

How Do Centrality Measures Choose the Root of Trees?

Cristian Riveros

Pontificia Universidad Católica de Chile, Santiago, Chile

Millennium Institute for Foundational Research on Data, Santiago, Chile

Jorge Salas

Pontificia Universidad Católica de Chile, Santiago, Chile

Millennium Institute for Foundational Research on Data, Santiago, Chile

University of Edinburgh, UK

Oskar Skibski

University of Warsaw, Poland

Abstract

Centrality measures are widely used to assign importance to graph-structured data. Recently, understanding the principles of such measures has attracted a lot of attention. Given that measures are diverse, this research has usually focused on classes of centrality measures. In this work, we provide a different approach by focusing on classes of graphs instead of classes of measures to understand the underlying principles among various measures. More precisely, we study the class of trees. We observe that even in the case of trees, there is no consensus on which node should be selected as the most central. To analyze the behavior of centrality measures on trees, we introduce a property of *tree rooting* that states a measure selects one or two adjacent nodes as the most important, and the importance decreases from them in all directions. This property is satisfied by closeness centrality but violated by PageRank. We show that, for several centrality measures that root trees, the comparison of adjacent nodes can be inferred by *potential functions* that assess the quality of trees. We use these functions to give fundamental insights on rooting and derive a characterization explaining why some measure root trees. Moreover, we provide an almost linear time algorithm to compute the root of a graph by using potential functions. Finally, using a family of potential functions, we show that many ways of tree rooting exist with desirable properties.

2012 ACM Subject Classification Information systems → Data management systems; Computing methodologies → Network science; Networks → Network structure; Theory of computation → Data structures and algorithms for data management

Keywords and phrases Databases, centrality measures, data centrality, graph theory, tree structures

Digital Object Identifier 10.4230/LIPIcs.ICDT.2023.12

Related Version *Full Version:* <https://arxiv.org/abs/2112.13736>

Funding C. Riveros and J. Salas were supported by ANID - Millennium Science Initiative Program - Code ICN17_002. Jorge Salas and Oskar Skibski were supported by the Polish National Science Centre Grant No. 2018/31/B/ST6/03201.

1 Introduction

Centrality measures are fundamental tools for network analysis. They are used in a plethora of applications from various areas of science, such as finding people who are more likely to spread a disease in the event of an epidemic [10] or highlighting cancer genes in proteomic data [15]. In all these applications, people use centrality measures to assess graph data and rank which nodes are the most relevant by only using the graph's structure.



© Cristian Riveros, Jorge Salas, and Oskar Skibski;
licensed under Creative Commons License CC-BY 4.0
26th International Conference on Database Theory (ICDT 2023).

Editors: Floris Geerts and Brecht Vandevoort; Article No. 12; pp. 12:1–12:17

Leibniz International Proceedings in Informatics



LIPIC Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

12:2 How Do Centrality Measures Choose the Root of Trees?

Database systems have also incorporated centrality measures to rank nodes in graph data. The first significant use of a centrality measure in a data management problem is by the Google search engine, which successfully introduced PageRank [22] for assessing the importance of a website on the web. Indeed, today graph databases, like Neo4j [1] or TigerGraph [2], natively support centrality algorithms for ranking nodes or analyzing the graph structure. Furthermore, new applications for centrality measures have emerged over knowledge graphs for entity linking [19] or semantic web search engines where ranking results is a core task [14]. More generally, centrality measures play a central role in *network science*, where they are one of the main algorithmic metrics for analyzing graphs [20].

Although more than 200 measures have been proposed in the literature today, it is less clear how one can compare them. Precisely, these measures assess the position of a vertex in the network based on various elements (e.g., degree, number of paths, eigenvector), which makes it hard to compare their properties. Since people propose dozens of new measures every year, choosing a measure for a specific application becomes more and more challenging. In recent years, efforts at understanding and explaining differences between existing measures have intensified, making the foundational aspects of centrality measures an active research topic [5, 26, 24, 27]. Given the diversity of these measures, people have usually followed an axiomatic approach by considering classes of measures like game-theoretical [27] or motif-based [24] measures. Until now, no research has focused on graph classes to classify different centrality measures.

This paper aims to understand how centrality measures operate on trees (i.e., acyclic undirected graphs), as this is a crucial graph family where we can compare different centrality measures and understand their underlying principles. Indeed, even in the case of trees, centrality measures vary significantly as different measures could indicate other vertices as the most central in the same tree. For example, consider a line graph: while the middle vertex (or vertices) is ranked first according to most centrality measures, Google's PageRank puts the second and the second-to-last vertices at the top of the ranking. This simple example shows that not only do different measures select different vertices, but even one measure may select several vertices from different parts of the tree as the most important. Moreover, although PageRank diverges, other centrality measures (e.g., closeness or all-subgraph) coincide following the same underlying principles.

A natural question here is why we should start considering centrality measures over the class of trees. We see several reasons for studying the principal aspects of centrality measures over trees. First, undirected trees are arguably the simplest and non-trivial graph class to study centrality measures. Indeed, trees have a more amenable structure than general graphs, given that, among other properties, every edge is a cutting edge, and there is a unique path between any two nodes. Second, every general result on centrality measures should include trees, and it should answer similar questions to the one studied in this paper. Thus, understanding centrality measures over trees could guide the study of other graph families. Last, trees are a graph structure ubiquitous in computer science and databases. Therefore, studying centrality measures over trees could be helpful for the application of centrality measures in data management and other areas (see Section 8 for some extensions and possible applications of centrality measures over trees).

What principal aspect of centrality measures can we study over the class of trees? In this paper, we focus on understanding one of the main questions over trees: how centrality measures choose the most central nodes in trees, why some measures define a single important node (usually called the root), and why others do not. For answering these questions, the main contributions are as follows:

1. We introduce the *tree rooting property* which states that a measure selects one or two adjacent vertices in a tree as the most important, and the importance decreases from them in all directions. We found that closeness, eccentricity, and all-subgraphs centralities (see Section 2) satisfy this rooting property but often rank different vertices at the top. Instead, measures like degree and betweenness do not satisfy this property. We call the vertex (or two vertices) ranked first the “root” and say that such measures *root trees*.
2. To understand what distinguishes measures that root trees we focus on the question: how to choose which out of two adjacent vertices is more central? We observe that most centrality measures, including all standard ones that root trees, answer this question by comparing subtrees of both vertices. More precisely, we introduce a framework of *potential functions* that assess the quality or “potential” of an arbitrary tree. Now, we show that most centrality measures admit a potential function such that the vertex which subtree has higher potential value is considered more central.
3. We show that if a centrality measure has a potential function, then it roots trees if, and only if, the potential function is *symmetric*. This property means that the potential of a tree is larger than the potential of any proper complete subtree. In particular, the potential function of closeness, eccentricity, and all-subgraphs centrality are symmetric, but the potential functions of degree and betweenness centralities are not; as a result, the latter centralities do not root trees.
4. We use our framework of potential functions to understand better the class of measures that root trees. More specifically, we show three applications of potential functions. Our first application is to study efficient algorithms for computing a root when centrality measures have potential functions and root trees. By exploiting symmetric potential functions, we show that, given a tree T , we can compute the most central vertex in time $\mathcal{O}(|T| \log(|T|))$ whenever one can calculate the potential function locally. Interestingly, this general algorithm works independently of the centrality measures and only depends on the potential function.
5. Our second application of potential functions is to understand desirable properties over tree rooting measures. Although a centrality measure could root trees, it can behave inconsistently. For instance, a rooting centrality measure could choose the root of a tree T close to a leaf when the size of T is even and far from the leaves when the size of T is odd. Therefore, we study when centrality measures consistently root trees through their potential functions. We propose a monotonicity property that imposes additional consistency conditions on how the root is selected and characterizes which potential functions satisfy this property.
6. Our last application shows how to design and build new potential functions that consistently root trees. Specifically, we present infinitely many constructive potential functions that satisfy all properties discussed so far. In particular, the algorithm for finding the root applies to any of them. We believe that this family of measures is interesting in its own right and can be used in several data-driven scenarios.

Related work. Our work could be included into a broad literature that focuses on the analysis of theoretical properties of centrality measures. The classic approach is to analyze standard centrality measures with respect to simple desirable properties. Different properties have been considered over the years [25, 21, 6, 5]. Most of them, however, are very simple (e.g., invariance under automorphism) or not satisfied by most measures. Similarly, in our work, we propose several properties specific for trees. Focusing on trees allows us to identify meaningful properties shared by many measures based on completely different principles.

Another approach is to create a common framework for large classes of centrality measures. In such frameworks, centrality measures are presented as a function of some underlying structure of the graph. Hence, the emphasis is focused on the differences between these functions and their implications for various measures defined under such framework. In this spirit, classes of measures based on distances [12], nodal statistics [4], coalitional game theory [27], subgraphs [24] and vitality functions [26] have been analyzed in the literature. On the opposite, our framework of potential functions can be considered as an approach focused on classes of graphs instead of classes of measures. Yet another approach, less related to our work, is to focus on one or several similar measures and provide their full axiomatic characterization. In this spirit, axiomatizations of PageRank [29], eigenvector [17], beta-measure and degree [28] and many more have been developed in the literature.

There is also a line of research that studies methods for solely choosing one most central vertex in a tree that does not necessarily come from the centrality analysis (see, e.g., [23] for an overview). However, such methods coincide with the top vertices selected by centrality measures that we consider in our work. In particular, the center of a tree coincides with eccentricity, and the median, as well as the centroid, coincides with closeness centrality.

2 Preliminaries

Undirected graphs. In this paper, we consider *finite undirected graphs* of the form $G = (V, E)$ where V is a finite non-empty set and $E \subseteq 2^V$ such that $|e| = 2$ for all $e \in E$. For convenience, given a graph $G = (V, E)$ we use $V(G)$ for indicating the set of *vertices* V and $E(G)$ for the set of *edges* E . We write $N_G(v)$ for the *neighbourhood* of v in G , namely, $N_G(v) \subseteq V(G)$ such that $u \in N_G(v)$ if, and only if, $\{u, v\} \in E(G)$. If this is the case, we say that u and v are *adjacent*.

We say that a graph $G' = (V', E')$ is a *subgraph* of G , denoted $G' \subseteq G$, if $V' \subseteq V$ and $E' \subseteq E$. Note that \subseteq forms a partial order between graphs. For a sequence of graphs $G_1 = (V_1, E_1), \dots, G_m = (V_m, E_m)$ we denote by $\cup_{i=1}^m G_i$ the graph (V, E) such that $V = \cup_{i=1}^m V_i$ and $E = \cup_{i=1}^m E_i$. Given a graph G and an edge $e = \{u, v\}$, we write $G + e$ to be the new graph G with the additional edge e , formally, $V(G + e) = V(G) \cup e$ and $E(G + e) = E(G) \cup \{e\}$.

From now on, fix an enumerable set \mathcal{V} of vertices. We define the *set of all graphs* using vertices from \mathcal{V} as \mathcal{G} . Further, we define the set \mathcal{VG} as all pairs *vertex-graph* (v, G) such that $v \in V(G)$ and $G \in \mathcal{G}$. In the sequel, for a pair (v, G) we assume that $(v, G) \in \mathcal{VG}$, unless stated otherwise. We say that graphs G_1 and G_2 are *isomorphic*, denoted by $G_1 \cong G_2$, if there exists a bijective function (*isomorphism*) $f : V(G_1) \rightarrow V(G_2)$ such that $\{u, v\} \in E(G_1)$ if, and only if, $\{f(u), f(v)\} \in E(G_2)$. We also say that (v_1, G_1) and (v_2, G_2) are isomorphic, denoted by $(v_1, G_1) \cong (v_2, G_2)$, if there exists an isomorphism f between G_1 and G_2 and $f(v_1) = v_2$. Note that \cong is an equivalence relation over \mathcal{G} and over \mathcal{VG} .

A *path* in G is a sequence of vertices $\pi = v_0, \dots, v_n$ such that $\{v_i, v_{i+1}\} \in E$ for every $i < n$, and we say that π is a *path* from v_0 to v_n . We say that π is *simple* if $v_i \neq v_j$ for every $0 \leq i < j < n$. From now on, we usually assume that paths are simple unless stated otherwise. We define the *length* of π as $|\pi| = n$. We agree that v_0 is the *trivial path* of length 0 from v_0 to itself. Given $R, R' \subseteq V(G)$, we say that $\pi = v_0, \dots, v_n$ is a path from R to R' if $v_0 \in R$, $v_n \in R'$, and $v_i \notin R \cup R'$ for every $i \in [1, n - 1]$. We say that a graph G is *connected* if there exists a path between every pair of vertices.

Centrality measures. A *centrality measure*, or just a *measure*, is any function $C : \mathcal{VG} \rightarrow \mathbb{R}$ that assigns a score $C(v, G)$ to v depending on its graph G . Here, we use the standard assumption that, the higher the score $C(v, G)$, the more important or “central” is v in G . We also assume that every centrality measure is *closed under isomorphism*, namely, if $(v_1, G_1) \cong (v_2, G_2)$ then $C(v_1, G_1) = C(v_2, G_2)$, which is a standard assumption in the literature [8, 25].

Next, we recall four centrality measures that we will regularly use as examples: *degree*, *closeness*, *eccentricity*, and *all-subgraphs* centralities. During this work, we also mention *betweenness* [11], *decay* [16], *PageRank* [22], and *eigenvector* [7] centralities. Given that we do not use them directly, we refer the reader to the corresponding works for a definition.

Degree centrality. Degree centrality is probably the most straightforward measure. Basically, the bigger the neighborhood of a vertex (i.e., adjacent vertices), the more central it is in the graph. Formally, we define *degree centrality* as follows: $\text{DEGREE}(v, G) = |N_G(v)|$.

Closeness centrality. For every graph G and vertices $v, u \in V(G)$ we define the *distance* between v and u in G as $d_G(v, u) = |\pi_{v,u}|$ where $\pi_{v,u}$ is a shortest path from v to u in G . Then *closeness centrality* [25] is defined as: $\text{CLOSENESS}(v, G) = 1 / \sum_{u \in K_v(G)} d_G(v, u)$ where $K_v(G)$ is the connected component of G containing v . Closeness is usually called a geometrical measure because it is based on the distance inside a graph. The intuition behind closeness centrality is simple: the closer a vertex is to everyone in the component (i.e., $\sum_{u \in K_v(G)} d_G(v, u)$ is small) the more important it is.

Eccentricity centrality. Another important notion in graph theory is radius. In simple words, we can define the center of a graph G as the vertex that minimizes the maximum distance in G . Formally, v is the center of G if it minimizes $\max_{u \neq v \in V(G)} d_G(v, u)$. Then, the radius of G is defined as the maximum distance from the center. Now, *eccentricity measure* [13] is precisely the one centrality that selects the center of a graph as the most important vertex, defined as $\text{ECCENTRICITY}(v, G) = 1 / \max_{u \in V(G)} d_G(v, u)$.

All-subgraphs centrality. Given a graph $G = (V, E)$ and a vertex $v \in V$, we denote by $\mathcal{A}(v, G)$ the set of all connected subgraphs of G that contain v , formally, $\mathcal{A}(v, G) = \{S \subseteq G \mid v \in V(S) \text{ and } S \text{ is connected}\}$. Then *all-subgraphs centrality* [24] of v in G is defined as: $\text{ALLSUBGRAPHS}(v, G) = \log_2 |\mathcal{A}(v, G)|$. All-subgraphs centrality was recently proposed in [24], proving that it satisfies several desirable properties as a centrality measure. Intuitively, it says that a vertex will be more relevant in a graph if it has more connected subgraphs surrounding it.

Undirected Trees. This paper is about *undirected trees* (or just trees), so we use some special notation for them. Specifically, we say that a graph T is a *tree* if it is connected and for every $u, v \in V(T)$ there exists a unique path that connects u with v in T . We usually use T to denote a tree. Further, we say that $v \in V(T)$ is a *leaf* of T if $|N_T(v)| = 1$. If v is a leaf and $N_T(v) = \{u\}$, then we say that u is the *parent* of v . Note that trees are a special class of undirected graphs, and all previous definitions apply. In particular, we can use and apply centrality measures over trees.

We say that T' is a *subtree* of T if $T' \subseteq T$ and T' is a tree. We also say that T' is a *complete subtree* of T if $T' \subseteq T$ and there is at most one vertex in $V(T')$ connected to some vertex in $V(T) \setminus V(T')$, namely, $|\{v \in V(T') \mid \exists u \in V(T) \setminus V(T'). \{u, v\} \in E(T)\}| \leq 1$.

The following notation will be useful in the paper to decompose trees. Given a tree T and two adjacent vertices $u, v \in V(T)$, we denote by $T_{u,v}$ the maximum subtree of T that contains u and not v . For example, if $T = \triangleright \circ \blacktriangleleft$, then $T_{\circ, \bullet} = \triangleright \circ$ and $T_{\bullet, \circ} = \bullet \blacktriangleleft$.

Finally, we consider some special trees to give examples or show some properties of centrality measures. For a vertex v we define $G_v = (\{v\}, \emptyset)$, and for an edge e we define $G_e = (e, \{e\})$, namely, the graphs with *one isolated vertex* v or *one isolated edge* e , respectively. Similarly, for any $n \geq 1$ we write L_n for the *line* with n vertices where $V(L_n) = \{0, \dots, n-1\}$ and $E(L_n) = \{\{i, i+1\} \mid 0 \leq i < n-1\}$.

3 Tree rooting centrality measures

We start by giving a formal definition of when a centrality measure C roots trees. For this, let C be a centrality measure. We define the set $\text{MAX}_C(T)$ to be the set of most central vertices with respect to C in a tree T , namely, $v \in \text{MAX}_C(T)$ iff $C(u, T) \leq C(v, T)$ for every $u \in V(T)$.

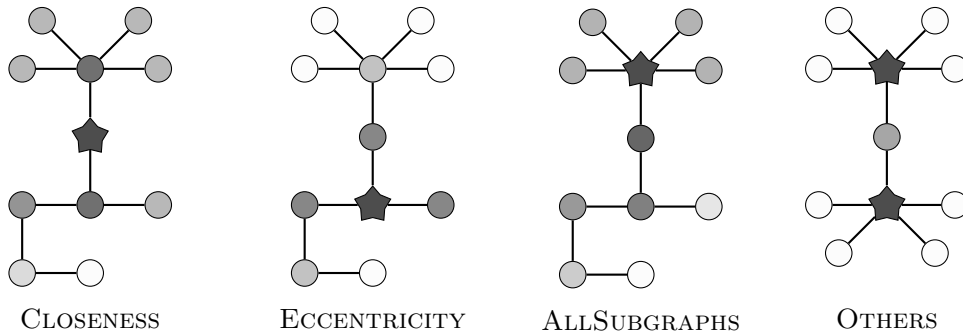
► **Definition 1.** We say that a centrality measure C roots trees if for every tree T , the set of most central vertices $\text{MAX}_C(T)$ consists of one vertex or two adjacent vertices. Moreover, for every $u \notin \text{MAX}_C(T)$ if $u_0 u_1 \dots u_n$ is the unique path from $\text{MAX}_C(T)$ to u , then $C(u_i, T) > C(u_{i+1}, T)$ for every $i \in [0, n-1]$.

In the following, if a centrality measure roots trees, we also say that it satisfies the *tree rooting property* (i.e., Definition 1). We can motivate this property as follows. We treat vertices with the highest centrality as “roots”. For the first part of the definition, we assume there is a single source of importance – one vertex or two adjacent vertices. We allow two adjacent vertices to be the roots, as in some graphs, due to their symmetrical structure, it is impossible to indicate one most central vertex. This is, for example, the case of a line with an even number of vertices (e.g., $\circ - \circ - \circ - \circ$). In such a scenario the edge between both can be considered the real root of the tree. For the second part, we assume that the centrality should decrease from the root through branches. This restriction aligns with the intuition that the closer a vertex is to the root, the more central it is.

We continue by giving examples of measures that root trees, and some others that do not.

► **Example 2.** Closeness, eccentricity, and all-subgraphs all root trees. It was already noticed [18] that over trees, closeness and eccentricity define at most two maximum vertices, and both are connected. On the other hand, it is more subtle to show that all-subgraphs centrality roots trees. This fact, however, will follow from the framework developed in Section 4. As an illustration, in Figure 1 we show how closeness, eccentricity, and all-subgraphs behave over the same tree. One can verify that each measure declares one vertex with the maximum centrality (this vertex is marked with a black star). Moreover, the centrality decreases through the branches (the lower the centrality the whiter the color). It is interesting that, although the three measures root trees, they declare different vertices as the most central.

► **Example 3.** One can easily check that degree centrality does not root trees. Indeed, the last tree at Figure 1 is an example where degree centrality declares two maximum vertices, and they are not adjacent. Indeed, all measures presented in Section 2, except closeness, eccentricity, and all-subgraphs, do not root trees. For all of them, the last tree at Figure 1 is a counterexample where they violate the tree rooting property. At Figure 2, we show a table that summarize which centrality measures considered in this paper root trees.



■ **Figure 1** The first three trees exemplify how closeness, eccentricity, and all-subgraphs centralities root trees. We mark the most central vertex with a black star. The colors show how the centrality value decreases through the branches (i.e., whiter vertices are less central). The fourth tree is a counter-example that shows why other centralities do not root trees.

An important consequence of assigning a root to a tree is that each vertex has a parent (except for the root). Here, the unique path from the root to the vertex defines its parent. Another possibility would be to use the centrality measure to find the neighbour with higher centrality, and declare it as the parent. We capture this intuition in the following property.

► **Definition 4.** We say that a measure C satisfies the at-most-one-parent property if for every tree T and $v \in V(T)$ there exists at most one neighbour $u \in N_T(v)$ such that $C(v, T) \leq C(u, T)$.

Clearly, if a centrality measure C roots trees, then it also satisfies at-most-one-parent. The other direction is also true, providing an alternative characterization for tree rooting.

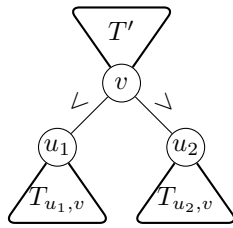
► **Proposition 5.** A centrality measure roots trees if, and only if, it satisfies the at-most-one-parent property.

Notice that tree rooting is a global property over a tree but, instead, at-most-one-parent is a local property of the neighbourhoods of a tree, which is easier to prove for a centrality measure. Indeed, in the next section we use this alternative definition to prove our main characterization for tree rooting.

In some trees the root is uniquely characterized solely by the tree rooting property. This happens because every centrality must be closed under isomorphism which implies that isomorphic vertices have the same centrality. For example, one can verify that for the line L_n the roots must be the set $\{\lfloor \frac{n-1}{2} \rfloor, \lceil \frac{n-1}{2} \rceil\}$. Indeed, every vertex $i \leq \frac{n-1}{2}$ is isomorphic with the vertex $n-1-i$ in L_n . Then, if i is the root, then $n-1-i$ must be the root as well. Given that vertices i and $n-1-i$ are not connected if $i < \frac{n-1}{2}$, we get that every centrality that roots trees must declare $\{\lfloor \frac{n-1}{2} \rfloor, \lceil \frac{n-1}{2} \rceil\}$ as the root. This idea can be extended to all symmetric trees: if T is a tree with a vertex \circ that connects two isomorphic subtrees (i.e., $T = \triangle \circ \triangle$), then \circ must be the root of T . We generalize this property as follows.

► **Definition 6.** We say that a centrality measure C is symmetric over trees if for every tree T , vertex v , and different neighbors $u_1, u_2 \in N_T(v)$ such that $(u_1, T_{u_1, v}) \cong (u_2, T_{u_2, v})$, then $C(u_1, T) < C(v, T)$ and $C(u_2, T) < C(v, T)$.

12:8 How Do Centrality Measures Choose the Root of Trees?



Measures	Can root?	Potential function?
Closeness	Yes	Yes
Eccentricity	Yes	Yes
All-Subgraphs	Yes	Yes
Degree	No	Yes
Betweenness	No	Yes
Decay	No	Yes
PageRank	No	?
EigenVector	No	?

■ **Figure 2** At the left, a graphic illustration of the symmetry property over a tree T . Subtrees $T_{u_1, v}$ and $T_{u_2, v}$ are isomorphic, and subtree T' represents the rest of T hanging from v . At the right, a table summarizes which centrality measures root trees and which one admits a potential function.

Figure 2 (left) is a graphical representation of the symmetry property. It generalizes the previously discussed intuition by considering any pair of isomorphic subtrees in a (not necessarily symmetric) tree. Interestingly, every centrality measure that roots trees must be symmetric over trees.

► **Proposition 7.** *If a centrality measure C roots trees, then C is symmetric over trees.*

A symmetric centrality measure may not root trees. For example, a centrality measure could root trees with non-trivial automorphisms but not root trees when there is none. Despite this, symmetry property is crucial for finding a characterization for tree rooting, as we will show in the next section.

4 Potential functions

What have in common closeness, eccentricity, and all-subgraphs centralities? What is the fundamental property so they can root trees? A crucial ingredient for understanding the connection between these measures is what we call a *potential function* for a centrality measure.

► **Definition 8.** *Given a centrality measure C , we say that $f : \mathcal{VG} \rightarrow \mathbb{R}$ is a potential function for C if f is closed under isomorphism on \mathcal{VG} and, for every tree T and every adjacent vertices $u, v \in V(T)$, it holds that $C(u, T) \leq C(v, T)$ if, and only if, $f(u, T_{u, v}) \leq f(v, T_{v, u})$.*

A potential function is a function that measures the “potential” of every *rooted* tree, i.e., a tree with one node selected and the assessment depends on the selection. Now, a centrality measure admits some potential function if the comparison between two adjacent vertices is determined by the potential of their corresponding subtrees. Interestingly, in the following examples we show that several centrality measures admit a potential function.

► **Example 9.** Degree, closeness, eccentricity, all-subgraphs centralities have the following potential functions:

$$\begin{aligned}
 f_d(v, T) &:= |N_T(v)| && (\text{degree}) \\
 f_c(v, T) &:= |V(T)| && (\text{closeness}) \\
 f_e(v, T) &:= \max_{u \in V(T)} d_T(v, u) && (\text{eccentricity}) \\
 f_a(v, T) &:= |\mathcal{A}(v, T)| && (\text{all-subgraphs})
 \end{aligned}$$

Let us verify that each function above is a potential function for its corresponding measure. Take an arbitrary tree T and two adjacent vertices u and v . It is straightforward to check that f_d is a potential function for degree centrality. Indeed, if u has a smaller degree in its subtree (i.e., without considering the common edge), then it also has a smaller centrality.

For closeness centrality, the potential function f_c is simply the number of vertices in a tree. To see this, note that vertex u has a distance smaller by one than v to all vertices from $T_{u,v}$; analogously, vertex v has a distance smaller by one than u to all vertices from $T_{v,u}$. As a result, out of both vertices, the one with the larger subtree has the smaller sum of distances which results in the higher closeness centrality.

For eccentricity the potential function f_e is the height of a tree, i.e., the distance to the farthest vertex. This is because if subtree $T_{u,v}$ is higher than $T_{v,u}$, then vertex u has smaller distance to the farthest vertex in T which results in the higher eccentricity. Interestingly, f_e is an inverse of eccentricity.

Finally, for all-subgraphs centrality the potential function f_a is the number of subgraphs that contain vertex v . The reason is that we have $|\mathcal{A}(u, T)| = |\mathcal{A}(u, T_{u,v})| + |\mathcal{A}(u, T_{v,u})| \cdot |\mathcal{A}(v, T_{v,u})|$ and, symmetrically, $|\mathcal{A}(v, T)| = |\mathcal{A}(v, T_{v,u})| + |\mathcal{A}(v, T_{u,v})| \cdot |\mathcal{A}(u, T_{u,v})|$. This implies that if u has more subgraphs in $T_{u,v}$ than v in $T_{v,u}$, then it also has higher all-subgraphs centrality. Hence, as in the case of degree centrality, the potential function coincides with the centrality itself.

One can also show that betweenness and decay centralities have a potential function. As we will see later, there are centrality measures that do not have potential functions. For the particular case of PageRank and EigenVector it is not clear whether they admit a potential function. The table at Figure 2 (right) summarizes which centrality measures have a potential function.

A potential function determines the centrality order between two adjacent vertices, but it does not imply the relation between non-adjacent vertices. Although this information is weaker than the centrality measure itself, it is exactly what we need to understand the centrality measures that root trees. Precisely, which measures with potential functions root trees? To answer this question, we first need to capture the symmetry property through the lens of potential functions.

► **Lemma 10.** *Let C be a centrality measure and f a potential function for C . Then C is symmetric over trees if, and only if, for every tree T and every pair of adjacent vertices $u, v \in V(T)$, it holds that $f(u, T_{u,v}) < f(v, T)$.*

By the previous result, we will call potential functions with this property *symmetric*. Next, we show that symmetric potential functions characterize the tree rooting property.

► **Theorem 11.** *Let C be a centrality measure that admits a potential function f . Then C roots trees if, and only if, f is symmetric.*

► **Example 12.** Continuing with Example 9 we can check that the potential functions f_c , f_e , and f_a for closeness, eccentricity, and all-subgraphs, respectively, are symmetric. Indeed, for all these functions a subtree always has less potential than the whole tree. For this reason, $f_x(u, T_{u,v}) < f_x(v, T)$ for $x \in \{c, e, a\}$ since $T_{u,v}$ is a subgraph of T . By Theorem 11, this proves that closeness, eccentricity, and all-subgraphs root trees.

In turn, the potential functions of degree centrality (as well as betweenness and decay centralities) are not symmetric and, therefore, do not root trees. For example, for degree centrality we can take $T = \begin{array}{c} \bullet \\ \diagup \quad \diagdown \\ u \text{---} v \\ \diagdown \quad \diagup \\ \bullet \end{array}$, and check that $f_d(u, T_{u,v}) > f_d(v, T)$. Therefore, f_d is not symmetric.

12:10 How Do Centrality Measures Choose the Root of Trees?

We showed that all standard centrality measures that root trees can be defined through potential functions. The natural question is: is it true for all measures that root trees? In the following result, we show that this is not the case and there exists a measure that roots a tree, but does not have a potential function.

► **Proposition 13.** *There exists a centrality measure that roots trees but does not admit a potential function.*

Potential functions and Theorem 11 explain why some measures root trees and others do not. In the following sections, we use this framework to further understand the tree rooting centrality measures in terms of algorithms, consistency, and the design of new measures.

5 An algorithm to find the root

Even though not every tree rooting measure has a potential function, having one gives us some essential properties that we can exploit. In particular, having a symmetric potential function implies finding the root of any tree in $\mathcal{O}(n \log(n))$ -time under some assumptions on the efficiency to compute the potential function. Notice that the naive approach of computing the centrality for each vertex separately and then choosing the one with the highest centrality runs in quadratic time (i.e., by assuming that computing the centrality of a single vertex takes linear time). Instead, the algorithm presented here runs in $\mathcal{O}(n \log(n))$ for every centrality measure that admits a (locally-computable) symmetric potential function.

The main intuition behind this algorithm is based on the following property satisfied by centrality measures with a symmetric potential function.

► **Proposition 14.** *Let C be a centrality measure that has a symmetric potential function f . Let T be any tree, $w_1, w_n \in V(T)$, and $w_1 w_2 \dots w_n$ be the unique path connecting w_1 to w_n in T . Whenever $f(w_1, T_{w_1, w_2}) \leq f(w_n, T_{w_n, w_{n-1}})$ then $C(w_1, T) \leq C(w_2, T)$.*

In other words, Proposition 14 says that if the potential of the subtree hanging from w_1 is less than the potential of the subtree hanging from w_n , then a root should be closer to the adjacent vertex of w_1 that is towards the direction of w_n . This result gives us a way to traverse a tree, starting from the leaves and going up until we find the root. More specifically, starting from the leaves, by Proposition 14 we can compare the potential of two opposite complete subtrees. Then, vertices with higher potential in their subtree indicate the direction of higher centrality. When we finally reach two connected vertices, the vertex with higher (or equal) potential is a root.

Algorithm 1 implements the above intuition based on Proposition 14. It receives as input a tree T and a symmetric potential function f , and outputs a root of T with respect to f . For implementing Algorithm 1 we need two data structures, denoted by H and Q . The first data structure H is a *key-value map* (i.e., a Hash-table), where a key can be any vertex $v \in V(T)$ and its value is a subset of $N_T(v)$. We denote the value of v (i.e., a key) in H by $H[v]$. By some abuse of notation, when $H[v]$ is a single vertex, we write $u \leftarrow H[v]$ to retrieve and store this vertex in u . The second data structure Q is a *priority-Queue*. For $v \in V(T)$ and $p \in \mathbb{R}$ we write $Q.\text{insert}(v, p)$ to insert v in Q with priority p . We also write $v \leftarrow Q.\text{pull}()$ to remove the vertex with the lowest priority from Q , and store it in v . For both structures, these operations can be implemented in $\mathcal{O}(\log(n))$ -time where n is the number of inserted objects [9].

Algorithm 1 starts by initializing Q as empty and H with all key-value pairs $(v, N_T(v))$ (lines 2-3). Then, it runs over all leaves v of T and inserts it into Q with priority $f(v, G_v)$ where G_v is the tree with an isolated vertex v (lines 4-5). As we already mentioned, the

■ **Algorithm 1** Find a root given a tree.

Input: A non-trivial tree T and a symmetric potential function f .
Output: The most central vertex of T according to f .

```

1 Function Find-a-root( $T, f$ ):
2    $Q \leftarrow$  Empty-queue
3    $H \leftarrow \{(v, N_T(v)) \mid v \in V(T)\}$ 
4   foreach  $v$  leaf of  $T$  do
5      $Q.insert(v, f(v, G_v))$ 
6   while  $Q.size() > 1$ 
7      $v \leftarrow Q.pull()$ 
8      $u \leftarrow H[v]$ 
9      $H[u] \leftarrow H[u] \setminus \{v\}$ 
10    if  $|H[u]| = 1$  then
11       $w \leftarrow H[u]$ 
12       $Q.insert(u, f(u, T_{u,w}))$ 
13    return  $Q.pull()$ 

```

intuition is to start from all leaves v and use its potential (as a single vertex) for comparing it with other vertices. Then we loop while the number of elements in Q is greater than 1. Recall that any non-trivial tree has at least two leaves, and therefore the algorithm reaches line 6 with $Q.size() \geq 2$ for the first time. Instead, if T is trivial, we return the single vertex directly in line 13.

We remove the vertex with the lowest priority from Q in each iteration and store it in v (line 7). This step discards v as a possible root (by Proposition 14) and moves towards its “parent” represented by $u \leftarrow H[v]$ (line 8). Given that we discarded v , we remove v as a child of u , where $H[u]$ contains the current children of u (line 9). Indeed, when $|H[u]| = 1$ (line 10) this means that we have reached u , its parent is $w \leftarrow H[u]$ (line 11) and its complete subtree $T_{u,w}$ hanging from u must be evaluated with f , and inserted in Q (line 12). An important invariant during the while-loop is that any vertex v in Q satisfies $|H[v]| = 1$ (except at the end of the last iteration). Conceptually, if $H[v] = \{w\}$, this invariant means that w is the parent of v and we are using the potential of the subtree $(v, T_{v,w})$ for comparing v with other vertices. Then when v is the vertex with the lowest priority on Q , it means that other vertices beat it, and a root must be towards its parent.

Finally, when there is only one vertex left in Q , it beats all other vertices, and it should be one of the roots. It is necessary to mention that if T has two roots, we could also output the second root by slightly modifying the algorithm.

Regarding time complexity, the reader can check that the for- and while-loops take linear time on $|T|$. Each operation over H and Q take at most $\log(|T|)$ steps, and overall it sum up to $\mathcal{O}(|T| \log(|T|))$ if computing f takes constant time.

Of course, the previous assumption is not always true, given that f can be any symmetric potential function. To solve this, we say that f is *locally-computable* if, for every T and $u \in V(T)$, $f(u, T)$ can be computed in $\mathcal{O}(k)$ -time from the values of its k -neighbors, namely, from $f(u_1, T_{u_1,v}), \dots, f(u_k, T_{u_k,v})$ where $N_T(u) = \{u_1, \dots, u_k\}$. Note that by book-keeping the values $f(u_1, T_{u_1,v}), \dots, f(u_k, T_{u_k,v})$ of the neighbors of u , we can compute $f(u, T_{u,w})$ (line 12) in $\mathcal{O}(|N_T(u)|)$ -time. If we sum this extra time over all vertices, it only adds $\mathcal{O}(|T|)$ -steps to the total running time of the algorithm.

12:12 How Do Centrality Measures Choose the Root of Trees?

► **Proposition 15.** *Given a tree T and a symmetric potential function f , Algorithm 1 returns a root of T with respect to f . Moreover, if f is locally-computable, the algorithm runs in $\mathcal{O}(|T| \cdot \log(|T|))$ -time.*

The reader can easily check that the potential functions of closeness, eccentricity, and all-subgraphs are locally-computable (see also Section 7), and then Algorithm 1 can be used for any of these measures to find the root of any tree in $\mathcal{O}(|T| \cdot \log(|T|))$.

6 Consistent rooting

The tree rooting property fixes the “shape” of a centrality measure in every possible tree. However, it does not impose any relation between roots in different trees. As a result, even a small change (e.g., adding a leaf) may move the root arbitrarily since there might be no relation between the roots in a tree and the altered tree. To give an example, a centrality measure may be defined in one way for trees with odd number of vertices, but in a completely different way for trees with even number of vertices.

To this end, we propose a notion of *consistency*. Consistency states that if we add a leaf to the tree, then the root may move only in its direction.

► **Definition 16.** *We say that a centrality measure C consistently roots trees if it roots trees and for every tree T and vertices $u, v \in V(T)$, $w \notin V(T)$ such that $u \in \text{MAX}_C(T)$ it holds $\text{MAX}_C(T + \{v, w\}) \subseteq \pi_{u,w} \cup \text{MAX}_C(T)$, where $\pi_{u,w}$ is the path between u and w in $T + \{v, w\}$.*

We can verify that closeness, eccentricity, and all-subgraphs centralities all consistently root trees.

Consistency is a property of measures that root trees. However, it can be also interpreted using a natural property for arbitrary centrality measures that we call *monotonicity*. Monotonicity states that if vertex v has a higher (or equal) centrality than its neighbour u in a tree, then this fact will not change if we add a leaf on the side of vertex v .

► **Definition 17.** *We say that a centrality measure C is monotonic if for every tree T , vertices $v, u, w \in T$ such that $\{v, u\} \in E(T)$, $w \in T_{u,v}$ and vertex $w' \notin V(T)$ if $C(v, T) < C(u, T)$, then $C(v, T + \{w, w'\}) < C(u, T + \{w, w'\})$.*

Monotonicity is in fact a general property satisfied by many centrality measures, including all geometric centralities such as closeness and decay. The following result ties both concepts: monotonicity and consistency of rooting.

► **Proposition 18.** *Let C be a centrality measure that roots trees. Then C consistently roots trees if, and only if, it is monotonic.*

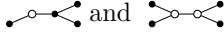
Let us turn our attention to the relation between tree rooting and potential functions. If a centrality that roots trees admits a potential function, then it must be consistent to some extent. However, as it turns out, it might not consistently root trees, as we show in the following counterexample.

► **Example 19.** Consider the following ad-hoc centrality measure:

$$C(v, T) = \text{ECCENTRICITY}(v, T) - (1/|T|^2) \cdot \text{CLOSENESS}(v, T).$$

Intuitively, if two vertices in a tree have different eccentricity, then their eccentricity differs by more than $1/|T|^2$. Also, $\text{CLOSENESS}(v, T) \in (0, 1]$. Hence, we have $C(u, T) < C(v, T)$ if, and only if, u has a smaller eccentricity than v or equal eccentricity, but higher closeness.

It is easy to verify that the following potential function corresponds to C : $f(v, T) = h(v, T) - 1/|T|$, where $h(v, T)$ is the distance from v to the farthest vertex in T .

To show that consistency is violated consider trees  (vertices with the highest centrality are marked with white color). Consistency states that in the second tree the root should stay on the left-hand side which is not the case here.

To characterize which of the centrality measures with potential function root trees consistently, we look at the restriction that monotonicity imposes on the potential function.

► **Proposition 20.** *Let C be a centrality measure that admits a potential function f . Then C is monotonic iff for every tree T , subtree T' of T , and $v \in V(T')$ it holds $f(v, T) \geq f(v, T')$.*

In particular, the potential function from Example 19 violates this condition, as adding vertices to a tree without increasing its height decreases the value of the potential function.

Now we can summarize rooting results and consistency regarding potential functions as one of our principal theorems.

► **Theorem 21.** *Let C be a centrality measure that admits a potential function f . Then C is monotonic and symmetric if, and only if, for every tree T , proper subtree T' of T and vertices $v \in V(T)$, $u \in V(T')$ it holds $f(v, T) \geq f(u, T')$ and $f(v, T) > f(u, T')$ if $u \neq v$.*

7 Families of potential functions

In this section, we apply the previous results by showing how to design potential functions that consistently root trees. Specifically, using the following results, we can derive an infinite family of potential functions. This family shows infinite ways to root trees with good characteristics, namely, that are consistent and computable in $\mathcal{O}(n \log(n))$ time. In the following, we recall some standard definitions of monoids, to then define potential functions through them.

A *monoid* (over \mathbb{R}) is a triple $(\mathbb{M}, *, \mathbb{1})$ where $\mathbb{M} \subseteq \mathbb{R}$, $*$ is a binary operation over \mathbb{M} , $*$ is associative, and $\mathbb{1} \in \mathbb{M}$ is the identity of $*$ (i.e., $r * \mathbb{1} = \mathbb{1} * r = r$). We further assume that monoids are commutative, namely, $*$ is commutative. Examples of (commutative) monoids are $(\mathbb{R}_{\geq 0}, +, 0)$ and $(\mathbb{R}_{\geq 1}, \times, 1)$, where we use $\mathbb{R}_{\geq c}$ for all reals greater or equal than c . For the sake of presentation, in the following we will usually refer the monoid

► **Definition 22.** *Given a monoid $(\mathbb{M}, *, \mathbb{1})$ and $\ell : \mathbb{M} \rightarrow \mathbb{M}$, we define the potential function $f_{*,\ell}$ recursively as follows:*

1. *For a vertex v , we define $f_{*,\ell}(v, G_v) = \mathbb{1}$, where G_v is the tree with an isolated vertex v .*
2. *For a tree T and a leaf $v \in V(T)$ hanging from its parent $u \in V(T)$, we define $f_{*,\ell}(v, T) = \ell(f_{*,\ell}(u, T_{u,v}))$. In other words, we apply ℓ to the potential function of the subtree rooted at u . We call ℓ the leaf-function of $f_{*,\ell}$.*
3. *Given two trees T_1 and T_2 with $V(T_1) \cap V(T_2) = \{v\}$, we define $f_{*,\ell}(v, T_1 \cup T_2) = f_{*,\ell}(v, T_1) * f_{*,\ell}(v, T_2)$.*

A potential function f is constructive if there exists a monoid $$ and a leaf-function ℓ such that $f = f_{*,\ell}$.*

Notice that $f_{*,\ell}(v, T)$ is uniquely determined by the three cases above. Specifically, suppose that $u_1, \dots, u_k \in N_T(v)$ are the neighbors of v on T . Then we can decompose T by considering all subtrees $T_{u_i, v} + \{u_i, v\}$ and compute $f_{*,\ell}(v, T)$ recursively:

$$f_{*,\ell}(v, T) = \ell(f_{*,\ell}(u_1, T_{u_1, v})) * \dots * \ell(f_{*,\ell}(u_k, T_{u_k, v}))$$

12:14 How Do Centrality Measures Choose the Root of Trees?

until we reach a single vertex. Furthermore, $f_{*,\ell}$ is closed under isomorphism over \mathcal{VG} given that $*$ is associative and commutative. Thus, we conclude that $f_{*,\ell}$ is well-defined and could work as a potential function. In addition, $f_{*,\ell}$ is locally-computable since $f_{*,\ell}(v, T)$ can be computed from its k -neighbors.

► **Example 23.** All potential functions presented in Example 9 are constructive by considering the following monoids and leaf-functions:

$(\mathbb{R}_{\geq 0}, +, 0)$	$\ell(x) = 1$	(degree)
$(\mathbb{R}_{\geq 1}, a + b - 1, 1)$	$\ell(x) = x + 1$	(closeness)
$(\mathbb{R}_{\geq 0}, \max, 0)$	$\ell(x) = x + 1$	(eccentricity)
$(\mathbb{R}_{\geq 1}, \times, 1)$	$\ell(x) = x + 1$	(all-subgraphs)

It is easy to check that each monoid and leaf-function defines the corresponding potential function of the above measures.

One advantage of the previous definition is that it shows a way for constructing potential functions. Moreover, we can study which properties are necessary over $*$ and ℓ to guarantee that $f_{*,\ell}$ consistently root trees. Towards this goal, we recall some standard definitions for monoids and functions. A function f is called *monotonic* if $x \leq y$, implies $f(x) \leq f(y)$ for every x, y . A monoid $(\mathbb{M}, *, \mathbb{1})$ is called *ordered* if $x \leq y$, implies $x * z \leq y * z$ for every $x, y, z \in \mathbb{M}$. Further, it is called *positively ordered* if in addition $\mathbb{1} \leq x$ for every $x \in \mathbb{M}$.

► **Lemma 24.** *Let $(\mathbb{M}, *, \mathbb{1})$ be a monoid and $\ell : \mathbb{M} \rightarrow \mathbb{M}$ a leaf-function. The potential function $f_{*,\ell}$ consistently roots trees whenever (1) $x < \ell(x)$ for every x , (2) ℓ is monotonic, and (3) $(\mathbb{M}, *, \mathbb{1})$ is positively ordered.*

For example, the monoids and leaf-functions of closeness, eccentricity, and all-subgraphs (see Example 23), satisfy properties (1) to (3) and, as we know, they consistently root trees. On the other hand, degree's leaf-function does not satisfy (1), and therefore, it does not root trees.

Lemma 24 shows sufficient conditions over $*$ and ℓ to consistently root trees. To get a necessary condition, we need to add some technical restrictions, and to slightly weaken conditions (2) and (3). Towards this goal, let $\text{Range}_{*,\ell}$ be the range of $f_{*,\ell}$. Define $\bar{*}$ and $\bar{\ell}$ to be the monoid $(\mathbb{M}, *, \mathbb{1})$ and function ℓ restricted to $\text{Range}_{*,\ell}$. For two values x and y , we say that x is a *subtree-value* of y if there exist T and T' such that T is a subtree of T' , $f_{*,\ell}(u, T) = x$, and $f_{*,\ell}(u, T') = y$ for some $u \in V(T)$. Then we say that $\bar{\ell}$ is *monotonic over subtrees* if $x \leq y$ and x is a subtree-value of y implies that $\bar{\ell}(x) \leq \bar{\ell}(y)$. Similarly, we say that $\bar{*}$ is *positively ordered over subtrees*, if $x \leq y$ and x is a subtree-value of y , then $x * z \leq y * z$ for every $z \in \text{Range}_{*,\ell}$, and $\mathbb{1} \leq x$ for every $x \in \text{Range}_{*,\ell}$.

► **Theorem 25.** *Let $(\mathbb{M}, *, \mathbb{1})$ be a monoid and $\ell : \mathbb{M} \rightarrow \mathbb{M}$ a leaf-function. The potential function $f_{*,\ell}$ consistently roots trees if, and only if, (1) $x < \bar{\ell}(x)$ for every $x \in \text{Range}_{*,\ell}$, (2) $\bar{\ell}$ is monotonic over subtrees, and (3) $\bar{*}$ is positively ordered over subtrees.*

Theorem 25 and, specifically, Lemma 24 give the ingredients to design potential functions that consistently root trees and, further, we have an algorithm to find the root in $\mathcal{O}(n \log(n))$. For instance, take a triple $(a, b, c) \in \mathbb{R}^3$. Then define the monoid $(\mathbb{R}_{\geq c}, *_c, c)$ and leaf-function $\ell_{a,b}$ such that: $x *_c y := \frac{x \cdot y}{c}$ and $\ell_{a,b}(x) := a \cdot x + b$. For example, if we consider $a = b = c = 1$, we get the monoid and leaf-function for the potential function of all-subgraph centrality (see Example 23). Interestingly, one can verify that, if $a \geq 1$, $b > 0$ and $c > 0$, then $*_c$ is a monoid. Moreover, $*_c$ and $\ell_{a,b}$ satisfy properties (1) to (3) of Lemma 24 and we get the following result.

► **Proposition 26.** *For every $(a, b, c) \in \mathbb{R}^3$ with $a \geq 1$, $b > 0$, and $c > 0$, the potential function $f_{*c, \ell_{a,b}}$ consistently root trees.*

Finally, we want to know if we can get different roots for different values (a, b, c) . In other words, is it the case that for every different triples (a, b, c) and (a', b', c') there exists a tree T such that the root of T according to $f_{*c, \ell_{a,b}}$ is different to one chosen by $f_{*c', \ell_{a', b'}}$? The next result shows that $\{f_{*c, \ell_{a,b}} \mid c \geq 1, b > 0, c > 0\}$ is indeed an infinity family of different potential functions for tree rooting.

► **Proposition 27.** *There exists an infinite set $S \subseteq \mathbb{R}^3$ such that for every $(a, b, c), (a', b', c') \in S$, there exists a tree T where the roots of T according to $f_{*c, \ell_{a,b}}$ are not the same as roots of T according to $f_{*c', \ell_{a', b'}}$.*

8 Discussion

We end the paper by discussing some extensions and applications.

Extensions to other classes of graphs. An obvious question is whether one can generalize the know-how acquired on trees to other classes of graphs. We agree that further research is needed to extend potential functions to new graph families. Nevertheless, we see some exciting directions in which our work can be extended. In particular, the idea of potential functions extends to arbitrary graphs by considering the endpoints of a bridge (i.e., a cut edge) instead of adjacent vertices of a tree. More in detail, let G be a connected graph and $\{u, v\}$ be an edge such that G_u, G_v are two connected components of $G - \{u, v\}$ that contain u, v , respectively. We say that $f : \mathcal{VG} \rightarrow \mathbb{R}$ is a *graph* potential function for C if for every such graph G it holds $C(u, G) \leq C(v, G)$ if, and only if, $f(u, G_u) \leq f(v, G_v)$. This property generalizes (tree) potential functions, as every two adjacent nodes in a tree form a bridge. As it turns out, all centrality measures listed on Figure 2 that have (tree) potential functions (degree, closeness, betweenness, eccentricity, all-subgraphs, decay) have identical graph potential functions. For example, $\text{ECCENTRICITY}(v, G) \leq \text{ECCENTRICITY}(u, G)$ if and only if $f_e(v, G_v) \leq f_e(u, G_u)$ for f_e defined in Example 9. Interestingly, some centrality measures have identical potential functions on trees but different ones on general graphs (for example, closeness and random-walk closeness centralities).

Applications. In this work, we focused on the foundational aspects of understanding centrality measures over trees, and we left for future work the application in the context of data management. Given the axiomatic approach of our work and given that tree structures are ubiquitous in data management, we believe that potential functions and their implications on rooting trees could find several exciting applications. In the following, we present some possible applications of this work in data management scenarios.

In conjunctive query answering, the class of acyclic queries is of particular interest, given that each query has a join tree that permits efficient evaluation in linear time on data complexity [3]. For this, the so-called Yannakakis algorithm [30] performs a bottom-up traversal of the join tree for filtering the tuples that will not be part of the output. In particular, the different ways one can root the join tree gives rise to several individual computational schedules to obtain the same results [3]. Here, choosing the root of a join tree by using some specific potential function could lead to improving existing join evaluation algorithms in practice. We leave for future work on how one can use this principle for query evaluation in the presence of join trees.

12:16 How Do Centrality Measures Choose the Root of Trees?

Another possible application is in the context of tree-structured data, like XML or JSON documents. Although this data is usually rooted, assessing the most crucial node using a centrality measure can lead to a better understanding of the document's structure. For instance, given a tree-structured document, one could measure the difference between the root provided by potential function and the original root and see how this difference affects query evaluation, document representation, or other metrics.

Finally, in a broader sense, one could see centrality measures over graphs as an instance of a general database problem: find the most central data object in the data model given its underlying structure. The data model could be a relational database, an object-oriented database, an RDF database, or even a tree-structure database. For all these cases, the principle should be the same: the more relevant the data object is for its data model, the more central it should be. The present work could be seen as the first step toward this direction, namely, understanding data centrality in the case of trees. We leave for future work on how to extend this line of research to other classes of graphs or data models.

References

- 1 Neo4j: Centrality. <https://neo4j.com/docs/graph-data-science/current/algorithms/centrality/>.
- 2 Tigergraph: Centrality algorithms. <https://docs.tigergraph.com/graph-ml/current/centrality-algorithms/>.
- 3 Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of databases*, volume 8. Addison-Wesley Reading, 1995.
- 4 Francis Bloch, Matthew O. Jackson, and Pietro Tebaldi. Centrality measures in networks. *arXiv preprint*, 2019. [arXiv:1608.05845](https://arxiv.org/abs/1608.05845).
- 5 Paolo Boldi, Alessandro Luongo, and Sebastiano Vigna. Rank monotonicity in centrality measures. *Network Science*, 5(4):529–550, 2017.
- 6 Paolo Boldi and Sebastiano Vigna. Axioms for centrality. *Internet Mathematics*, 10(3-4):222–262, 2014.
- 7 Phillip Bonacich. Factoring and weighting approaches to status scores and clique identification. *Journal of mathematical sociology*, 2(1):113–120, 1972.
- 8 Ulrik Brandes. *Network analysis: methodological foundations*, volume 3418. Springer Science & Business Media, 2005.
- 9 Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.
- 10 Zoltán Dezső and Albert-László Barabási. Halting viruses in scale-free networks. *Physical Review E*, 65:055103, 2002.
- 11 Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.
- 12 Manuj Garg. Axiomatic foundations of centrality in networks. *Available at SSRN 1372441*, 2009.
- 13 Per Hage and Frank Harary. Eccentricity and centrality in networks. *Social networks*, 17(1):57–63, 1995.
- 14 Aidan Hogan, Andreas Harth, Jürgen Umbrich, Sheila Kinsella, Axel Polleres, and Stefan Decker. Searching and browsing linked data with swse: The semantic web search engine. *Journal of web semantics*, 9(4):365–401, 2011.
- 15 Gábor Iván and Vince Grolmusz. When the web meets the cell: using personalized PageRank for analyzing protein interaction networks. *Bioinformatics*, 27(3):405–407, 2010.
- 16 Matthew O Jackson. *Social and economic networks*. Princeton university press, 2010.
- 17 Mitri Kitti. Axioms for centrality scoring with principal eigenvectors. *Social Choice and Welfare*, 46(3):639–653, 2016.

- 18 Dirk Koschützki, Katharina Anna Lehmann, Leon Peeters, Stefan Richter, Dagmar Tenfelde-Podehl, and Oliver Zlotowski. Centrality indices. In *Network analysis*, pages 16–61. Springer, 2005.
- 19 Jose L Martinez-Rodriguez, Aidan Hogan, and Ivan Lopez-Arevalo. Information extraction meets the semantic web: a survey. *Semantic Web*, 11(2):255–335, 2020.
- 20 Mark Newman. *Networks*. Oxford university press, 2018.
- 21 Juhani Nieminen. On the centrality in a directed graph. *Social Science Research*, 2(4):371–378, 1973.
- 22 Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
- 23 K Brooks Reid. Centrality measures in trees. In *Advances in Interdisciplinary Applied Discrete Mathematics*, pages 167–197. World Scientific, 2011.
- 24 Cristian Riveros and Jorge Salas. A family of centrality measures for graph data based on subgraphs. In Carsten Lutz and Jean Christoph Jung, editors, *ICDT*, volume 155, pages 23:1–23:18, 2020.
- 25 Gert Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581–603, 1966.
- 26 Oskar Skibski. Vitality indices are equivalent to induced game-theoretic centralities. In Zhi-Hua Zhou, editor, *IJCAI*, pages 398–404. International Joint Conferences on Artificial Intelligence Organization, 2021.
- 27 Oskar Skibski, Tomasz P. Michalak, and Talal Rahwan. Axiomatic characterization of game-theoretic centrality. *Journal of Artificial Intelligence Research*, 62:33–68, 2018.
- 28 René van den Brink and Robert P. Gilles. Measuring domination in directed networks. *Social Networks*, 22(2):141–157, 2000.
- 29 Tomasz Wąs and Oskar Skibski. Axiomatization of the PageRank centrality. In *IJCAI*, pages 3898–3904. International Joint Conferences on Artificial Intelligence Organization, 2018.
- 30 Mihalis Yannakakis. Algorithms for acyclic database schemes. In *VLDB*, volume 81, pages 82–94, 1981.