Proxying Betweenness Centrality Rankings in Temporal Networks

Ruben Becker
Ca' Foscari University of Venice, Italy

Pierluigi Crescenzi
Gran Sasso Science Institute, L'Aquila, Italy

Antonio Cruciani
Gran Sasso Science Institute, L'Aquila, Italy

Bojana Kodric
Ca' Foscari University of Venice, Italy

Abstract

Identifying influential nodes in a network is arguably one of the most important tasks in graph mining and network analysis. A large variety of centrality measures, all aiming at correctly quantifying a node's importance in the network, have been formulated in the literature. One of the most cited ones is the *betweenness centrality*, formally introduced by Freeman (Sociometry, 1977). On the other hand, researchers have recently been very interested in capturing the dynamic nature of real-world networks by studying *temporal graphs*, rather than static ones. Clearly, centrality measures, including the betweenness centrality, have also been extended to temporal graphs. Buß et al. (KDD, 2020) gave algorithms to compute various notions of temporal betweenness centrality, including the perhaps most natural one – *shortest temporal betweenness*. Their algorithm computes centrality values of all nodes in time $O(n^3T^2)$, where n is the size of the network and T is the total number of time steps. For real-world networks, which easily contain tens of thousands of nodes, this complexity becomes prohibitive. Thus, it is reasonable to consider proxies for shortest temporal betweenness rankings that are more efficiently computed, and, therefore, allow for measuring the relative importance of nodes in very large temporal graphs. In this paper, we compare several such proxies on a diverse set of real-world networks. These proxies can be divided into *global* and *local* proxies. The considered global proxies include the exact algorithm for static betweenness (computed on the underlying graph), prefix foremost temporal betweenness of Buß et al., which is more efficiently computable than shortest temporal betweenness, and the recently introduced approximation approach of Santoro and Sarpe (WWW, 2022). As all of these global proxies are still expensive to compute on very large networks, we also turn to more efficiently computable local proxies. Here, we consider temporal versions of the ego-betweenness in the sense of Everett and Borgatti (Social Networks, 2005), standard degree notions, and a novel temporal degree notion termed the *pass-through degree*, that we introduce in this paper and which we consider to be one of our main contributions. We show that the pass-through degree, which measures the number of pairs of neighbors of a node that are temporally connected through it, can be computed in nearly linear time for all nodes in the network and we experimentally observe that it is surprisingly competitive as a proxy for shortest temporal betweenness.

2012 ACM Subject Classification Theory of computation → Shortest paths; Networks → Network algorithms; Mathematics of computing → Graph algorithms

Keywords and phrases node centrality, betweenness, temporal graphs, graph mining

Digital Object Identifier 10.4230/LIPIcs.SEA.2023.6

Supplementary Material *Software (Source Code)*: `https://github.com/piluc/TSBProxy` archived at `swh:1:dir:02ece196b54034f669645bc2b220c28d43cbc423`

Funding *Pierluigi Crescenzi*: Partially supported by the French government through the UCAJEDI, while visiting COATI, INRIA d'Université Côte d'Azur, Sophia Antipolis, France.

1 Introduction

Centrality measures are notions for evaluating the importance of nodes in networks, used in network analysis and graph theory. The aim is to assign real values to all the nodes, in such a way that the values are monotonously dependent of the nodes' importance, i.e., more important nodes should have higher centrality values. It is evident that this task is among the most important ones in network analysis. Consequently, there is a vast variety of centrality notions in the existing literature. Popular measures include spectral notions, such as Katz's index [24], Seeley's index [41], and PageRank [10], and combinatorial notions, like the straightforward concept of degree centrality, the closeness centrality [4], the harmonic centrality [32, 6], and the betweenness centrality [18]. This diversity of notions indicates that there is no consensus among researchers on which notion is the "correct one". While Boldi and Vigna [6] provide an axiomatic approach to this question, their work mainly transfers the discord from which centrality notion to use to the question of which axioms to agree upon. In fact, the choice of which centrality notion to employ is mainly dependent on the application which may stem from a diverse set of fields [30, 17, 11]. In many scenarios, the considered networks are characterized by the following challenges: (1) they are very large and (2) they are dynamic or temporal, i.e., they change over time. In the context of these two challenges it is, thus, essential to consider temporal variants of the most important centrality notions, alongside algorithms for computing them, that have a good scaling behaviour. In this work, we focus on the *betweenness centrality*, which is certainly among the most used and most cited centrality notions, and study it in the context of these challenges.

Buß et al. [12] defined the *shortest temporal betweenness* as a temporal counterpart of the *betweenness centrality*, and gave an algorithm to compute all centrality values in time $O(n^3T^2)$, where n is the size of the network and T is the total number of time steps. For nowadays networks, such time complexity easily becomes infeasible. Thus, it is reasonable to consider proxies for shortest temporal betweenness rankings that are more efficiently computed. In this work, we use the following general approach. We employ a set of competitor algorithms that we each use as proxies for temporal betweenness rankings, i.e., for each algorithm, we compute a complete ranking of the nodes and evaluate how this ranking relates to the "correct" ranking. While different scenarios may exist, centrality values are frequently used to rank nodes and our proxy notion is motivated exactly by such applications.

Some of the considered proxies have the property that they still try to capture the global nature inherent in the definition of the shortest temporal betweenness and, as a consequence, still suffer from a comparatively bad running time, meaning that their running times are far from linear in the input size. Note however that, as argued, e.g., by Teng [47], in the age of Big Data, an algorithm should be considered efficient or scalable if its time complexity is nearly-linear. In fact, there is even theoretical evidence, in form of several conditional lower bound results [2, 7], for believing that no such algorithm is achievable, even for *approximately* computing the betweenness values in *static* graphs. We thus shift our focus away from these *global proxies* towards *local proxies* for shortest temporal betweenness rankings. We classify a proxy as local if the centrality values of nodes are completely determined by the induced subgraph of their neighborhood (including themselves).

For measuring proxy quality, we employ several different metrics, most prominently a weighted version of Kendall's τ correlation coefficient and the intersection of the top-k ranked nodes (for different values of k). Note that the latter is directly translatable into the Jaccard similarity of the top ranked nodes. We would like to stress here that it is quite uncomplicated to show that all proxies considered in the present work can perform arbitrarily bad on adversarial examples (in terms of all considered metrics) and no reasonable theoretical guarantees can therefore be given for their ranking quality.

A diverse set of temporal betweenness notions has been defined in the literature (see Section 1.2). Clearly, if the notion of centrality is already vague in static graphs, it becomes even more so in the temporal setting, where in addition the time dimension has to be considered. In this study, we focus on the definition given by Buß et al. [12] as it arguably represents one of the most "natural" and direct temporal analogues of the static betweenness (once the notion of distance has been defined).

1.1 Contribution

We compare a variety of approaches for proxying shortest temporal betweenness rankings in terms of their scalability and output quality. We start our study in Section 3 with a comparison of the following proxies: (1) exact algorithm for the static betweenness computed on the underlying graph, (2) the more efficiently computable prefix foremost temporal betweenness of Buß et al. and (3) the recently introduced (absolute) approximation approach of Santoro and Sarpe [39]. Our evaluation indicates that the static betweenness rankings turn out to be quite competitive, the performance of the prefix foremost temporal betweenness seems somewhat inconsistent, while the quality of the ranking returned by the considered temporal betweenness approximation algorithm very much depends on the provided time.

Next, motivated by the fact that static degree centrality is often compared to other centrality measures, we follow this approach in the temporal setting. In Section 4, we describe our main theoretical contribution: the *pass-through degree*, a new *temporal* degree notion which we believe to be interesting in its own right. Informally the pass-through degree of a node v measures the number of neighbor pairs of v that are temporally connected through v, i.e., that have a temporal path of length two between them that *passes through* v. We proceed by giving an algorithm that computes the pass-through degree of all nodes in a given (directed) temporal graph in $O(M \log m)$ time, where M is the number of temporal arcs and m the number of arcs in the underlying static graph. In other words, the proposed algorithm is scalable in the sense of Teng [47].

In Section 5 we compare the following set of local proxies in terms of their efficiency and quality: (1) temporal versions of the ego-betweenness in the sense of Everett and Borgatti [16], which entails to compute the betweenness centrality values of the nodes in their respective ego-networks (the induced subgraph of a node's neighborhood including himself) (2) the pass-through degree, and (3) the approximation algorithm for temporal betweenness centrality also used as one of the global proxies in Section 3, as it is the only choice from that section that offers scalability in terms of computation time. We note that the pass-through degree falls somewhere between the simple degree notions and the ego-betweenness notion in terms of complexity. Our evaluation here indicates that the ego-networks can be of comparable size as the whole network and, thus, prohibitively large on some data sets, the pass-through degree usually does not perform worse than the ego-variants and is at the same time much faster, while the considered approximation algorithm for temporal betweenness has a more inconsistent performance over different data sets.

Our experimental evaluation is based on a diverse set of real-world networks that includes almost all publicly available networks from the works of Buß et al. [12] and Santoro and Sarpe [39]. We did not include the `Karlsruhe` network [20] (used in [12]) because it does not appear to be available anymore. Moreover, we replaced `Mathoverflow` [29] network (used in [39]) by a bigger temporal network from a different domain to make the set of analyzed temporal graphs more diverse. Finally, we excluded `Ask Ubuntu` and `Super User` [29] (also analyzed in [39]) because of the excessive amount of time needed to compute their exact temporal betweenness rankings.

1.2 Further Related Work

The literature on centrality measures being vast, we restrict our attention to approaches that are closest to ours. We, thus, particularly focus on centrality notions in temporal graphs.

First of all, several works give introductions to temporal graphs that include surveys on temporal centrality measures (see, e.g., [23, 28, 40]). Nicosia et al. [33] introduced different temporal graph notions, such as temporal centralities, temporal motif, temporal clustering, temporal modularity, and temporal communities. Providing top-k algorithms for estimating temporal closeness centrality has also already been treated in the literature [15, 34]. Subsequently, a closeness variant based on bounded random-walks, related to the concept of influence spreading, has been proposed by Haddadan et al. [22]. Furthermore, Tang et al. [46] introduced temporal variants of both closeness and betweenness centrality based on foremost temporal paths, and experimentally showed the effectiveness of such metrics in spotting influential users in real-world temporal graphs. Building upon this direction, Tang et al. [44] used the notion of temporal closeness to provide an empirical analysis of the containment of malware in real-world mobile phone networks. The Katz centrality [24] has been adapted to the temporal setting [21, 5] as well, while Rozenshtein et al. [37] defined the temporal PageRank by replacing random walks with temporal random walks.

Tsalouchidou et al. [48] extended the well-known Brandes algorithm [9] to allow for distributed computation of betweenness in temporal graphs. Specifically, they studied shortest-fastest paths, considering the bi-objective of shortest length and shortest duration. Buß et al. [12] analysed the temporal betweenness centrality considering several temporal path optimality criteria, such as shortest (foremost), foremost, fastest, and prefix-foremost, along with their computational complexities. They showed that, when considering paths with increasing time labels, the foremost and fastest temporal betweenness variants are $\#P$-hard, while the shortest and shortest foremost ones can be computed in $O(n^3T^2)$, and the prefix-foremost one in $O(nM \log M)$. Here n is the number of nodes and M the number of temporal arcs. The complexity analysis of these measures has been further refined since [38].

Santoro and Sarpe [39] provide a sampling-based approximation algorithm for estimating the temporal betweenness centrality of nodes based on shortest path criterion, for situations in which the computational cost of computing exact values is too large.

Ghanem et al. [19] defined a temporal version of ego betweenness based on most recent paths, which are paths that give the most recent information to the destination vertex about the status of the source, i.e., no other path starts from the source at a later point in time. Their definition of temporal ego betweenness is snapshot based, i.e., it gives the ego betweenness of the temporal ego graph at a specific time instant. Simard et al. [42], on the other hand, studied a continuous-time scenario of the shortest paths betweenness.

Finally, Oettershagen et al. [35] defined a random temporal walks based centrality that quantifies the importance of a node by measuring its ability to obtain and distribute information in a temporal network. They provide exact and approximate algorithms for computing their centrality measures and compare it with the state-of-the-art temporal centralities, i.e., with PageRank [37], Katz [5], closeness [15, 34], and betweenness [12].

2 Preliminaries

We proceed by formally introducing the terminology and concepts that we use in what follows. For $k \in \mathbb{N}$, we let $[k] := \{1, \ldots, k\}$.

Static Graphs. We start by introducing standard *static*, i.e., non-temporal, graphs[1]. An *undirected graph* is an ordered pair $G = (V, E)$, where V is a set whose elements are called vertices or nodes, and E is a set of unordered pairs of vertices, whose elements are called edges. We denote by $N(u) = \{v \in V : \{u, v\} \in E\}$ the set of neighbors of a vertex $u \in V$. The *degree* of a vertex $u \in V$ is defined as $\mathrm{d}(u) := |N(u)|$. A *directed graph* is a an ordered pair $G = (V, A)$, where V is a set whose elements are called vertices or nodes, and A is a set of ordered pairs of vertices, whose elements are called arcs. We denote by $N^{\mathrm{in}}(u) = \{v \in V : (v, u) \in A\}$ and $N^{\mathrm{out}}(u) = \{w \in V : (u, w) \in A\}$ the set of *in-neighbors* and of *out-neighbors* of a vertex $u \in V$, respectively. For a subset of nodes $U \subseteq V$, we call $G[U] := (U, A')$, where $A' := \{(u, v) \in A : u, v \in U\}$, the *induced subgraph* of U. We note that an undirected graph can be modeled as a directed graph by introducing, for every edge $e = \{u, v\} \in E$, both arcs (u, v) and (v, u), resulting in the corresponding bidirected graph. We thus focus on directed graphs in what follows.

Temporal Graphs. A directed *temporal graph* is an ordered triple $\mathcal{G} = (V, A, \lambda)$, where (V, A) is a directed graph, called the *underlying graph* of the temporal graph $\mathcal{G}$, and $\lambda : A \to 2^{[T]}$ is a function assigning to every arc in A a finite set of elements from the set of *time labels* $[T]$.[2] We let $\mathcal{A} = \{(u, v, t) : (u, v) \in A, t \in \lambda(u, v)\}$ denote the set of *temporal arcs* of $\mathcal{G}$. Undirected temporal graphs can be modeled via directed graphs resulting in a bidirected underlying graph. Static graphs can be modeled by temporal graphs by defining $\lambda(a) := [T]$ for all arcs $a \in A$.

Temporal Betweenness. A *walk* from node u to node w in a static graph $G = (V, A)$ is a sequence $a_1, \dots, a_k$ such that $a_i = (v_i, v_{i+1}) \in A$, $v_1 = u$, and $v_{k+1} = w$. We call k the *length* of the walk. A *path* is a walk such that $v_i \neq v_j$ for all $i, j \in [k]$ with $i \neq j$. A *shortest path* from u to w is a path of minimum length among all paths from u to w. We denote by $\sigma_{u,w}$ the total number of shortest paths between u and w in G, while $\sigma_{u,w}(v)$ is the number of shortest paths between u and w that pass through v. The *betweenness* or *betweenness centrality* of a node v in G, formally introduced by Freeman [18] in 1977, is defined as

$$\mathrm{b}_G(v) := \sum_{u,w \in V \setminus \{v\} : \sigma_{u,w} \neq 0} \frac{\sigma_{u,w}(v)}{\sigma_{u,w}}.$$

A *temporal walk* from node u to node w in a temporal graph $\mathcal{G} = (V, A, \lambda)$ is a walk $a_1, \dots, a_k$ from u to w in the underlying graph $G = (V, A)$ such that there exist time labels $t_1, \dots, t_k$ with $t_1 < \dots < t_k$ and $t_i \in \lambda(a_i)$ for every $i \in [k]$. We call k the *length* of the temporal walk and t_k the *arrival time* of the walk at w. A *temporal path* is a temporal walk such that $v_i \neq v_j$ for all $i, j \in [k]$ with $i \neq j$. A *prefix temporal path* of a temporal path is its subpath starting at the same vertex. A *shortest temporal path* from u to w is a temporal path of minimum length among all temporal paths from u to w. Analogously to the static case, we denote by $\sigma^{\mathrm{temp}}_{u,w}$ the total number of shortest temporal paths between u and w in $\mathcal{G}$, while $\sigma^{\mathrm{temp}}_{u,w}(v)$ is the number of shortest temporal paths between u and w that pass through v. The *shortest temporal betweenness (centrality)* of a node v in the temporal graph $\mathcal{G}$ is defined as

[1] We use the terms "graph" and "network" interchangeably.

[2] The value T denotes the *life-time* of the temporal graph, and, without loss of generality for our purposes, we assume that, for any $t \in [T]$, there exists at least one temporal arc a such that $\lambda(a) = t$.

$$\mathrm{stb}_{\mathcal{G}}(v) := \sum_{\substack{u,w \in V\setminus\{v\}:\\ \sigma^{\mathrm{temp}}_{u,w}\neq 0}} \frac{\sigma^{\mathrm{temp}}_{u,w}(v)}{\sigma^{\mathrm{temp}}_{u,w}}.$$

Different notions of temporal betweenness were recently studied by Buß et al. [12]. Their foremost and fastest variants are both $\#P$-hard, making them very impractical. From the remaining variants, the shortest temporal betweenness seems to be the most natural one. We do not consider walk-based betweenness notions as we agree with Buß et al. that "paths are more suitable than walks for defining temporal betweenness centrality" [12]. Buß et al. [12] gave an algorithm to compute the shortest temporal betweenness of all nodes in time $O(n^3T^2)$. We will next introduce the notion of *prefix foremost temporal betweenness* from the work of Buß et al. as we will use it as a proxy for the shortest temporal betweenness. A *prefix foremost shortest path* from u to w is a shortest temporal path from u to w such that no other shortest temporal path has an earlier arrival time at w and such that its every prefix path satisfies the same property. Let $\tau^{\mathrm{temp}}_{u,w}$ now be the total number of prefix foremost shortest paths between u and w in $\mathcal{G}$ and let $\tau^{\mathrm{temp}}_{u,w}(v)$ be the number of those paths that pass through v. The prefix foremost temporal betweenness pftb of v is then defined analogously to the shortest temporal betweenness by replacing σ by τ , i.e.,

$$\mathrm{pftb}_{\mathcal{G}}(v) := \sum_{\substack{u,w \in V\setminus\{v\}:\\ \tau^{\mathrm{temp}}_{u,w}\neq 0}} \frac{\tau^{\mathrm{temp}}_{u,w}(v)}{\tau^{\mathrm{temp}}_{u,w}}.$$

Buß et al. give an algorithm for computing the prefix foremost temporal betweenness of all nodes in time $O(nM \log M)$, where n is the number of vertices and M the total number of temporal arcs.

Temporal Ego-Betweenness The *ego-network* $G^{[v]}$ of a node v in a static graph G is the induced subgraph of its in- and out-neighbors, i.e., $G^{[v]} := G[N^{\mathrm{in}}(v) \cup N^{\mathrm{out}}(v)]$. The *ego-betweenness (centrality)* of v is the betweenness of v in its ego-network, i.e., ego-b$(v) := \mathrm{b}_{G^{[v]}}(v)$. The ego-betweenness (in undirected graphs) was introduced by Everett and Borgatti [16] as a more tractable variant of betweenness. We extend their ego-betweenness to temporal graphs as follows. The *ego-network* $\mathcal{G}^{[v]}$ of a node v in a temporal graph $\mathcal{G}$ is the temporal graph with the underlying graph $G^{[v]} := G[N^{\mathrm{in}}(v) \cup N^{\mathrm{out}}(v)]$ and with the time label function λ' being the restriction of λ to arcs in $G^{[v]}$. The *ego-shortest temporal betweenness* of v is the shortest temporal betweenness of v in its ego-network, i.e., ego-stb$(v) := \mathrm{stb}_{\mathcal{G}^{[v]}}(v)$. Similarly, we define the *ego-prefix foremost temporal betweenness* of v as the prefix foremost temporal betweenness of v in its ego-network, i.e, ego-pftb$(v) := \mathrm{pftb}_{\mathcal{G}^{[v]}}(v)$.

Everett and Borgatti [16] propose an algorithm to compute the ego-betweenness of a single node in an undirected static graph via computation of the square of the incidence matrix of the node's ego-network. We note that in the worst case the ego-network is of the same size as the original graph. For computing the temporal ego-betweennesses of all nodes, this algorithm can thus be implemented in time $O(n^{\omega+1})$, where ω is matrix multiplication exponent, i.e., the smallest real number such that two $n \times n$ matrices can be multiplied within $O(n^{\omega+\varepsilon})$ field operations for all $\varepsilon > 0$. The current best bound on ω is 2.3728596 [3].

3 Global Proxies for Shortest Temporal Betweenness

In this section, we summarize the results of our experimental study on proxying the shortest temporal betweenness values in large real-world networks using global proxies. Recall that a proxy is global, if the centrality value of each node is not purely dependent on its neighborhood. Our general experimental approach here is as follows. We employ a set of competitor algorithms that we each use as a proxy for shortest temporal betweenness centrality rankings. That is, for each algorithm, we compute a complete ranking of the nodes and evaluate (using various metrics) how this ranking relates to the "correct" ranking computed by the algorithm of Buß et al. [12]. In what follows, we will call this benchmark algorithm TempBrandes for "Temporal Brandes algorithm". Recall that TempBrandes computes the shortest temporal betweenness values of all nodes in time $O(n^3T^2)$.

3.1 Experimental Setting

Global Proxies. As global proxies for shortest temporal betweenness, our study includes the following algorithms.

- **Brandes:** The classical algorithm of Brandes, which computes the static betweenness of all nodes in time $O(nm)$ on the underlying graph, i.e., the graph obtained by a union over all the time steps.[3]
- **Pref:** The algorithm of Buß et al. [12] for computing the prefix foremost temporal betweennesses pftb in $O(nM \log M)$.
- **Onbra:** The approximation algorithm of Santoro and Sarpe [39], which is a sampling technique for obtaining an absolute approximation of the shortest temporal betweenness values. The work that introduced this algorithm is rather vague in terms of how to choose the sample size, stating only that they choose it so as to make the algorithm run "within a fraction of the time required by the exact algorithm". In our study, we choose the number of samples such that the running time of Onbra is a tenth, a half and roughly equal to the running time of TempBrandes. We achieve this by first estimating the time taken per sample, and then computing the number of samples by dividing the (fraction of) time needed by TempBrandes with the computed estimate.

Besides Brandes, which is available in the *Graphs.jl* library, we implemented TempBrandes and all competitor algorithms in Julia. We chose to re-implement TempBrandes, Pref and Onbra because the available implementations of TempBrandes and Pref have issues with the number of paths in the tested networks, causing overflow errors (indicated by negative centralities). Since Onbra is based on the TempBrandes code, it results in the same errors. Our implementation uses a sparse matrix representation of the $n \times |T|$ table used in [12, 39], making the implemented algorithms space-efficient and allowing to compute the exact temporal shortest betweenness on big temporal graphs (for which the original version of the code gives out of memory errors). Furthermore, we noticed another error in TempBrandes and Pref, related to time relabeling causing an underestimation of centralities. Our code is available at `https://github.com/piluc/TSBProxy`.

Networks. We evaluate all of the above competitors on real-world temporal graphs of different nature, whose properties are summarized in Table 1. The networks come from two different domains.

[3] We are aware of fast approximation algorithms like Kadabra [8] for the computation of the static betweenness, but for our purpose here the efficiency of the exact algorithm is sufficient.

Table 1 The temporal networks used in our evaluation, where n denotes the number of nodes, m the number of arcs in the underlying static graph, M the number of temporal arcs, T the number of unique time labels, t_{STB} the execution time of TempBrandes, and n_e^{max} the maximum number of nodes in the ego network (type **D** stands for directed and **U** for undirected). The networks are sorted in increasing order with respect to t_{STB}.

Data set	n	m	M	T	t_{STB}	n_e^{max}	Type	Source
`Hypertext 2009`	113	4392	41636	5246	263	99	U	[1]
`High school 2011`	126	3418	57078	5609	446	56	U	[1]
`Hospital ward`	75	2278	64848	9453	659	62	U	[1]
`College msg`	1899	20296	59798	58911	894	256	D	[29]
`Wiki elections`	7115	103680	106985	101012	1192	1066	D	[29]
`High school 2012`	180	4440	90094	11273	1345	57	U	[1]
`Digg reply`	30360	85247	86203	82641	1762	284	D	[36]
`Infectious`	10972	89034	831824	76944	4985	65	U	[1]
`Primary school`	242	16634	251546	3100	5607	135	U	[1]
`Facebook wall`	35817	104942	198028	194904	5751	89	D	[36]
`Slashdot reply`	51083	130370	139789	89862	8653	2916	D	[36]
`High school 2013`	327	11636	377016	7375	20642	88	U	[1]
`Topology`	16564	122140	198038	32823	22453	1401	U	[27]
`SMS`	44090	67190	544607	467838	25178	407	D	[36]
`Email EU`	986	24929	327336	207880	31840	346	D	[29]

Social networks: This domain includes most of the considered networks: `College msg`, `Wiki elections`, `Digg reply`, `Slashdot reply`, a subgraph of `Facebook wall` [50] containing the last $\sim 200k$ temporal arcs (as in the work of Santoro and Sarpe [39]), `SMS` and `Email EU`. These are social networks from different realms, where nodes correspond to users and temporal arcs indicate messages sent between them at specific points in time.

Contact networks: This domain includes the six networks `Hypertext 2009`, `High school 11/12/13`, `Hospital ward`, `Infectious`, `Primary school` and `Topology`. In the first case nodes correspond to individuals, while in the second case they correspond to computers. In both cases temporal arcs indicate a contact between nodes at a specific time.

In Appendix C we briefly discuss another type of temporal networks, that is, public transport networks (see, for example, [14, 15]), which, due to their topology, have quite peculiar properties in terms of both the execution times and of the quality of the analysed proxies.

Evaluation Details. We executed the experiments on a server running Ubuntu 20.04.5 LTS 112 with processors Intel(R) Xeon(R) Gold 6238R CPU @ 2.20GHz and 112GB RAM. All the correlation coefficients were computed by making use of the corresponding functions available in the Python `scipy.stats` module [13].

3.2 Experimental Results

Experiment 1: Global Proxies' Correlation to TempBrandes

In our first experiment, we run both TempBrandes and all the discussed global proxies on the networks listed in Table 1. We then, for each of these four algorithms (TempBrandes plus three proxies), compute the resulting node ranking and evaluate the correlation of the rankings computed by the proxies with the ranking computed by TempBrandes. Here, we employ two different rank correlation measures, i.e., (1) a weighted version of Kendall's τ coefficient based on the work of Vigna [49], and (2) the number of common highest rank nodes among the first k. We also investigated the Spearman's ρ coefficient [43] and Kendall's τ coefficient [25] of the rankings, but we omit these results here due to space constraints. We, however, note that these measures indicated similar proxy performance as (1), and at the same time we find (1) more relevant, as it gives more importance to approximating the upper part of the ranking. For the weighted Kendall's τ coeffficient, we use a hyperbolic weighting scheme, as proposed by Vigna [49], that gives weights to the positions in the ranking which decay harmonically with the ranks, i.e., the weight of rank r is $1/(r+1)$. We refrain from comparing the proxies with respect to average correlation due to outliers.

Table 2 For each network, we show the execution times of TempBrandes and of all proxies (except for Onbra) in seconds. Dashes indicate that the experiment was interrupted after the time of TempBrandes elapsed. We omit Onbra from the table as its running time is fixed to approximately 1/10, 1/2, or 1 times the running time of TempBrandes due to the choice of the sample size.

Network	Execution Time (seconds)					
	TempBrandes	Brandes	Prefix	EgoPrefix	EgoSTB	PTD
`Hypertext 2009`	262.58	**0.01**	2.29	25.14	–	**0.01**
`High school 2011`	445.62	0.02	3.39	15.81	–	**0.01**
`Hospital ward`	659.13	**0.01**	2.01	37.97	–	**0.01**
`College msg`	894.44	1.12	21.58	4.83	116.53	**0.02**
`Wiki elections`	1192.42	6.52	49.84	45.54	586.75	**0.06**
`High school 2012`	1345.06	0.03	7.77	232.90	–	**0.01**
`Digg reply`	1762.09	123.37	224.58	1.61	4.43	**0.05**
`Infectious`	4985.19	3.28	50.26	26.97	820.73	**0.11**
`Primary school`	5607.17	0.08	39.22	492.73	–	**0.04**
`Facebook wall`	5750.73	349.01	429.38	2.00	17.86	**0.07**
`Slashdot reply`	8652.54	442.75	1116.99	7.08	38.78	**0.07**
`High school 2013`	20641.71	0.11	95.49	200.89	–	**0.09**
`Topology`	22452.98	124.98	1017.78	905.69	–	**0.08**
`SMS`	25178.27	129.53	591.98	4.18	476.71	**0.09**
`Email EU`	31839.72	0.54	180.86	411.07	–	**0.05**

Running Times. The running times of the global proxies can be found in the first three columns of Table 2. We note that Prefix always terminates in at most 15% of the running time of TempBrandes, while Brandes always finishes in at most 7% of the running time of TempBrandes. The efficiency of both proxies is particularly pronounced on contact networks with lots of temporal edges and comparatively few edges in the underlying graph. As a result the underlying graph is comparatively small, which is beneficial for Brandes.

On the other hand, the number of prefix foremost shortest paths is also much smaller than the total number of shortest temporal paths, which is beneficial for PREFIX. The running times of the three ONBRA versions are fixed to approximately 1/10, 1/2, and 1 times the running times of TEMPBRANDES due to the choice of the sample size.

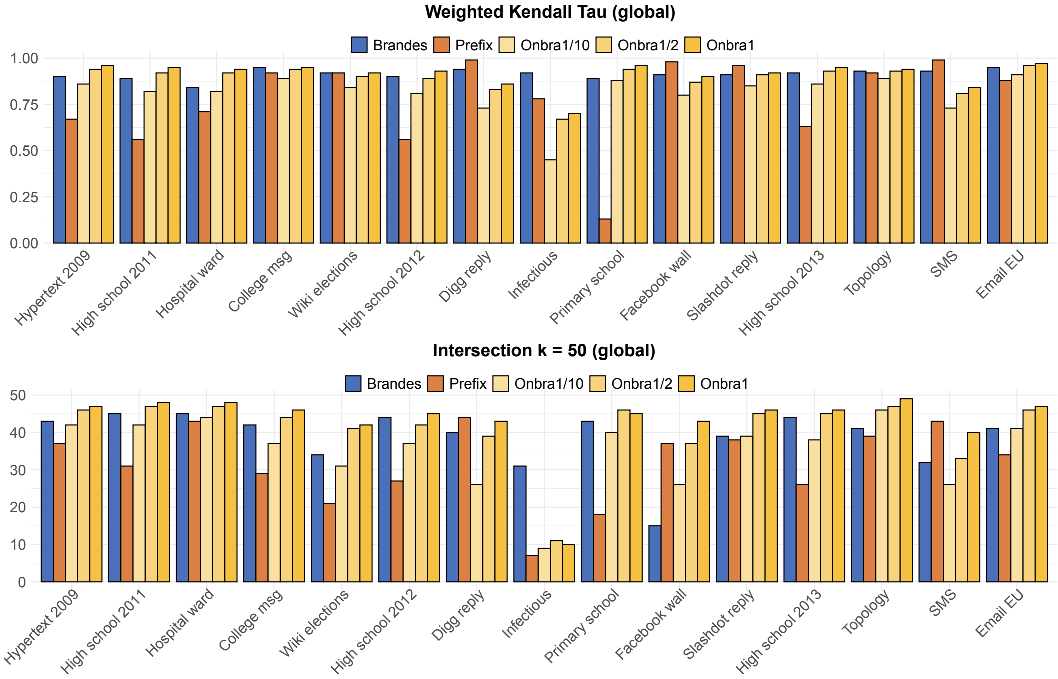

Figure 1 Comparison of the centrality ranking produced by TEMPBRANDES and the rankings produced by the global proxies. The comparison is given in terms of the weighted Kendall's τ coefficient and the intersection of the top 50 nodes.

Ranking Correlation. An illustration of the ranking correlation results of this experiment can be found in Figure 1. On top of the figure, we show the Weighted Kendall's τ correlation of the rankings computed by the respective proxies and the ranking computed by TEMPBRANDES. On the bottom, we show the results in terms of the intersection of the top-k nodes. We choose the value of k to be 50 here, while further results for $k = 1$ and $k = 25$ can be found in Table 5 in the appendix.

In terms of the weighted Kendall's τ correlation (see Table 3), we first observe that there are three (3) networks in which BRANDES performs best, five (5) networks in which PREFIX performs best, and ten (10) networks in which ONBRA with maximal sample size performs best (we count networks with ties multiple times). We, however, also notice that the ONBRA's performance heavily relies on the used sample size. Indeed, if the sample size is such that ONBRA needs roughly 10% of TEMPBRANDES running time, we observe that the numbers change as follows: there are eleven (11) networks in which BRANDES achieves the best correlation and there are five (5) networks in which PREFIX performs best, while ONBRA never performs best.

As BRANDES always terminates in less than 7% of TEMPBRANDES' running time, and in most cases much faster, we can conclude that the static betweenness rankings are actually quite competitive in situations where we are restricted in terms of running time. In other words, it seems really necessary to give ONBRA a running time similar to the exact algorithm

Table 3 For each network, we show the weighted Kendall's τ coefficient of the rankings computed by the three global proxies and the ranking computed by TempBrandes. For Onbra we show the results using, respectively, a sample size such that Onbra's execution time is $1/10, 1/2$, and exactly the one of TempBrandes. For each instance, we highlight the best result in bold font.

Network	weighted Kendall's τ coefficient				
	Brandes	Prefix	Onbra$_{\frac{1}{10}}$	Onbra$_{\frac{1}{2}}$	Onbra$_1$
`Hypertext 2009`	0.90	0.67	0.86	0.94	**0.96**
`High school 2011`	0.89	0.56	0.82	0.92	**0.95**
`Hospital ward`	0.84	0.71	0.82	0.92	**0.94**
`College msg`	**0.95**	0.92	0.89	0.94	**0.95**
`Wiki elections`	**0.92**	**0.92**	0.84	0.90	**0.92**
`High school 2012`	0.90	0.56	0.81	0.89	**0.93**
`Digg reply`	0.94	**0.99**	0.73	0.83	0.86
`Infectious`	**0.92**	0.78	0.45	0.67	0.70
`Primary school`	0.89	0.13	0.88	0.94	**0.96**
`Facebook wall`	0.91	**0.98**	0.80	0.87	0.90
`Slashdot reply`	0.91	**0.96**	0.85	0.91	0.92
`High school 2013`	0.92	0.63	0.86	0.93	**0.95**
`Topology`	0.93	0.92	0.89	0.93	**0.94**
`SMS`	0.93	**0.99**	0.73	0.81	0.84
`Email EU`	0.95	0.88	0.91	0.96	**0.97**

in order for it to outperform Brandes. At this point, we would like to emphasize that our choice of sample size for Onbra is inherently impractical as it requires to run the exact algorithm first. We chose this approach in order to be as fair as possible when evaluating its performance in terms of quality. Choosing its sample size based on the time of other proxies, as, e.g., Brandes, makes its performance much worse in comparison. The results based on the intersection measure are somewhat similar, with Onbra performing slightly better.

4 Pass-Through Degree

Motivated by the fact that the running times of the global proxies employed in the previous section all grow much faster than linearly, we now turn to local proxies, i.e., proxies which compute centrality values purely based on nodes' neighborhoods. In the case of static graphs, it is common practice to compare more involved centrality notions to the simple degree centrality. Motivated by this fact, we here introduce a new degree notion for temporal graphs, which we will evaluate as a local proxy for shortest temporal betweenness in what follows. This new degree notion is somewhat related to the ego-betweenness, but it is in fact even simpler. In the end of this section, we will show that it can be computed for all nodes in nearly linear time in the number of temporal arcs.

Static Pass-Through Degree. With the aim of a simpler exposition, we start by giving the definition of our new degree notion for static directed graphs. We first note that the two standard degree notions in directed graphs, the in-degree $\mathrm{d}^{\mathrm{in}}(u) = |N^{\mathrm{in}}(u)|$ and the out-degree $\mathrm{d}^{\mathrm{out}}(u) = |N^{\mathrm{out}}(u)|$, both fail to observe the vertex as a whole, by taking in-going and out-going arcs into account in isolation. In undirected graphs, on the other hand, the

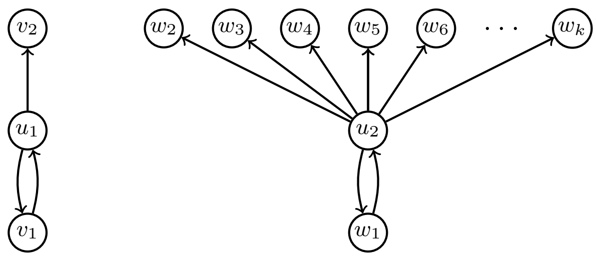

Figure 2 For the first variant, the pass-through degree of vertices u_1 and u_2 in the example graphs depicted above is equal. Namely, $d_1(u_1) = |N^{\text{in}}(u_1)| = 1 = |N^{\text{in}}(u_2)| = d_1(u_2)$. For the second variant this is not the case, as $d_2(u_1) = \sqrt{2}$ and $d_2(u_2) = \sqrt{k} = \Theta(\sqrt{n})$, where n denotes the number of nodes in the graph.

degree of a vertex also measures the number of neighbor pairs that can reach each other by passing through u, albeit normalized by the size of the neighborhood of u. In other words, $\mathrm{d}(u) = \frac{|N(u)|\cdot|N(u)|}{|N(u)|}$. This is, of course, just an overly complicated way of writing down the identity $\mathrm{d}(u) = |N(u)|$, but we use it as motivation for defining the analogous degree notion in directed graphs. We actually give two candidate definitions first, both generalizing the above equality to directed graphs, and then argue which of the two notions is more reasonable. The two variants of a directed degree notion that we propose, for a node $u \in V$, are $d_1(u) := \frac{|N^{\text{in}}(u)|\cdot|N^{\text{out}}(u)|}{|N^{\text{in}}(u)\cup N^{\text{out}}(u)|}$ and $d_2(u) := \sqrt{|N^{\text{in}}(u)| \cdot |N^{\text{out}}(u)|}$. When modeling an undirected graph $G = (V, E)$ as a directed graph $D = (V, A)$, by introducing two arcs (u, v) and (v, u) for every edge $\{u, v\} \in E$, we obtain, for every node u in the undirected graph, $N(u) = N^{\text{in}}(u) = N^{\text{out}}(v)$ and $d_1(u) = d_2(u) = d(u)$. Thus, both notions are proper generalizations of the undirected degree.

While at first sight it is not obvious which vertex degree definition is more suitable, both of them being legitimate generalizations of the undirected degree, one of the two turns out to be better suited for measuring vertex importance. As the examples in Figure 2 illustrate, the first candidate, d_1, has a serious drawback. More formally, when $N^{\text{in}}(u) \cup N^{\text{out}}(u) \in \{N^{\text{in}}(u), N^{\text{out}}(u)\}$, then $d_1(u) \in \{|N^{\text{in}}(u)|, |N^{\text{out}}(u)|\}$. This in particular means that in such a case, contrary to our initial intention, the degree of a node depends only on the in-going or the out-going arcs. Since the second candidate does not suffer from this issue, we find it more suitable for defining our directed degree notion. We now formally define it as the *pass-through degree* of a node.

▶ **Definition 1.** *In a static directed graph $G = (V, A)$, the* pass-through-degree *of $u \in V$ is defined as*

$$\mathrm{ptd}(u) := \sqrt{|N^{\text{in}}(u)| \cdot |N^{\text{out}}(u)|}$$

We point out that the pass-through degree is the geometric mean of in- and out-degree, the two classical notions of directed degree.

Temporal Pass-Through Degree. The introduced pass-through degree notion nicely generalizes to temporal graphs. Recall that the pass-through degree of a node u is equal to the geometric mean of the number of ordered neighbor pairs v, w that are connected through u. We generalize this to temporal nodes via pairs of neighbors that are temporally connected via exactly two hops through u. Formally, we write $v \xrightarrow{u} w$ if and only if there exist $a_1 = (v, u) \in A$ and $a_2 = (u, w) \in A$ such that $\lambda(a_1) < \lambda(a_2)$. We are now ready to define the temporal pass-through degree.

▶ **Definition 2.** *In a temporal graph* $\mathcal{G} = (V, A, \lambda)$*, the* temporal pass-through-degree *of* $u \in V$ *is*

$$\text{t-ptd}(u) := \sqrt{|\{(v, w) \in (V \setminus \{u\})^2 : v \xrightarrow{u} w\}|}$$

Algorithm 1 Temporal Pass-Through Degree.

Data: temporal arc list $\mathcal{A}$
Result: temporal pass-through degree of all vertices t-ptd
1 $\overline{G}, \underline{G} = \{\emptyset\}$ `// initialize two empty temporal graphs`
2 **for** ***each*** $(u, v, t) \in \mathcal{A}$ **do**
  `// check if the edge already exists in` $\overline{G}, \underline{G}$
3   **if** $(u, v) \in E(\overline{G})$ **then**
    `// update max and min encountered label`
4     $\overline{G}(u, v) = \max\left(\overline{G}(u, v), t\right)$, $\underline{G}(u, v) = \min\left(\underline{G}(u, v), t\right)$
5   **else**
6   add (u, v) to $E(\overline{G}), E(\underline{G})$
7     $\overline{G}(u, v) = t$, $\underline{G}(u, v) = t$
8 sort edges of $\underline{G}$ in ascending order according to time labels
9 $L_{eat} = [[\cdot], [\cdot], \ldots, [\cdot]]$ `// list of` n `empty lists`
10 **for** ***each*** $(u, v, \underline{t}) \in \underline{G}$ **do**
11   $L_{eat}[v].append(\underline{t})$
12 t-ptd $= [0, 0, \ldots, 0]$`// initialize array of` n `zeros`
13 **for** ***each*** $(v, w, \overline{t}) \in \overline{G}$ **do**
  `// compute the pass-through degrees`
14   t-ptd$[v]$ = t-ptd$[v] + \left|\{t \in L_{eat}[v] : t < \overline{t}\}\right|$
15 **return** t-ptd

Computation of the Temporal Pass-Through Degree. Algorithm 1, given the temporal arc list $\mathcal{A}$ of a temporal graph, computes the pass-through degrees in $O(M \log m) = \tilde{O}(M)$ time and $O(m+n)$ space, where M is the number of temporal arcs and m, n are, respectively, the number of arcs and the number of nodes of the underlying static graph. More precisely, the first **for** loop (lines 2-7) iterates over all the temporal arcs and builds two simple labeled directed graphs, $\overline{G}$ and $\underline{G}$, which respectively keep track of the maximum and the minimum appearance time of each arc from the underlying graph. Building $\overline{G}$ and $\underline{G}$ requires $O(M \log m)$ time, as we can maintain a vertex-sorted list of already added arcs, and $O(m+n)$ space. Subsequently, the algorithm sorts the m arcs of the temporal graph $\underline{G}$ according to their time labels in time $O(m \log m)$. The second **for** loop (lines 10-11) iterates over all the (now sorted) arcs of the temporal graph $\underline{G}$, and appends the appearance time $\underline{t}$ of the arc $(u, v, \underline{t})$ to the minimum incoming times list of node v. Since $\underline{G}$ has exactly m arcs, this loop requires $O(m)$ steps and uses $O(m + n)$ space. Finally, using $O(n)$ space, the last **for** loop (lines 13-14) iterates over all the m temporal arcs $(u, w, \overline{t})$ of $\overline{G}$ and increments the t-ptd variables. More specifically, when encountering the temporal arc $(v, w, \overline{t})$, it increases the previous t-ptd value of v by the number of distinct in-going temporal arcs of v in $\underline{G}$ with $t < \overline{t}$ (line 14). Since $L_{eat}[v]$ is a sorted lists of length at most n (as $\underline{G}$ is a simple graph), for each $v \in V$ we can compute the new $ptd[u]$ in $O(\log n)$ via binary-search. Therefore, the last loop requires $O(m \log n)$ steps. The overall time and space complexities are therefore $O(M \log m) = \tilde{O}(M)$ and $O(m + n)$, respectively.

5 Local Proxies for Shortest Temporal Betweenness

We now turn to an experimental analysis of local proxies for shortest temporal betweenness. Our approach here is the same as in Section 3 and, besides the different choice of proxies, our experimental setting is identical. We first list the set of local proxies for shortest temporal betweenness that our study includes.

EgoSTB: The algorithm for computing the ego-shortest temporal betweenness ego-stb of all nodes by going through them iteratively, computing the ego-network of the respective node, and then calling the algorithm of Buß et al. [12] for computing the shortest temporal betweenness of the node in its ego-network.

EgoPrefix: The algorithm that, analogously to the one above, computes the ego-prefix foremost temporal betweenness ego-pftb of all nodes.

PTD: The algorithm for computing the temporal pass-through degree of all nodes in nearly linear time in the number of temporal arcs, described in Section 4.

We in addition examine the rankings produced by both the static and temporal versions of the in- and out-degree. These results are omitted from the main body of the paper (but can be found in Table 7, Appendix B). The quality of the rankings returned by PTD is usually much better, and only in a single case (on `Infectious`) is the obtained Weighted Kendall's τ value more than 0.01 worse than for another degree notion.

Table 4 For each network, we show the weighted Kendall's τ coefficient of the rankings computed by the three local proxies and the ranking computed by TempBrandes. For Onbra we show the results using a sample size such that Onbra's execution time is 1/10 the one of TempBrandes. For each instance, we highlight the best result in bold font.

Network	weighted Kendall's τ coefficient			
	Onbra$_{\frac{1}{10}}$	EgoPrefix	EgoSTB	PTD
`Hypertext 2009`	0.86	0.73	–	**0.89**
`High school 2011`	**0.82**	0.69	–	0.76
`Hospital ward`	**0.82**	0.77	–	**0.82**
`College msg`	0.89	0.94	0.94	**0.95**
`Wiki elections`	0.84	**0.94**	**0.94**	**0.94**
`High school 2012`	**0.81**	**0.81**	–	**0.81**
`Digg reply`	0.73	**0.96**	**0.96**	**0.96**
`Infectious`	0.45	0.76	**0.81**	0.65
`Primary school`	**0.88**	0.63	–	0.83
`Facebook wall`	0.8	**0.94**	**0.94**	0.93
`Slashdot reply`	0.85	**0.97**	**0.97**	0.96
`High school 2013`	**0.86**	0.83	–	0.83
`Topology`	0.89	**0.92**	–	**0.92**
`SMS`	0.73	0.95	**0.96**	0.94
`Email EU`	**0.91**	**0.91**	–	**0.91**

Experiment 2: Local Proxies' Correlation to TempBrandes

Running Times. The local proxies' running times can be found in the last three columns of Table 2. We note that the running time of EgoSTB easily becomes prohibitively large: in fact, we interrupted its execution once the time of TempBrandes elapsed, which resulted in eight (8) missing values for EgoSTB. We note that this is due to the large size of the ego networks, which can be deduced from the $n_e^{\max}$ column in Table 1. We emphasize that the nearly linear time algorithm from the previous section computes the pass-through degree of all nodes in less than 0.005% of the running time of TempBrandes on all data sets.

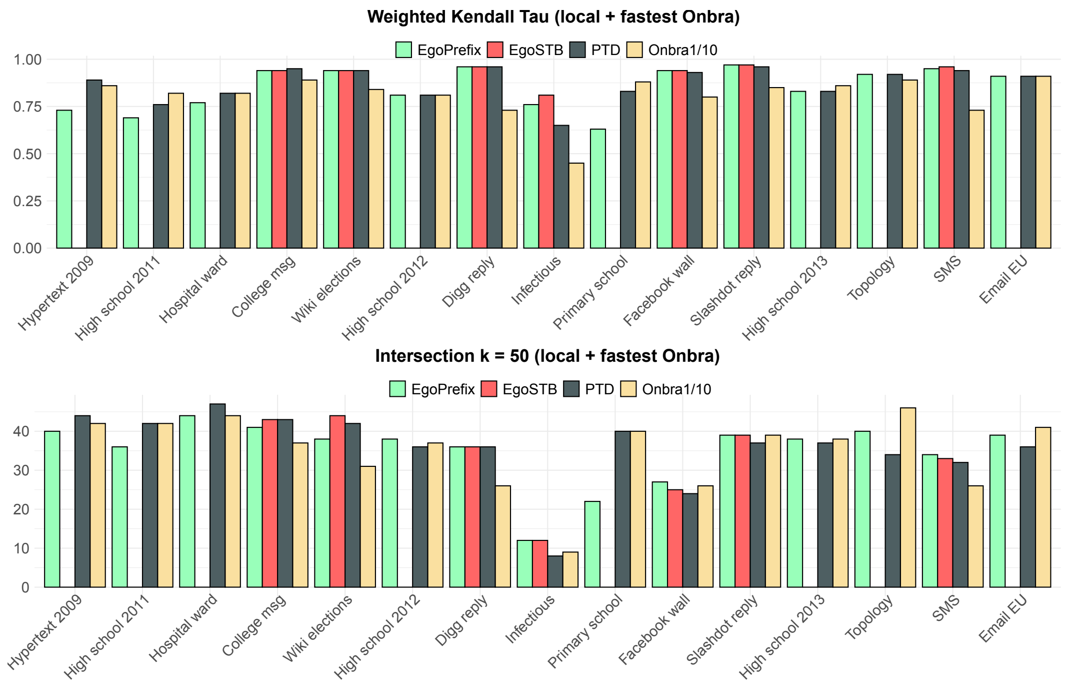

Figure 3 Comparison of the centrality ranking produced by TempBrandes and the rankings produced by the local proxies and Onbra with the smallest considered sample size. The comparison is given in terms of the weighted Kendall's τ coefficient and the intersection of the top 50 nodes.

Ranking Correlation. An illustration of the ranking correlation results of this experiment can be found in Figure 3. On top of the figure, we show the Weighted Kendall's τ correlation coefficient of the rankings computed by the respective proxies and the ranking computed by TempBrandes (see also Table 4). On the bottom, we show the results in terms of the intersection of the top-k nodes (again $k = 50$, see Table 6, Appendix A, for $k = 1$ and $k = 25$). In order to allow for better comparison with the results for global proxies from Section 3, in all the tables and plots that follow, we also include the fastest variant of Onbra, i.e, the variant with roughly 10% of TempBrandes' running time. We observe that the pass-through degree usually does not perform worse than the ego-variants of the shortest temporal betweenness and is at the same time much faster. In terms of both weighted Kendall's τ coefficient and the intersection measure, the pass-through degree performs better or at least as good as the considered version of Onbra on 10 out of 15 instances. At the same time its running time is between 3 and 4 orders of magnitudes smaller on all instances.

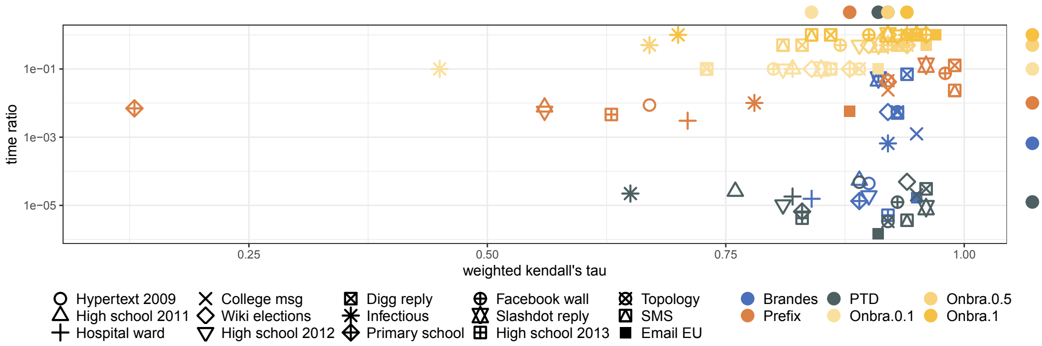

Figure 4 A two-dimensional illustration of ranking quality in terms of weighted Kendall's τ coefficient (on the horizontal linear axis) and the ratio between the proxies execution time and the time of TempBrandes (on the vertical logarithmic axis). The shapes of the points indicate the network, while the color indicates the proxy. On the top and on the right we plot the median value of the weighted Kendall's τ and the time ratio, respectively. We note that the running time ratios of the three Onbra variants are fixed to 1/10, 1/2, and 1, respectively.

6 Conclusion

We experimentally compared three global and three local proxies for shortest temporal betweenness rankings. One of these local proxies is a novel temporal degree notion, called the pass-through degree, which computes the number of neighbor pairs that are temporally connected by a two-hop path passing through the node at hand. Our experimental results are summarized in Figure 4, which depicts the performance of both global and local proxies discussed in previous sections (both in terms of running time and ranking quality). When applied to very large temporal networks, the pass-through degree clearly outperforms all the other competitors in terms of time performance. As indicated by the median time ratios that are depicted on the right of the plot, the pass-through degree achieves a time ratio that is around two (2), three (3), and four (4) orders of magnitude better than Brandes, Prefix, and the fastest considered Onbra variant, respectively. In terms of ranking quality, the medians of the two time-intense Onbra variants are best, followed by Brandes, PTD, Prefix, and the fastest Onbra variant.[4]

One future direction is explaining the correlations between PTD and the shortest temporal betweenness by using temporal graph parameters, such as the ones defined in the works of Tang et al. [45] and Nicosia et al. [33]. It would also be interesting to use PTD as a proxy for both static and temporal centralities in the context of routing schemes [31], as its local character enables an efficient distributed computation. From a theoretical point of view, possible directions of research include finding a conditional lower bound on the time complexity of computing shortest temporal betweenness that is better than the lower bound implied by its non-temporal counterpart. Proving a conditional lower bound on the computation of the ego-network betweenness measures (or designing a better algorithm) is also a very challenging question. Finally, the pass-through degree easily generalises to k-hop paths (instead of 2-hop paths). We believe that designing a quasi-linear time algorithm for computing such a generalisation, and verifying its quality in terms of proxying the shortest temporal betweenness, is the most natural continuation of this work.

[4] We note that we chose to compute medians rather than averages here, as the data seems to include several outliers (see, e.g., Prefix on the primary school network).

References

1 Sociopatterns. `https://www.sociopatterns.org/`, last checked on February 10, 2023.

2 Amir Abboud, Fabrizio Grandoni, and Virginia Vassilevska Williams. Subcubic equivalences between graph centrality problems, APSP and diameter. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1681–1697. SIAM, 2015. `doi:10.1137/1.9781611973730.112`.

3 Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 522–539. SIAM, 2021. `doi:10.1137/1.9781611976465.32`.

4 Alex Bavelas. Communication patterns in task-oriented groups. *The journal of the acoustical society of America*, 22(6):725–730, 1950.

5 Ferenc Béres, Róbert Pálovics, Anna Oláh, and András A Benczúr. Temporal walk based centrality metric for graph streams. *Applied network science*, 2018.

6 Paolo Boldi and Sebastiano Vigna. Axioms for centrality. *Internet Mathematics*, 10(3-4):222–262, 2014.

7 Michele Borassi, Pierluigi Crescenzi, and Michel Habib. Into the square: On the complexity of some quadratic-time solvable problems. *Electron. Notes Theor. Comput. Sci.*, 322:51–67, 2016. `doi:10.1016/j.entcs.2016.03.005`.

8 Michele Borassi and Emanuele Natale. KADABRA is an adaptive algorithm for betweenness via random approximation. *ACM J. Exp. Algorithmics*, 24(1):1.2:1–1.2:35, 2019. `doi:10.1145/3284359`.

9 Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 2001.

10 Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

11 Laura F. Bringmann, Timon Elmer, Sacha Epskamp, Robert W. Krause, David Schoch, Marieke Wichers, Johanna Wigman, and Evelien Snippe. What do centrality measures measure in psychological networks? *Journal of Abnormal Psychology*, 128(8):892, 2019.

12 Sebastian Buß, Hendrik Molter, Rolf Niedermeier, and Maciej Rymar. Algorithmic aspects of temporal betweenness. In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 2084–2092. ACM, 2020. `doi:10.1145/3394486.3403259`.

13 The SciPy community. Statistical functions. `https://docs.scipy.org/doc/scipy/reference/stats.html`, last checked on February 10, 2023.

14 Pierluigi Crescenzi, Clémence Magnien, and Andrea Marino. Approximating the temporal neighbourhood function of large temporal graphs. *Algorithms*, 12(10), 2019.

15 Pierluigi Crescenzi, Clémence Magnien, and Andrea Marino. Finding top-k nodes for temporal closeness in large temporal graphs. *Algorithms*, 13(9), 2020.

16 Martin G. Everett and Stephen P. Borgatti. Ego network betweenness. *Soc. Networks*, 27(1):31–38, 2005. `doi:10.1016/j.socnet.2004.11.007`.

17 Robert Faris and Diane Felmlee. Status struggles: Network centrality and gender segregation in same-and cross-gender aggression. *American Sociological Review*, 76(1):48–73, 2011.

18 Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, March 1977. `doi:10.2307/3033543`.

19 Marwan Ghanem, Florent Coriat, and Lionel Tabourier. Ego-betweenness centrality in link streams. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, Sydney, Australia, July 31 - August 03, 2017*. ACM, 2017.

20 R. Goerke. Email network of KIT informatics. `https://i11www.iti.kit.edu/en/projects/spp1307/emaildata`, 2011. Online; accessed 10 February 2023.

21 Peter Grindrod, Mark C Parsons, Desmond J Higham, and Ernesto Estrada. Communicability across evolving networks. *Physical Review E*, 2011.

22 Shahrzad Haddadan, Cristina Menghini, Matteo Riondato, and Eli Upfal. Repbublik: Reducing polarized bubble radius with link insertions. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*. ACM, 2021.

23 Petter Holme. Modern temporal network theory: a colloquium. *The European Physical Journal B*, 2015.

24 Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

25 Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

26 R. Kujala, C. Weckström, R. Darst, M. Madlenocić, and J. Saramäki. A collection of public transport network data sets for 25 cities. *Sci. Data*, 5:article number: 180089, 2018.

27 J. Kunegis. The KONECT Project. `http://konect.cc`, last checked on February 10, 2023.

28 Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. Stream graphs and link streams for the modeling of interactions over time. *Soc. Netw. Anal. Min.*, 2018.

29 Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, last checked on February 10, 2023.

30 Carlos Lozares, Pedro López-Roldán, Mireia Bolibar, and Dafne Muntanyola. The structure of global centrality measures. *International Journal of Social Research Methodology*, 18(2):209–226, 2015.

31 Leonardo Maccari, Lorenzo Ghiro, Alessio Guerrieri, Alberto Montresor, and Renato Lo Cigno. On the distributed computation of load centrality and its application to DV routing. In *2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, HI, USA, April 16-19, 2018*, pages 2582–2590. IEEE, 2018. `doi:10.1109/INFOCOM.2018.8486345`.

32 Massimo Marchiori and Vito Latora. Harmony in the small-world. *Physica A: Statistical Mechanics and its Applications*, 285(3-4):539–546, 2000.

33 Vincenzo Nicosia, John Tang, Cecilia Mascolo, Mirco Musolesi, Giovanni Russo, and Vito Latora. Graph metrics for temporal networks. In *Temporal networks*. Springer, 2013.

34 Lutz Oettershagen and Petra Mutzel. Efficient top-k temporal closeness calculation in temporal networks. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2020.

35 Lutz Oettershagen, Petra Mutzel, and Nils M. Kriege. Temporal walk centrality: Ranking nodes in evolving networks. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. ACM, 2022.

36 Ryan A. Rossi and Nesreen K. Ahmed. Network repository. `https://networkrepository.com`, last checked on February 10, 2023.

37 Polina Rozenshtein and Aristides Gionis. Temporal pagerank. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016.

38 Maciej Rymar, Hendrik Molter, André Nichterlein, and Rolf Niedermeier. Towards classifying the polynomial-time solvability of temporal betweenness centrality. In *Graph-Theoretic Concepts in Computer Science - 47th International Workshop, WG 2021, Warsaw, Poland, June 23-25, 2021, Revised Selected Papers*, Lecture Notes in Computer Science. Springer, 2021.

39 Diego Santoro and Ilie Sarpe. ONBRA: rigorous estimation of the temporal betweenness centrality in temporal networks. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. ACM, 2022.

40 Nicola Santoro, Walter Quattrociocchi, Paola Flocchini, Arnaud Casteigts, and Frédéric Amblard. Time-varying graphs and social network analysis: Temporal indicators and metrics. *CoRR*, 2011.

41 John R Seeley. The net of reciprocal influence. a problem in treating sociometric data. *Canadian Journal of Experimental Psychology*, 3:234, 1949.

42 Frédéric Simard, Clémence Magnien, and Matthieu Latapy. Computing betweenness centrality in link streams. *CoRR*, 2021.

43 Charles Spearman. The proof and measurement of association between two things. *American Journal of Psychology*, 15:72–101, 1904.

44 John Kit Tang, Cecilia Mascolo, Mirco Musolesi, and Vito Latora. Exploiting temporal complex network metrics in mobile malware containment. In *12th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WOWMOM 2011, Lucca, Italy, 20-24 June, 2011*. IEEE Computer Society, 2011.

45 John Kit Tang, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Temporal distance metrics for social network analysis. In *Proceedings of the 2nd ACM Workshop on Online Social Networks, WOSN 2009, Barcelona, Spain, August 17, 2009*. ACM, 2009.

46 John Kit Tang, Mirco Musolesi, Cecilia Mascolo, Vito Latora, and Vincenzo Nicosia. Analysing information flows and key mediators through temporal centrality metrics. In *Proceedings of the 3rd Workshop on Social Network Systems, Paris, France, April 13, 2010*. ACM, 2010.

47 Shang-Hua Teng. Scalable algorithms for data and network analysis. *Found. Trends Theor. Comput. Sci.*, 12(1-2):1–274, 2016. doi:10.1561/0400000051.

48 Ioanna Tsalouchidou, Ricardo Baeza-Yates, Francesco Bonchi, Kewen Liao, and Timos Sellis. Temporal betweenness centrality in dynamic graphs. *Int. J. Data Sci. Anal.*, 2020.

49 Sebastiano Vigna. A weighted correlation index for rankings with ties. In Aldo Gangemi, Stefano Leonardi, and Alessandro Panconesi, editors, *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pages 1166–1176. ACM, 2015. doi:10.1145/2736277.2741088.

50 Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*, 2009.

A Top-k intersection value tables

Table 5 For each network, we show the intersections between the top 1, 25 and 50 nodes in the rankings computed by the three global proxies and the ranking computed by TempBrandes. For Onbra we show the results when its running time is, respectively, one tenth, half and exactly TempBrandes' execution time.

Network	Intersection														
	Brandes			Prefix			Onbra$_{\frac{1}{10}}$			Onbra$_{\frac{1}{2}}$			Onbra$_1$		
k	1	25	50	1	25	50	1	25	50	1	25	50	1	25	50
`Hypertext 2009`	1	22	43	1	15	37	1	20	42	1	22	46	1	22	47
`High school 2011`	0	20	45	0	14	31	0	19	42	0	20	47	1	22	48
`Hospital ward`	0	22	45	1	18	43	0	20	44	0	22	47	0	24	48
`College msg`	1	22	42	1	12	29	1	19	37	0	23	44	1	23	46
`Wiki elections`	1	16	34	0	7	21	1	15	31	1	21	41	1	22	42
`High school 2012`	1	21	44	0	8	27	1	18	37	1	20	42	1	22	45
`Digg reply`	1	20	40	1	22	44	0	14	26	0	20	39	1	20	43
`Infectious`	1	19	31	0	3	7	0	6	9	0	5	11	0	6	10
`Primary school`	0	18	43	0	3	18	1	20	40	1	23	46	1	23	45
`Facebook wall`	0	10	15	1	17	37	1	15	26	1	17	37	1	19	43
`Slashdot reply`	1	18	39	1	20	38	1	19	39	1	22	45	1	23	46
`High school 2013`	1	20	44	0	11	26	1	20	38	0	22	45	1	23	46
`Topology`	1	21	41	1	20	39	1	23	46	1	23	47	1	24	49
`SMS`	0	13	32	1	20	43	0	15	26	0	18	33	1	19	40
`Email EU`	1	20	41	0	14	34	1	19	41	1	23	46	1	23	47

Table 6 For each network, we show the intersections between the top 1, 25 and 50 nodes in the rankings computed by the three global proxies and the ranking computed by TempBrandes. For Onbra we show the results when its running time is one tenth of TempBrandes execution time.

Network		Intersection											
		$\text{ONBRA}_{\frac{1}{10}}$			EgoPrefix			EgoSTB			PTD		
	k	1	25	50	1	25	50	1	25	50	1	25	50
Hypertext 2009		1	20	42	1	17	40	–	–	–	1	20	44
High school 2011		0	19	42	0	16	36	–	–	–	1	17	42
Hospital ward		0	20	44	1	18	44	–	–	–	0	21	47
College msg		1	19	37	0	20	41	0	22	43	1	23	43
Wiki elections		1	15	31	1	19	38	1	21	44	1	19	42
High school 2012		1	18	37	1	16	38	–	–	–	1	13	36
Digg reply		0	14	26	1	19	36	1	19	36	1	19	36
Infectious		0	6	9	0	4	12	0	4	12	0	2	8
Primary school		1	20	40	0	11	22	–	–	–	0	17	40
Facebook wall		1	15	26	1	14	27	1	14	25	1	13	24
Slashdot reply		1	19	39	1	20	39	1	20	39	1	19	37
High school 2013		1	20	38	0	18	38	–	–	–	0	17	37
Topology		1	23	46	1	21	40	–	–	–	0	16	34
SMS		0	15	26	0	13	34	0	13	33	0	12	32
Email EU		1	19	41	0	18	39	–	–	–	1	18	36

B Comparison among Degree Notions

Table 7 For each network, we show the weighted Kendall's τ coefficient of the rankings computed by the static/temporal degree notions and the pass-through degree and the ranking computed by TempBrandes.

Network	weighted Kendall's τ coefficient				
	PTD	in-degree	out-degree	t-in-degree	t-out-degree
Hypertext 2009	**0.89**	**0.89**	**0.89**	0.72	0.72
High school 2011	0.76	**0.77**	**0.77**	0.40	0.40
Hospital ward	0.82	0.83	0.83	**0.85**	**0.85**
College msg	**0.95**	0.91	0.92	0.90	0.91
Wiki el's	**0.94**	0.74	0.72	0.72	0.72
High school 2012	0.81	**0.82**	**0.82**	0.50	0.50
Digg reply	**0.96**	0.84	0.83	0.84	0.83
Infectious	0.65	**0.70**	**0.70**	0.42	0.42
Faceb'k w'l	**0.93**	0.86	0.89	0.85	0.88
Primary school	0.83	**0.84**	**0.84**	0.70	0.70
Slashdot reply	**0.96**	0.78	0.94	0.80	0.94
SMS	**0.94**	0.84	0.88	0.69	0.81
High school 2013	0.83	**0.84**	**0.84**	0.50	0.50
Topology	**0.92**	**0.92**	**0.92**	**0.92**	**0.92**
Email EU	**0.91**	0.87	0.90	0.77	0.83

C Public transport networks

A special class of temporal networks are public transport networks, in which the existence of a temporal arc (u, v, t) indicates that it is possible to reach location v from u by taking a mean of public transport at time t. Indeed, these networks are characterized by a sort of “regularity” (that is, nodes are very similar in terms of in- and out-degree), which makes the local proxies quite bad in proxying the temporal shortest betweenness. At the same time, they contain a huge amount of shortest temporal paths, which forced us to use big integer data structures for TempBrandes and Onbra, thus significantly slowing down their execution time. As an example, we considered the two networks `Venice` and `Bordeaux` that stem from the work of Kujala et al. [26], and are chosen to be analysed because of their different sizes and geographies. The main characteristics of these two networks are summarised in the following table.

Data set	n	m	M	T	t_{STB}	n_e^{max}	Type	Source
`Venice`	1874	3465	113670	1691	7758	20	D	[26]
`Bordeaux`	3435	4040	236075	60582	50937	13	D	[26]

The execution times TempBrandes and of all proxies (except for Onbra) in seconds are shown in the following table (once again dashes indicate that the experiment was interrupted after the time of TempBrandes elapsed and we omit Onbra from the table as its running time is fixed to approximately 1/10, 1/2, or 1 times the running time of TempBrandes due to the choice of the sample size).

Network	Execution Time (seconds)					
	TempBrandes	Brandes	Prefix	EgoPrefix	EgoSTB	PTD
`Venice`	7758	0.7374	72.9	31	48	0.0168
`Bordeaux`	50937	2.8722	443.0	93	107	0.0161

The weighted Kendall’s τ coefficient of the rankings computed by the three global proxies and the ranking computed by TempBrandes are shown in the following table (once again, for Onbra we show the results using, respectively, a sample size such that Onbra’s execution time is $1/10, 1/2$, and exactly the one of TempBrandes, and, for each instance, we highlight the best result in bold font).

Network	weighted Kendall’s τ coefficient				
	Brandes	Prefix	Onbra$_{\frac{1}{10}}$	Onbra$_{\frac{1}{2}}$	Onbra$_1$
`Venice`	0.90	0.80	0.93	0.96	**0.98**
`Bordeaux`	0.96	0.82	0.94	0.97	**0.98**

We can observe that, in this case, Onbra even in the case of the smallest sample size is better than Brandes and Prefix. However, it is also worth observing that Brandes performs quite well in the case of both networks, suggesting that, in these cases, the temporality of the network does not influence so much the ranking of the nodes. This might be intuitively justified by the fact that a “temporally” central node in this kind of networks is also central in the underlying graphs. This is confirmed by the following table, which shows the intersection values for all global proxies for values of $k = 1, 25, 50$ (once again, a value of x in the table means that the top-k nodes with respect to the ranking computed by a given proxy contain x of the top-k nodes of the ranking computed by TempBrandes).

Network	Intersection														
	Brandes			Prefix			Onbra$_{\frac{1}{10}}$			Onbra$_{\frac{1}{2}}$			Onbra$_1$		
k	1	25	50	1	25	50	1	25	50	1	25	50	1	25	50
`Venice`	0	16	36	0	12	23	1	23	41	0	24	47	0	24	48
`Bordeaux`	1	23	46	1	17	29	1	19	48	1	23	48	1	24	49

The next table shows the weighted Kendall's τ coefficient of the rankings computed by the three local proxies and the ranking computed by TempBrandes (once again, for Onbra we show the results using a sample size such that Onbra's execution time is 1/10 the one of TempBrandes, and, for each instance, we highlight the best result in bold font).

Network	weighted Kendall's τ coefficient			
	Onbra$_{\frac{1}{10}}$	EgoPrefix	EgoSTB	PTD
`Venice`	**0.93**	0.64	0.62	0.61
`Bordeaux`	**0.94**	0.63	0.55	0.61

As expected, the local proxies perform quite bad and Onbra is by far better than all of them. This is confirmed by the following table, which shows the intersections between the top 1, 25 and 50 nodes in the rankings computed by the three global proxies and the ranking computed by TempBrandes (once again, for Onbra we show the results when its running time is one tenth of TempBrandes execution time).

Network	Intersection											
	ONBRA$_{\frac{1}{10}}$			EgoPrefix			EgoSTB			PTD		
k	1	25	50	1	25	50	1	25	50	1	25	50
`Venice`	1	23	41	0	5	13	0	4	13	0	5	11
`Bordeaux`	1	19	48	0	4	5	0	4	5	0	4	5