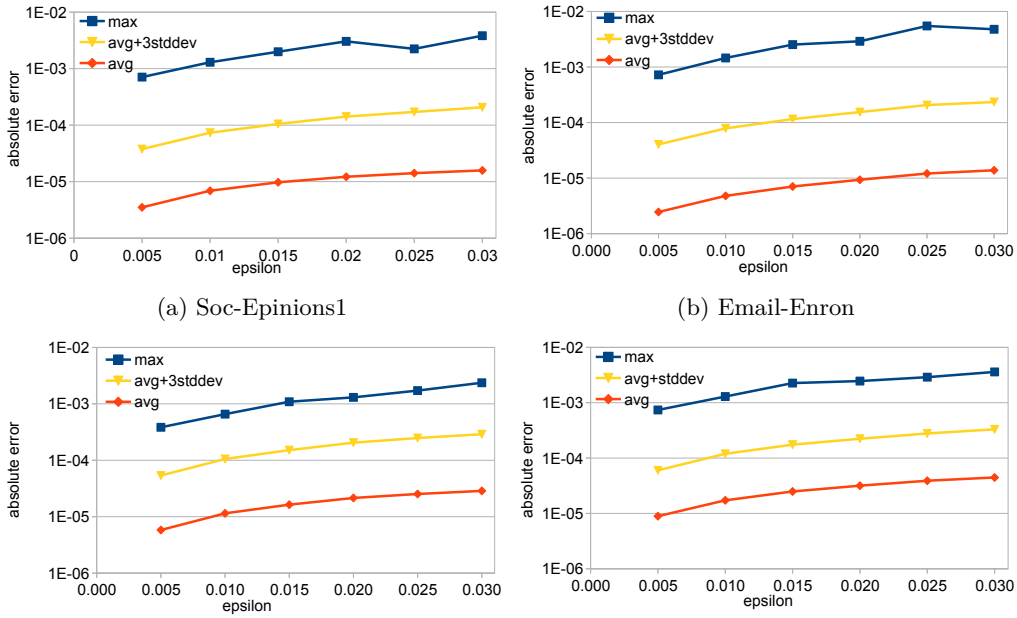
6.4. Dynamic BC Approximation

We did not evaluate ABRA-d experimentally, but, given its design, it is reasonable to expect that, when compared to previous contributions offering the same quality guarantees [Bergamini and Meyerhenke 2016; Hayashi et al. 2015], it would exhibit similar or even larger speedups and reductions in the sample size than what ABRA-s had w.r.t. RK. Indeed, the algorithm by Bergamini and Meyerhenke [2015] uses RK as a building block and it needs to constantly keep track of (an upper bound on) the vertex diameter of the graph, a very expensive operation. On the other hand, the analysis of the sample size by Hayashi et al. [2015] uses very loose simultaneous deviation bounds (the union bound). As already shown by Riondato and Kornaropoulos [2016], the resulting sample size is extremely large and they already showed how RK can use a smaller sample size. Since we built over the work by Hayashi et al. [2015] and ABRA-s improves over RK, we can reasonably expect it to have better performances than the algorithm by Hayashi et al. [2015]

6.5. Scalability

7. CONCLUSIONS

We presented ABRA, a family of sampling-based algorithms for computing and maintaining high-quality approximations of (variants of) the BC of all nodes in static and dynamic graphs with updates (both deletions and insertions). We discussed a number of variants of our basic algorithms, including finding the top-k nodes with higher BC, using improved estimators, and special cases when there is a single SP. ABRA greatly improves, theoretically and



(a) Soc-Epinions1

(b) Email-Enron

(c) P2p-Gnutella

(d) Cit-HepPh

Fig. 4: Absolute error evaluation. The vertical axis has a logarithmic scale.

experimentally, the current state of the art. The analysis relies on Rademacher averages and on pseudodimension, fundamental concepts from statistical learning theory. To our knowledge this is the first application of these results and ideas to graph mining, and we believe that they should be part of the toolkit of any algorithm designer interested in efficient algorithms for data analysis.

APPENDIX

A. IMPROVED BOUNDS

In this section we present the proofs for Thms. 3.3 and 3.4, which simultaneously extend and fine-tune [Oneto et al. 2013, Thms. 3.10 and 3.11].

In the rest of this section, we let $\mathcal{F}$ be a family of functions from some domain $\mathcal{D}$ to $[0, b]$. The non-negativity of the members of $\mathcal{F}$ is crucial in many of the results we now present. Let $\mathcal{S} = \{s_1, \ldots, s_\ell\}$ be a collection of ℓ elements from $\mathcal{D}$, sampled independently.

We start with a few facts about the conditional Rademacher average $\mathsf{R}_{\mathcal{F}}(\mathcal{S})$.

Definition A.1 (*[Boucheron et al. 2000]*). A *non-negative* function f from a domain $\mathcal{X}^\ell$ to $\mathbb{R}$ is *c-self-bounding* if there exist functions g_i, $1 \leq fi \leq \ell$, from $\mathcal{X}^{\ell-1}$ to $\mathbb{R}$ such that, for all $(x_1, \ldots, x_n) \in \mathcal{X}^\ell$, *both* following conditions hold:

(1) $0 \leq f(x_1, \ldots, x_\ell) - g_i(x_1, x_{i-1}, x_{i+1}, \ldots, x_\ell) \leq c$ for all i, $1 \leq i \leq \ell$; and

(2) $\sum_{i=1}^{\ell} (f(x_1, \ldots, x_\ell) - g_i(x_1, x_{i-1}, x_{i+1}, \ldots, x_\ell) \leq f(x_1, \ldots, x_\ell)$.

The following specializes and finely tunes part of the proof of [Oneto et al. 2013, Lemma 3.6] to functions with co-domain $[0, b]$.

LEMMA A.2. *The conditional Rademacher average*

$$\mathsf{R}_{\mathcal{F}}(\mathcal{S}) = \mathbb{E}_{\lambda}\left[\sup_{f\in\mathcal{F}} \frac{1}{\ell}\sum_{j=1}^{\ell}\lambda_i f(s_i)\right]$$

is a c-self-bounding function for $c = \frac{b}{2\ell}$.

PROOF. The proof follows the same steps as the proof for [Oneto et al. 2013, Lemma 3.6], with the additional observation that, for $\mathcal{F}$ satisfying the hypothesis, we have, for any i, $1 \le i \le \ell$,

$$\mathbb{E}_{\lambda}\left[\sup_{f\in\mathcal{F}} \frac{1}{\ell}\lambda_i f(s_i)\right] = \sup_{f\in\mathcal{F}} \frac{1}{\ell}\cdot\frac{1}{2}\cdot -1\cdot f(s_i) + \sup_{f\in\mathcal{F}} \frac{1}{\ell}\cdot\frac{1}{2}\cdot 1\cdot f(s_i) \le 0 + \sup_{f\in\mathcal{F}} \frac{1}{\ell}\cdot\frac{1}{2}\cdot 1\cdot f(s_i) \le \frac{b}{2\ell},$$

where the first inequality comes from the fact that $f(s_i) \ge 0$, hence $\sup_{f\in\mathcal{F}} \frac{1}{\ell}\cdot\frac{1}{2}\cdot -1\cdot f(s_i) \le 0$. □

THEOREM A.3 (THM. 2.1 [BOUCHERON ET AL. 2000]). *Let Z be a c-self-bounding function. Define, for $x \in \mathbb{R}$,*

$$\phi(x) = (1+x)\ln(1+x) - x \ .$$

Then for any $0 < t \le \mathbb{E}[Z]$,

$$\Pr\left(\mathbb{E}[Z] - Z \ge t\right) \le \exp\left[-\frac{\mathbb{E}[Z]}{c}\phi\left(-\frac{t}{\mathbb{E}[Z]}\right)\right] \le \exp\left[-\frac{t^2}{2c\mathbb{E}[Z]}\right] \ .$$

The following fact is immediate from the definition of conditional Rademacher average.

FACT A.4. *Define $\mathcal{F}^-$ as the family of functions containing a function $f^- = -f$ for every function $f \in \mathcal{F}$ (i.e., f^- is such that $f^-(x) = -f(x)$, for every $x \in \mathcal{D}$.) Then*

$$\mathsf{R}_{\mathcal{F}^-}(\mathcal{S}) = \mathsf{R}_{\mathcal{F}}(\mathcal{S}) \ .$$

We now present some results about the supremum of the deviations $\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right)$, and then prove Thms. 3.3 and 3.4.

LEMMA A.5 ([KOLTCHINSKII AND PANCHENKO 2002]). *We have*

$$\mathbb{E}\left[\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right)\right] \le 2\mathbb{E}\left[\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right] \ .$$

Hence, using Fact A.4, we also also have that

$$\mathbb{E}\left[\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)\right)\right] = \mathbb{E}\left[\sup_{f^-\in\mathcal{F}^-}\left(\mathsf{m}_{\mathcal{S}}(f^-) - \mathsf{m}_{\mathcal{D}}(f^-)\right)\right] \le 2\mathbb{E}\left[\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right] \ . \tag{15}$$

Definition A.6 (*Bounded differences inequality*). Let $g : \mathcal{X}^{\ell} \to \mathbb{R}$ be a function of ℓ variables. The function g is said to satisfy the bounded difference inequality if and only if, for each i, $1 \le i \le \ell$, there is a nonnegative constant c_i such that:

$$\sup_{\substack{x_1,\ldots,x_\ell \\ x_i'\in\mathcal{X}}} |g(x_1,\ldots,x_\ell) - g(x_1,\ldots,x_{i-1},x_i',x_{i+1},\ldots,x_\ell)| \le c_i \ . \tag{16}$$

FACT A.7. *The quantity*

$$g(\mathcal{S}) = g(s_1,\ldots,s_\ell) = \sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right)$$

satisfies the bounded difference inequality with $c_i = \frac{b}{\ell}$, $1 \le i \le \ell$.

The following concentration inequality is a classic result.

THEOREM A.8 (MCDIARMID [1989]'S INEQUALITY). *Let* $g : \mathcal{X}^\ell \to \mathbb{R}$ *be a function satisfying the bounded difference inequality with constants* c_i, $1 \le i \le \ell$. *Let* $x_1, \ldots, x_\ell$ *be* ℓ independent *random variables taking value in* $\mathcal{X}$. *Then we have*

$$\Pr\left(g(x_1, \ldots, x_\ell) - \mathbb{E}[g] > t\right) \le e^{-2t^2/C},$$

where $C = \sum_{i=1}^{\ell} c_i^2$.

We now prove Thm. 3.3 for the more general case of functions to $[0, b]$. We can recover the results in the statement of Thm. 3.3 by setting $b = 1$. The proof is informed by the one for [Oneto et al. 2013, Thm. 3.10].

For the purpose of extending the proof to functions to $[0, b]$, redefine the function $g_{\mathcal{S}}$ from (4) as

$$g_{\mathcal{S}}(x, y) = 2\exp\left[-\frac{2\ell}{b^2}x^2(y - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))^2\right] + \exp\left[-\frac{\ell}{b}\left((1-x)y + 2x\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right)\phi\left(\frac{2\mathsf{R}_{\mathcal{F}}(\mathcal{S})}{(1-x)y + 2x\mathsf{R}_{\mathcal{F}}(\mathcal{S})} - 1\right)\right] . \quad (17)$$

The optimization problem from (5) stays the same, but using this definition of $g_{\mathcal{S}}$.

PROOF OF THM. 3.3. Fix $x \in [0, 1]$ and $\xi \in (2\mathsf{R}_{\mathcal{F}}(\mathcal{S}), 1]$. Define the following three events:

— $E_1 =$ "$\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right) \le \mathbb{E}\left[\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right)\right] + x(\xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))$";
— $E_2 =$ "$\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)\right) \le \mathbb{E}\left[\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)\right)\right] + x(\xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))$";
— $E_3 =$ "$\mathbb{E}[\mathsf{R}_{\mathcal{F}}(\mathcal{S})] \le \mathsf{R}_{\mathcal{F}}(\mathcal{S}) + \frac{1-x}{2}(\xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))$";

When all these events hold simultaneously, we have, using Fact A.4, Lemma A.5, and (15), that

$$\sup_{f\in\mathcal{F}}\left|\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right| = \max\left\{\sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right), \sup_{f\in\mathcal{F}}\left(\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)\right)\right\} \le 2\mathbb{E}\left[\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right] + x(\xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S})) \le 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}) + \xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}) \le \xi \ .$$

Using the union bound, we have that the probability of the *complementary* event "$\sup_{f\in\mathcal{F}}\left|\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right| > \xi$" is upper bounded by the sum of the probabilities of the events *complementary* to E_1, E_2, and E_3. These probabilities can be obtained from Thms. A.3 and A.8, using also Lemma A.2 and Fact A.7, and we have:

$$\Pr\left(\sup_{f\in\mathcal{F}}\left|\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)\right| > \xi\right) \le 2\exp\left(-\frac{2\ell}{b^2}x^2(\xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))^2\right) + \exp\left(-\frac{2\ell}{b}\mathbb{E}\left[\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right]\phi\left(-\frac{(1-x)\left(\xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right)}{2\mathbb{E}\left[\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right]}\right)\right) . \quad (18)$$

Since the second exponential on the r.h.s. of (18) *increases* as $\mathbb{E}[\mathsf{R}_{\mathcal{F}}(\mathcal{S})]$ *decreases*, we can consider the worst case that happens when E_3 does not hold and it is $\mathbb{E}[\mathsf{R}_{\mathcal{F}}(\mathcal{S})] = \mathsf{R}_{\mathcal{F}}(\mathcal{S}) +$

$\frac{1-x}{2}(\xi - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))$, and we obtain:

$$\Pr\left(\sup_{f\in\mathcal{F}}|\mathsf{m}_{\mathcal{S}}(f)-\mathsf{m}_{\mathcal{D}}(f)|>\xi\right) \leq 2\exp\left(-\frac{2\ell}{b^2}x^2(\xi-2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))^2\right) + \exp\left(-\frac{2\ell}{b}\left(\frac{(1-x)\xi}{2}+x\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right)\phi\left(-\frac{(1-x)\left(\xi-2\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right)}{(1-x)\xi+2x\mathsf{R}_{\mathcal{F}}(\mathcal{S})}\right)\right) .$$

The r.h.s. of the above equation is $g_{\mathcal{S}}(x,\xi)$ for $g_{\mathcal{S}}$ as in (17), which is bounded by η for all feasible solutions (x,ξ) of the optimization problem (5) and this allows us to obtain the thesis. □

We now prove Thm. 3.4 for the more general case of functions in $[0,b]$. We can recover the results in the statement of Thm. 3.4 by setting $b=1$. The proof follows, for the most part, the same steps as in the proof for [Oneto et al. 2013, Thm. 3.11].

Proof of Thm. 3.4. Assume that the following three events all hold simultaneously:

— $E_1 =$ "$\sup_{f\in\mathcal{F}}(\mathsf{m}_{\mathcal{S}}(f)-\mathsf{m}_{\mathcal{D}}(f)) \leq \mathbb{E}\left[\sup_{f\in\mathcal{F}}(\mathsf{m}_{\mathcal{S}}(f)-\mathsf{m}_{\mathcal{D}}(f))\right] + b\sqrt{\frac{\ln(3/\eta)}{2\ell}}$";
— $E_2 =$ "$\sup_{f\in\mathcal{F}}(\mathsf{m}_{\mathcal{D}}(f)-\mathsf{m}_{\mathcal{S}}(f)) \leq \mathbb{E}\left[\sup_{f\in\mathcal{F}}(\mathsf{m}_{\mathcal{D}}(f)-\mathsf{m}_{\mathcal{S}}(f))\right] + b\sqrt{\frac{\ln(3/\eta)}{2\ell}}$";
— $E_3 =$ "$\mathbb{E}[\mathsf{R}_{\mathcal{F}}(\mathcal{S})] \leq \mathsf{R}_{\mathcal{F}}(\mathcal{S}) + \sqrt{\mathbb{E}[\mathsf{R}_{\mathcal{F}}(\mathcal{S})]\frac{b\ln(3/\eta)}{\ell}}$";

From Thms. A.3 and A.8, using also Lemma A.2 and Fact A.7, we have that each of the complementary events to E_1, E_2, and E_3 holds with probability at most $\eta/3$, hence the event $E_1\cap E_2\cap E_3$ holds with probability at least $1-\eta$.

We then proceed as in the proof of Thm. 3.3 to obtain

$$\sup_{f\in\mathcal{F}}|\mathsf{m}_{\mathcal{S}}(f)-\mathsf{m}_{\mathcal{D}}(f)| \leq 2\mathbb{E}\left[\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right] + b\sqrt{\frac{\ln(3/\eta)}{2\ell}} . \tag{19}$$

In the rest of the proof, we show how to bound $\mathbb{E}\left[\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right]$ using the sample-dependent quantity $\mathsf{R}_{\mathcal{F}}(\mathcal{S})$. Given that E_3 is verified, and, for any $\alpha>0$, it holds that

$$\sqrt{xy} \leq \frac{\alpha}{\sqrt{2}}x + \frac{1}{\alpha\sqrt{2}}y,$$

we have

$$\mathbb{E}[\mathsf{R}_{\mathcal{F}}(\mathcal{S})] \leq \min_{\alpha\in(0,\sqrt{2})}\left[\frac{\sqrt{2}\mathsf{R}_{\mathcal{F}}(\mathcal{S})}{\sqrt{2}-\alpha} + \frac{b\ln(3/\eta)}{\ell\alpha(\sqrt{2}-\alpha)}\right] . \tag{20}$$

The minimum on the r.h.s. is attained for

$$\alpha^* = \frac{\sqrt{b\ln(3/\eta)(b\ln(3/\eta)+2\ell\mathsf{R}_{\mathcal{F}}(\mathcal{S}))}-b\ln(3/\eta)}{\sqrt{2}\ell\mathsf{R}_{\mathcal{F}}(\mathcal{S})} .$$

We plug α^* into the argument of the min on the r.h.s. of (20), and then use the resulting bound to $\mathbb{E}[\mathsf{R}_{\mathcal{F}}(\mathcal{S})]$ in the r.h.s. of (19) to obtain the thesis:

$$\sup_{f\in\mathcal{F}} |\mathsf{m}_{\mathcal{S}}(f) - \mathsf{m}_{\mathcal{D}}(f)| \leq 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}) + 2\frac{b\ln(3/\eta) + \sqrt{b\ln(3/\eta)(b\ln(3/\eta) + 2\ell\mathsf{R}_{\mathcal{F}}(\mathcal{S}))}}{\ell} + b\sqrt{\frac{\ln(3/\eta)}{2\ell}} \ .$$

□

B. PSEUDODIMENSION

Before introducing the pseudodimension, we must recall some notions and results about the Vapnik-Chervonenkis (VC) dimension. We refer the reader to the books by Shalev-Shwartz and Ben-David [2014] and by Anthony and Bartlett [1999] for an in-depth exposition of VC-dimension and pseudodimension.

Let D be a domain and let $\mathcal{R}$ be a collection of subsets of D ($\mathcal{R} \subseteq 2^D$). We call $\mathcal{R}$ a *rangeset on* D. Given $A \subseteq D$, the *projection of* $\mathcal{R}$ *on* A is $P_{\mathcal{R}}(A) = \{R \cap A \ : \ R \in \mathcal{R}\}$. When $P_{\mathcal{R}}(A) = 2^A$, we say that A is *shattered* by $\mathcal{R}$. Given $B \subseteq D$, the *empirical VC-dimension* of $\mathcal{R}$, denoted as $\mathsf{EVC}(\mathcal{R}, B)$ is the size of the largest subset of B that can be shattered. The *VC-dimension* of $\mathcal{R}$, denoted as $\mathsf{VC}(\mathcal{R})$ is defined as $\mathsf{VC}(\mathcal{R}) = \mathsf{EVC}(\mathcal{R}, D)$.

Let $\mathcal{F}$ be a class of functions from some domain D to $[0,1]$. Consider, for each $f \in \mathcal{F}$, the subset R_f of $D \times [0,1]$ defined as

$$R_f = \{(x,t) \ : \ t \leq f(x)\} \ .$$

We define a rangeset $\mathcal{F}^+$ on $D \times [0,1]$ as

$$\mathcal{F}^+ = \{R_f, f \in \mathcal{F}\} \ .$$

The *empirical pseudodimension* [Pollard 1984] of $\mathcal{F}$ on a subset $B \subseteq D$, denoted as $\mathsf{EPD}_{\mathcal{F}}(B)$, is the empirical VC-dimension of $\mathcal{F}^+$:

$$\mathsf{EPD}_{\mathcal{F}}(B) = \mathsf{EVC}(\mathcal{F}^+, B) \ .$$

The pseudodimension of $\mathcal{F}$, denoted as $\mathsf{PD}(\mathcal{F})$ is the VC-dimension of $\mathcal{F}^+$ [Anthony and Bartlett 1999, Sect. 11.2]:

$$\mathsf{PD}(\mathcal{F}) = \mathsf{VC}(\mathcal{F}^+) \ .$$

The following two technical lemmas are, to the best of our knowledge, new. We use them later to bound the pseudodimension of a family of functions related to betweenness centrality.

LEMMA B.1. *Let $B \subseteq D \times [0,1]$ be a set that is shattered by $\mathcal{F}^+$. Then B can contain* at most one *element $(d,x) \in D \times [0,1]$ for each $d \in D$.*

PROOF. Let $d \in D$ and consider any two distinct values $x_1, x_2 \in [0,1]$. Let, w.l.o.g., $x_1 < x_2$ and let $B = \{(\tau, x_1), (\tau, x_2)\}$. From the definitions of the ranges, there is no $R \in \mathcal{F}^+$ such that $R \cap B = \{(d, x_1)\}$, therefore B can not be shattered, and so neither can any of its supersets, hence the thesis. □

LEMMA B.2. *Let $B \subseteq D \times [0,1]$ be a set that is shattered by $\mathcal{F}^+$. Then B does not contain* any *element in the form $(d,0)$, for any $d \in D$.*

PROOF. For any $d \in D$, $(d,0)$ is contained in every $R \in \mathcal{F}^+$, hence given a set $B = \{(d,0)\}$ it is impossible to find a range $R_\emptyset$ such that $B \cap R_\emptyset = \emptyset$, therefore B can not be shattered, nor can any of its supersets, hence the thesis. □

C. RELATIVE-ERROR APPROXIMATION

In this section we discuss how to obtain an upper bound the supremum of a specific relative (i.e., multiplicative) deviation of sample means from their expectations, for a family $\mathcal{F}$ of functions from a domain $\mathcal{D}$ to $[0,1]$.

Let $\mathcal{S} = \{s_1, \ldots, s_\ell\}$ be a collection of ℓ elements from $\mathcal{D}$. Given a parameter $p \in (0,1]$, we are interested, specifically in giving probabilistic bounds to the quantity

$$\sup_{f \in \mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{\max\{p, \mathsf{m}_{\mathcal{D}}(f)\}} \ . \tag{21}$$

This quantity is inspired by the definition of *relative* (p,ε)*-approximations* [Har-Peled and Sharir 2011].

Li et al. [2001] used *pseudodimension* to study the quantity

$$\sup_{f \in \mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{\mathsf{m}_{\mathcal{D}}(f) + \mathsf{m}_{\mathcal{S}}(f) + p} \ . \tag{22}$$

Har-Peled and Sharir [2011] derived their concept of relative (p,ε)-approximation from this quantity and were only concerned with binary functions. The quantity in (22) has been studied often in the literature of statistical learning theory, see for example [Anthony and Bartlett 1999, Sect. 5.5], [Boucheron et al. 2005, Sect. 5.1], and [Haussler 1992], while other works (e.g., [Boucheron et al. 2005, Sect. 5.1], [Cortes et al. 2013], [Anthony and Shawe-Taylor 1993], and [Bartlett and Lugosi 1999]) focused on the quantity

$$\sup_{f \in \mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{\sqrt{\mathsf{m}_{\mathcal{D}}(f)}} \ .$$

We focus on the quantity in (21) because more useful in our specific case.

It is easy to see that

$$\sup_{f \in \mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{\max\{p, \mathsf{m}_{\mathcal{D}}(f)\}} \leq \sup_{f \in \mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{p} \ . \tag{23}$$

For any $f \in \mathcal{F}$, define $f_{/p}$ to be the function from $\mathcal{D}$ to $[0, 1/p]$ such that $f_{/p}(x) = f(x)/p$, for any $x \in \mathcal{D}$. Define the family

$$\mathcal{F}^{/p} = \{f_{/p}, f \in \mathcal{F}\} \ .$$

We have from (23) that

$$\sup_{f \in \mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{\max\{p, \mathsf{m}_{\mathcal{D}}(f)\}} \leq \sup_{f_{/p} \in \mathcal{F}^{/p}} |\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)| \ .$$

Therefore, a bound to the r.h.s. of this equation implies a bound to the quantity from (21) that we are interested in. We can use the results from Appendix A to obtain such a bound to the supremum of the absolute deviations of the sample means from their expectations for the functions in $\mathcal{F}^{/p}$. In particular, we can use the versions of Thms. 3.3 and 3.4 for functions in $[0,b]$ that we proved in Appendix A to derive the following results for the quantity from (21), using $b = 1/p$.

For the rest of this section, let $\mathcal{S}$ be a collection of ℓ elements of $\mathcal{D}$ sampled independently. Let, for any $x \in [0,1]$ and $y \in [2\mathsf{R}_{\mathcal{F}}(\mathcal{S}), 1]$ let

$$\begin{aligned} g_{\mathcal{S}}(x,y) = {} & 2\exp\left[-2p^2\ell x^2 (y - 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}))^2\right] \\ & + \exp\left[-p\ell\left((1-x)y + 2x\mathsf{R}_{\mathcal{F}}(\mathcal{S})\right)\phi\left(\frac{2\mathsf{R}_{\mathcal{F}}(\mathcal{S})}{(1-x)y + 2x\mathsf{R}_{\mathcal{F}}(\mathcal{S})} - 1\right)\right] \ . \end{aligned} \tag{24}$$

THEOREM C.1. *Let $\eta \in (0,1)$ and let ξ^* be the optimal value (if it exists) of the minimization problem* (5)*, using $g_{\mathcal{S}}$ as in* (24)*. Then*

$$\Pr\left(\sup_{f\in\mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{\max\{p, \mathsf{m}_{\mathcal{D}}(f)\}} \geq \xi^*\right) \leq \eta \ . \tag{25}$$

THEOREM C.2. *Let $\eta \in (0,1)$. Then, with probability at least $1-\eta$,*

$$\sup_{f\in\mathcal{F}} \frac{|\mathsf{m}_{\mathcal{D}}(f) - \mathsf{m}_{\mathcal{S}}(f)|}{\max\{p, \mathsf{m}_{\mathcal{D}}(f)\}} \leq 2\mathsf{R}_{\mathcal{F}}(\mathcal{S}) + 2\frac{p^{-1}\ln(3/\eta) + \sqrt{(p^{-1}\ln(3/\eta) + 2\ell\mathsf{R}_{\mathcal{F}}(\mathcal{S}))p^{-1}\ln(3/\eta)}}{\ell}$$
$$+ p^{-1}\sqrt{\frac{\ln(3/\eta)}{2\ell}} \ .$$

ACKNOWLEDGMENTS

The authors are thankful to Elisabetta Bergamini and Christian Staudt for their help with the NetworKit code, and to Emanuele Natale and Michele Borassi for finding a mistake, now corrected, in one of our proofs.

This work was supported in part by NSF grant IIS-1247581 and NIH grant R01-CA180776.

References

D. Anguita, A. Ghio, L. Oneto, and S. Ridella. A deep connection between the Vapnik-Chervonenkis entropy and the Rademacher complexity. *IEEE Transactions on Neural Networks and Learning Systems*, 25(12):2202–2211, 2014.

J. M. Anthonisse. The rush in a directed graph. Technical Report BN 9/71, Stichting Mathematisch Centrum, Amsterdam, Netherlands, 1971.

M. Anthony and P. L. Bartlett. *Neural Network Learning – Theoretical Foundations*. Cambridge University Press, 1999.

M. Anthony and J. Shawe-Taylor. A result of Vapnik with applications. *Discrete Applied Mathematics*, 47(3):207–217, 1993.

D. A. Bader, S. Kintali, K. Madduri, and M. Mihail. Approximating betweenness centrality. In A. Bonato and F. Chung, editors, *Algorithms and Models for the Web-Graph*, volume 4863 of *Lecture Notes in Computer Science*, pages 124–137. Springer Berlin Heidelberg, 2007.

P. L. Bartlett and G. Lugosi. An inequality for uniform deviations of sample averages from their means. *Statistics & Probability Letters*, 44(1):55–62, 1999.

E. Bergamini and H. Meyerhenke. Fully-dynamic approximation of betweenness centrality. In *Proceedings of the 23rd European Symposium on Algorithms*, ESA '15, pages 155–166, 2015.

E. Bergamini and H. Meyerhenke. Approximating betweenness centrality in fully-dynamic networks. *Internet Mathematics*, 12(5):281–314, 2016.

E. Bergamini, H. Meyerhenke, and C. L. Staudt. Approximating betweenness centrality in large evolving networks. In *17th Workshop on Algorithm Engineering and Experiments*, ALENEX '15, pages 133–146. SIAM, 2015.

S. Boucheron, G. Lugosi, and P. Massart. A sharp concentration inequality with application. *Random Structures & Algorithms*, 16(3):277–292, 2000.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: A survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

U. Brandes and C. Pich. Centrality estimation in large networks. *International Journal of Bifurcation and Chaos*, 17(7):2303–2318, 2007.

C. Cortes, S. Greenberg, and M. Mohri. Relative deviation learning bounds and generalization with unbounded loss functions. *CoRR*, abs/1310.5796, Oct 2013.

T. Elomaa and M. Kääriäinen. Progressive Rademacher sampling. In R. Dechter and R. S. Sutton, editors, *AAAI/IAAI*, pages 140–145. AAAI Press / The MIT Press, 2002.

D. Erdős, V. Ishakian, A. Bestavros, and E. Terzi. A divide-and-conquer algorithm for betweenness centrality. In *SIAM International Conference on Data Mining*, SDM '15, pages 433–441. SIAM, 2015.

L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40:35–41, 1977.

R. Geisberger, P. Sanders, and D. Schultes. Better approximation of betweenness centrality. In *10th Workshop on Algorithm Engineering and Experiments*, ALENEX '08, pages 90–100. SIAM, 2008.

O. Green, R. McColl, and D. A. Bader. A fast algorithm for streaming betweenness centrality. In *2012 International Conference on Privacy, Security, Risk and Trust*, PASSAT '12, pages 11–20. IEEE, sep 2012.

S. Har-Peled and M. Sharir. Relative (p, ε)-approximations in geometry. *Discrete & Computational Geometry*, 45(3):462–496, 2011.

D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

T. Hayashi, T. Akiba, and Y. Yoshida. Fully dynamic betweenness centrality maintenance on massive networks. *Proceedings of the VLDB Endowment*, 9(2):48–59, 2015.

R. Jacob, D. Koschützki, K. Lehmann, L. Peeters, and D. Tenfelde-Podehl. Algorithms for centrality indices. In U. Brandes and T. Erlebach, editors, *Network Analysis*, volume 3418 of *Lecture Notes in Computer Science*, pages 62–82. Springer Berlin Heidelberg, 2005.

M. A. Jenkins and J. F. Traub. Algorithm 419: Zeros of a complex polynomial [c2]. *Communications of the ACM*, 15(2):97–99, Feb. 1972.

S. Ji and Z. Yan. Refining approximating betweenness centrality based on samplings. *CoRR*, abs/1608.04472, 2016.

S. G. Johnson. The NLopt nonlinear-optimization package. http://ab-initio.mit.edu/nlopt, 2014.

M. Kas, M. Wachs, K. M. Carley, and L. R. Carley. Incremental algorithm for updating betweenness centrality in dynamically growing networks. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '13, pages 33–40. IEEE/ACM, 2013.

V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, July 2001.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30(1):1–50, 2002.

N. Kourtellis, T. Alahakoon, R. Simha, A. Iamnitchi, and R. Tripathi. Identifying high betweenness centrality nodes in large social networks. *Social Network Analysis and Mining*, 3(4):899–914, 2012.

N. Kourtellis, G. D. F. Morales, and F. Bonchi. Scalable online betweenness centrality in evolving graphs. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2494–2506, 2015.

M.-J. Lee, J. Lee, J. Y. Park, R. H. Choi, and C.-W. Chung. QUBE: A quick algorithm for updating betweenness centrality. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 351–360. IW3C2, 2012.

J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

Y. Li, P. M. Long, and A. Srinivasan. Improved bounds on the sample complexity of learning. *Journal of Computer and System Sciences*, 62(3):516–527, 2001.

C. McDiarmid. On the method of bounded differences. *Surveys in combinatorics*, 141(1): 148–188, 1989.

M. Nasre, M. Pontecorvi, and V. Ramachandran. Betweenness centrality – incremental and faster. In *International Symposium on Mathematical Foundations of Computer Science*, MFCS '14, pages 577–588, 2014a.

M. Nasre, M. Pontecorvi, and V. Ramachandran. Decremental all-pairs ALL shortest paths and betweenness centrality. In *Proceedings of the 25th International Symposium on Algorithms and Computation*, ISAAC '14, pages 766–778, 2014b.

M. E. J. Newman. *Networks – An Introduction*. Oxford University Press, 2010.

L. Oneto, A. Ghio, D. Anguita, and S. Ridella. An improved analysis of the Rademacher data-dependent bound using its self bounding property. *Neural Networks*, 44:107–111, 2013.

L. Oneto, A. Ghio, S. Ridella, and D. Anguita. Global Rademacher complexity bounds: From slow to fast convergence rates. *Neural Processing Letters*, 43(2):567–602, 2016.

D. Pollard. *Convergence of stochastic processes*. Springer-Verlag, 1984.

M. Pontecorvi and V. Ramachandran. Fully dynamic betweenness centrality. In *Proceedings of the 26th International Symposium on Algorithms and Computation*, ISAAC '15, pages 331–342, 2015.

M. Riondato and E. M. Kornaropoulos. Fast approximation of betweenness centrality through sampling. *Data Mining and Knowledge Discovery*, 30(2):438–475, 2016.

M. Riondato and E. Upfal. Mining frequent itemsets through progressive sampling with Rademacher averages. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1005–1014. ACM, 2015. Extended version available from http://matteo.rionda.to/papers/RiondatoUpfal-FrequentItemsetsSamplingRademacher-KDD.pdf.

M. Riondato and E. Upfal. Approximating betweenness centrality in static and dynamic graphs with Rademacher averages. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1145–1154. ACM, 2016.

A. E. Sarıyüce, E. Saule, K. Kaya, and U. V. Çatalyürek. Shattering and compressing networks for betweenness centrality. In *SIAM International Conference on Data Mining*, SDM '13, pages 686–694. SIAM, 2013.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

C. L. Staudt, A. Sazonovs, and H. Meyerhenke. NetworKit: An interactive tool suite for high-performance network analysis. *Network Science*, to appear, 2016.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1999.

Received Month Year; revised Month Year; accepted Month Year