Applications of the Graph Tukey Depth

Florian Seiffarth[1], Tamás Horváth[1,2,3], and Stefan Wrobel[1,2,3]

[1]Dept. of Computer Science, University of Bonn, Bonn, Germany
[2]Fraunhofer IAIS, Schloss Birlinghoven, Sankt Augustin, Germany
[3]Fraunhofer Center for Machine Learning, Sankt Augustin, Germany
{seiffarth,horvath,wrobel}@cs.uni-bonn.de

Abstract. We study a recently introduced adaptation of Tukey depth to graphs and discuss its algorithmic properties and potential applications to mining and learning with graphs. In particular, since it is NP-hard to compute the Tukey depth of a node, as a first contribution we provide a simple heuristic based on maximal closed set separation in graphs and show empirically on different graph datasets that its approximation error is small. Our second contribution is concerned with geodesic core-periphery decompositions of graphs. We show empirically that the geodesic core of a graph consists of those nodes that have a high Tukey depth. This information allows for a parameterized deterministic definition of the geodesic core of a graph.

1 Introduction

Centrality measures are of high importance in data analysis, as they typically capture the elements' "importance" quantitatively. Of course, the meaning of *importance* depends on the choice of the particular centrality measure. Different types of centrality measures have been introduced for networks (see, e.g., [13]), including *degree centrality*, *eigenvector centrality*, *Katz centrality*, *closeness centrality*, *betweenness centrality*, *page rank*, and *hubs and authorities*. In Fig. 1 we present a graphical illustration of some of these centrality measures for some small graphs for a visual comparison.

In [2], a relatively new centrality measure, the *Tukey depth* has been introduced for graphs. The notion of Tukey depth was originally defined over finite subsets of $\mathbb{R}^d$ [5,22]. It depends only on the traces of *half-spaces* of $\mathbb{R}^d$ on the ground set, without utilizing the geometric position of the elements. This property allows for adapting it to other domains associated with a *closure system*, by using (abstract) half-spaces [3] or other types of disjoint closed sets [17]) instead of half-spaces in $\mathbb{R}^d$. For $\mathbb{R}^d$ and in other more general metric spaces [4], Tukey depth has been studied in the context of *machine learning*, in particular, object classification in [12]. In the case of learning linear classifiers, the *Tukey median*, i.e., the points with the highest Tukey depth are related to the Bayes point [6].

Regarding *graphs*, the Tukey depth of a node v of a graph is defined by the size (i.e., number of nodes) of the underlying graph subtracted by the maximum size of a *geodesically closed* set that does not contain v itself [2] (see Fig. 1 for

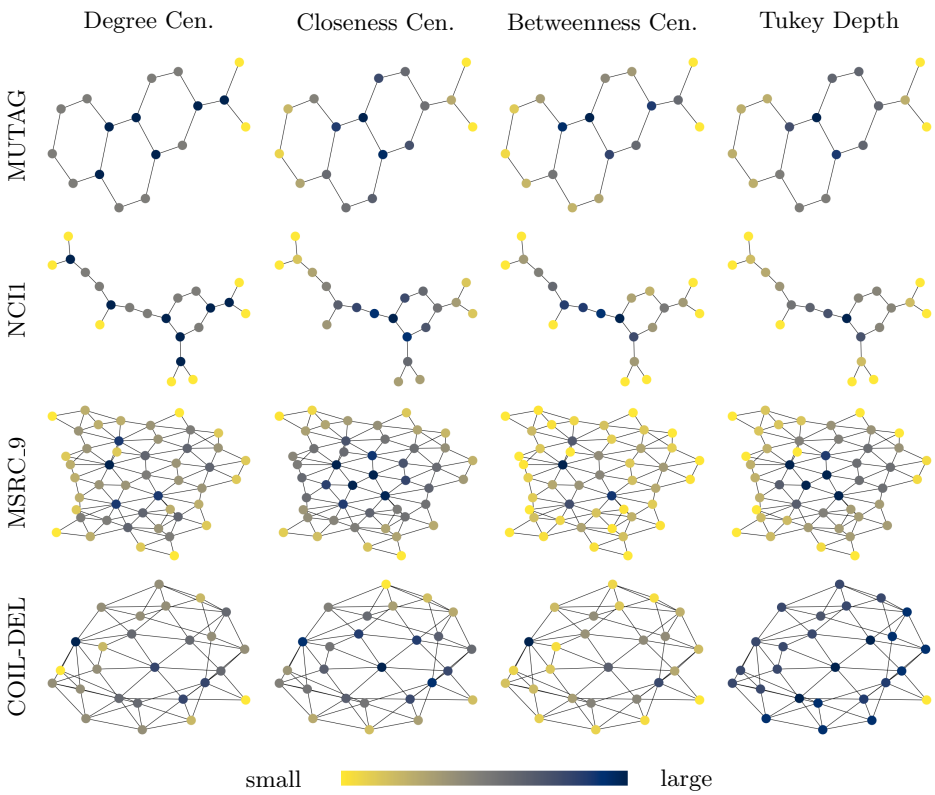


Fig. 1: *Degree Centrality*, *Closeness Centrality*, *Betweenness Centrality* and *Tukey Depth* of nodes in graphs selected from different graph datasets [11]. The centrality (resp. depth) values are *normalized* (i.e, mapped to the interval $[0, 1]$) by their maximum values in the graph. In particular, nodes of the smallest (resp. highest) centrality values are denoted by yellow (resp. blue).

an example of the Tukey depth on graphs). This is closely related to the original definition for $\mathbb{R}^d$ [5,22]. The difference is that in $\mathbb{R}^d$, a half-space is used, while maximum size geodesic closed sets are not necessarily half-spaces. Similarly to $\mathbb{R}^d$, it is NP-hard to compute the graph Tukey depth of a node [2,7].

Motivated by this negative result, one of the main contributions of this work is a *heuristic* algorithm for *approximating* graph Tukey depth. It runs in time polynomial in the size of the input graph and approximates the Tukey depth of a node with one-sided error by *overestimating* it. Our experimental results with small graphs clearly demonstrate that the approximation is close to the exact Tukey depth, by noting that for larger graphs we were not able to evaluate the approximation performance of our algorithm, as it was *not* possible to calculate the *exact* Tukey depth in practically feasible time.

Our heuristic is based on the greedy algorithm designed in [17] for solving the more general maximal closed set separation (MCSS) problem. In the particular case of graphs, the underlying closure operator is defined by the graph geodesic closure in [17]. It is therefore natural to ask the following question: *Is there a connection between graph Tukey depth and node separation with geodesically closed node sets?* We give an affirmative answer to this question by showing that the size of a *maximum* separating closed node set always depends on the Tukey depth of its nodes. In particular, for any set containing at least one node of high Tukey depth, there exists *no* large disjoint closed set.

It follows from the definition of graph Tukey depth that it is related to other concepts based on geodesic convexity. One of these notions is the recent *probabilistic* definition of geodesic *core-periphery decomposition* of graphs. It was introduced in [21] and studied in [19,21,23,24]. Our second question is concerned with the following problem: *Is there a connection between graph Tukey depth and geodesic core-periphery decompositions?* The geodesic core-periphery decomposition separates some graphs (including social networks) into a *dense* core and a *sparse* "surrounding" periphery [21] (see Fig. 3 for a visual example). While some of the graphs (e.g., Erdős-Rényi, Barabási-Albert, and Watts-Strogatz random graphs) seem to have no periphery, others (e.g., trees and fully connected graphs) seem to have no core. This behavior is not well-understood up to now. It seems that if all nodes in the graph are of high Tukey depth, then the graph contains a core, which is the entire graph. In contrast, if there is only a small set of nodes of high Tukey depth, then its core consists of those nodes (see Fig. 4 for some examples). This observation allows for a *parameterized deterministic* definition of the cores. That is, the core of a graph can be defined by those nodes that have a Tukey depth greater than a user specified threshold. Our empirical results clearly demonstrate that using the right threshold, the probabilistic definition of cores in [21] *coincides* with our deterministic one.

The rest of the paper is organized as follows. In Sec. 2 we first collect necessary notions and notations. In Sec. 3 we present our heuristic for approximating Tukey depth and evaluate it empirically on small graph datasets. Sec. 4 contains some examples which show that graph Tukey depth is strongly related to existing mining and learning algorithms on graphs that rely on graph geodesic convexity. Finally, in Sec. 5 we mention some open questions for future research.

2 Preliminaries

In this section, we collect the necessary notions and fix the notation. For a graph $G = (V, E)$, $V(G)$ and $E(G)$ denote the set V of nodes and the set E of edges, and n and m stands for $n = |V(G)|$ and $m = |E|$, respectively. Unless otherwise stated, by graphs we always mean undirected graphs without loops and multiedges. For any $u, v \in V(G)$, the (geodesic) interval $[u, v]$ is the set of *all* nodes on *all* shortest paths between u and v (see Fig. 2a for an example). A set of nodes $X \subseteq V(G)$ is called (geodesically) closed iff for all $u, v \in V(G)$, $u, v \in X$

(a) (b)

Fig. 2: (a) The interval $[u, v]$ in blue and (b) the closure $\rho(\{u, v\})$ in red.

implies $[u, v] \subseteq X$. The closure $\rho(X)$ of a set $X \subseteq V(G)$ is the smallest closed set containing X (see Fig. 2b for an example of $\rho(\{u, v\})$).

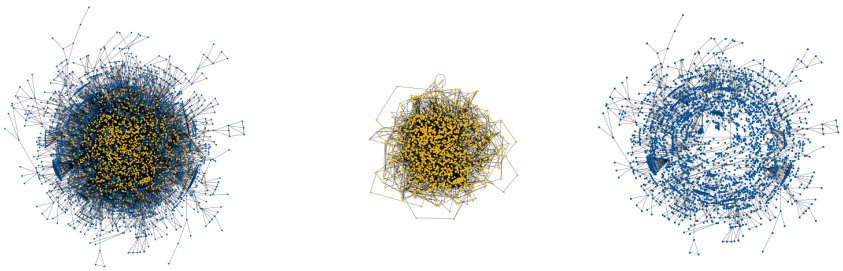
Graph Tukey Depth For a graph G the *Tukey depth* of a node $v \in V(G)$ is defined as follows [2]: Let $C \subset V(G)$ be a closed set of *maximum* cardinality such that $v \notin C$. The *Tukey depth* of v, denoted by $\mathrm{td}(v)$ is defined by $\mathrm{td}(v) = |V(G)| - |C|$. The definition implies that the larger a closed set which does not contain v, the smaller its depth is.

Geodesic Cores Up to now, *geodesic cores* [21] are defined *probabilistically* only. Informally, the geodesic core of a graph consists of those nodes which are contained in "every" geodesic closed set that is generated by a small number of random nodes. Of course, the core defined in this way can be empty, but it turns out that this is not the case for most social networks. Adapting the definition of [21] slightly, we define the geodesic core of a graph G, denoted by $\mathcal{C}$ as follows. Let $X_1, X_2, \ldots$ be a sequence of sets where each set consists of $k > 0$ nodes selected independently and uniformly at random from $V(G)$. Then $\mathcal{C} = \bigcap_{j=1}^{i} \rho(X_j)$, where i is the *smallest* integer satisfying $\bigcap_{j=1}^{i} \rho(X_j) = \bigcap_{j=1}^{i+1} \rho(X_j)$ is the core of G. Obviously, this definition is not deterministic since different choices of X_j and of k can lead to different cores. Nevertheless, the experiments in [19] with large real-world networks show that for $k \approx 10$, the core (if it exists) does *not* depend on the particular choice of the generator elements. The *core-periphery decomposition* of a graph is composed of the subgraph induced by the core nodes and that by the remaining nodes, called *periphery*. In Fig. 3 we give a visual example of the core-periphery decomposition of the CA-GrQc network [9][1].

3 Approximating the Tukey Depth

Motivated by the negative complexity result concerning the calculation of Tukey depth, in Sec. 3.1 below we first propose a *heuristic* based on the maximal closed set separation (MCSS) algorithm in [17]. It approximates Tukey depths with one-sided error. We then show experimentally on different types of *small* graphs that the results obtained by our heuristic are *fairly close* to the exact ones.

[1] This network is build by the co-authorships in the general relativity and quantum cosmology community.



(a) Entire Network (b) (Geodesic) Core (c) Periphery

Fig. 3: (a) CA-GrQc network [9], (b) its (geodesic) core, (c) its periphery. [19]

Furthermore, our algorithm is, even on small graphs, up to 200 times faster than the exact one (see Sec. 3.2). It is important to emphasize that it was not possible to calculate the exact Tukey depths for larger graphs in a feasible time.

3.1 The Heuristic

Recall that the *exact* Tukey depth of a node v is defined by $\mathrm{td}(v) := |V(G)| - |C|$, where $|C|$ is the maximum size of a closed set C not containing v. It can be computed exactly using an *integer linear program* (see [2] for the details). The computationally *hard* part of the problem is that a closed set of *maximum* size has to be found. Our heuristic addresses this problem by considering an inclusion *maximal* closed set only, instead of a maximum sized closed set. This relaxation, which distorts of course the exact value of Tukey depth, allows us to apply the efficient greedy algorithm proposed in [17] for solving the maximal closed separation problem. In what follows, for any $v \in V(G)$, $\widehat{\mathrm{td}}(v)$ denotes the approximation of $\mathrm{td}(v)$ obtained with our heuristic.

Given a graph G, the rough idea to approximate the Tukey depth of a node $v \in V(G)$ is to find an inclusion maximal geodesically closed set $C \subseteq V(G)$ with $v \notin C$. Such a set C can be found by applying the *MCSS* algorithm from [17] with v and some distinct node v' as input. The output of the algorithm consists of two node sets $H_v, H_{v'} \subseteq V(G)$ with $v \in H_v$ and $v' \in H_{v'}$ such that they are disjoint, closed, and inclusion maximal concerning these properties. That is, neither of $H_v, H_{v'}$ can properly be extended into a larger closed set without violating the disjointness. The Tukey depth can then be approximated using the sizes of H_v resp. $H_{v'}$. Given v, the result depends on the particular choice of v'. To improve the approximation quality, we therefore call the *MCSS* algorithm for each node v several times with *different* nodes $v' \neq v$.

The description of the above heuristic is given in Alg. 1. In Line 1 we initialize the Tukey depth of all nodes in G by setting them to the maximum possible value, i.e., to $|V(G)|$. We repeat the procedure described above for all nodes $v \in V(G)$ and all their neighbors $v' \in \Gamma(v)$. In this way we separate v from all neighbors

Algorithm 1: Approximation of Graph Tukey Depth

Input : graph G
Output: approximation $\widehat{\mathrm{td}}(v)$ of $\mathrm{td}(v)$ for all $v \in V(G)$

1 $\widehat{\mathrm{td}}(v) \longleftarrow |V(G)|$ for all $v \in V(G)$;
2 **for** $v \in V(G)$ **do**
3 **for** $v' \in \Gamma(v)$ **do**
4 $H_{v'}, H_v = MCSS(\{v'\}, \{v\})$;
5 **for** $x \in V(G)$ **do**
6 **if** $x \notin H_{v'}$ **then**
7 $\widehat{\mathrm{td}}(x) = \min\{\widehat{\mathrm{td}}(x), |V(G)| - |H_{v'}|\}$;
8 **if** $x \notin H_v$ **then**
9 $\widehat{\mathrm{td}}(x) = \min\{\widehat{\mathrm{td}}(x), |V(G)| - |H_v|\}$;
10 **return** $\widehat{\mathrm{td}}(v)$ for all $v \in V(G)$

v' by maximal disjoint closed sets $H_v, H_{v'}$ (see Line 4). We then update the current Tukey depth of *all* graph nodes $x \in V(G)$ by taking the minimum over the current approximation value and the new approximation which is the size of the graph subtracted by the size of the output closed set not containing x (see Line 7 and Line 9).

By construction, Alg. 1 finds only *maximal* and *not* maximum closed sets, resulting in an one-sided error in the estimate of Tukey depths. This property is formulated in the proposition below.

Proposition 1 *Alg. 1 overestimates the Tukey depth, i.e., for the output* $\widehat{\mathrm{td}}(v)$ *returned by Alg. 1 we have* $\widehat{\mathrm{td}}(v) \geq \mathrm{td}(v)$*, for all* $v \in V(G)$*.*

Regarding the runtime of Alg. 1, note that it depends on the number of *MCSS* computations in the inner loop (Lines 3–9). Iterating over all neighbors, the runtime is quadratic in the number of edges and linear in the number of nodes. This follows from the facts that we call the closure algorithm for each edge at most twice and that the closure algorithm runs in time $\mathcal{O}(m \cdot n)$ [14]. Thus, we have the following result for the total runtime of Alg. 1:

Proposition 2 *Alg. 1 outputs an upper bound of the Tukey depth for all nodes of* G *in* $\mathcal{O}(m^2 \cdot n)$ *time.*

The runtime of the approximation algorithm can be improved by considering for each node v a *fixed* number of distinct nodes v', or by considering a fixed subset $W \subseteq V(G)$, instead of the whole node set $V(G)$ in the outer loop (see Lines 2–9). It is left to further research to analyze how these changes affect the quality of the approximation performance.

3.2 Experimental Evaluation

In this section we empirically evaluate the approximation quality and runtime of Alg. 1 on datasets containing *small* graphs[2]. Regarding the approximation

[2] See https://github.com/fseiffarth/AppOfTukeyDepth for the code.

quality, we compare the results obtained by our algorithm to the exact Tukey depths computed with the algorithm in [2]. For the evaluation we consider 19 graph datasets of different types (small molecules, small graphs from bioinformatics and computer vision, and small social networks) from [11] (see columns 2–4 of Tab. 1 for the number of graphs and their average number of nodes and edges). The average size of the graphs ranges from 14 (*PTC_MM*) up to 82 (*OHSU*); their average edge numbers from 14 to 200. The reason for considering small graphs only is that the exact algorithm from [2] was unable to calculate the Tukey depth for larger graphs in less than one day (see the last two columns). For practical reasons, we removed all disconnected graphs from the original datasets, by noting that our heuristic works for disconnected graphs as well.

The results are presented in Tab. 1. It contains the approximation qualities measured in different ways (columns 5–10) and the runtime of the exact (column 11) and our heuristic algorithm (column 12). The datasets are sorted according to their absolute approximation error (column 5 of Tab. 1), i.e., the sum of all approximation errors over all nodes over all graphs in the dataset.

Regarding the absolute error, our approximation results are equal to the exact Tukey depths for 5 out of the 19 datasets, while their computation was faster by a factor of up to 100 (see row *PTC_MM*). Our algorithm has the largest absolute error of 4155 on the *COIL-DEL* graphs, by noting that this dataset consists of 3900 graphs. Hence, the error per graph is only slightly above one. Additionally, we look at the relative errors (column 6), i.e., the absolute error divided by the sum of all depths. We use this measure to validate that our algorithm performs very well, by noting that the relative errors are below $4 \cdot 10^{-3}$ for all graph datasets. The *per node error* (column 7) is the average error our algorithm makes per node, while the *per graph error* (column 8) is the error it has on average per graph. Regarding the per node error, the worst case is for the *COIL-DEL* dataset (last row) with an average error of 0.05. For the per graph error, the worst result has been obtained for the *OHSU* dataset, where the approximation overestimates the sum of all node depths by 1.65 per graph on average. This shows that our approximation algorithm performs very well, especially, if considering the averages over the datasets. Finally, we studied also the worst case approximations for nodes and graphs. In particular, the columns *Max. Node Error* resp. *Max. Graph Error* denote the maximum error of the algorithm on single nodes resp. single graphs. The results show that there is a very low error of at most 3 per node for 13 out of the 19 datasets. For three graph datasets, the maximal error per node is at most 7 and we have a maximal error between 11 and 19 in three cases. Regarding the maximum error per graph, a similar behavior can be observed by noting that except for *OHSU* and *Peking_1*, the maximum node errors and maximum graph errors are close to each other. This implies that there are only a *few* nodes with a *high* approximation error. It is an interesting question to pinpoint the properties of such nodes and graphs that are responsible for the high approximation errors. The last two columns show the runtimes of the two algorithms. Our algorithm (last column) is faster than the exact one on all of the datasets by at least one order of magnitude.

Data	Graph Number	Avg. Nodes	Avg. Edges	Error (absolute)	Error (relative)	Error per Node	Error per Graph	Max. Node Error	Max. Graph Error	Exact Runtime (s)	Approx. Runtime (s)
BZR	405	35.75	38.36	0	0	0	0	0	0	56.60	1.04
PTC_MM	336	13.97	14.32	0	0	0	0	0	0	21.09	0.21
COX2	467	41.22	43.45	0	0	0	0	0	0	76.10	1.27
Cuneiform	267	21.27	44.80	0	0	0	0	0	0	2.00	0.61
DHFR	756	42.43	44.54	0	0	0	0	0	0	266.33	3.19
PTC_FR	351	14.56	15.00	1	4.50e-05	1.96e-04	2.85e-03	1	1	23.81	0.25
PTC_FM	349	14.11	14.48	1	4.80e-05	2.03e-04	2.86e-03	1	1	20.64	0.22
MUTAG	188	17.93	19.79	1	6.50e-05	2.97e-04	5.32e-03	1	1	3.01	0.09
PTC_MR	344	14.29	14.69	2	9.20e-05	4.07e-04	5.81e-03	1	1	23.29	0.23
KKI	83	26.96	48.42	12	1.01e-03	5.36e-03	1.45e-01	2	4	40.45	2.43
IMDB-BINARY	1000	19.77	96.53	19	4.37e-04	9.61e-04	1.90e-02	1	3	723.07	113.14
NCI1	3530	29.27	31.88	34	5.20e-05	3.29e-04	9.63e-03	6	8	1194.63	7.76
Peking_1	85	39.31	77.35	40	1.30e-03	1.20e-02	4.71e-01	3	10	4761.33	48.08
MSRC_21C	209	40.28	96.60	86	1.28e-03	1.02e-02	4.11e-01	12	12	51.78	3.06
MSRC_9	221	40.58	97.94	89	1.21e-03	9.92e-03	4.03e-01	7	8	49.33	2.92
OHSU	79	82.01	199.66	130	8.54e-04	2.01e-02	1.65e+00	2	13	42887.32	235.40
ENZYMES	569	31.68	61.44	307	1.68e-03	1.70e-02	5.40e-01	7	10	933.79	8.25
MSRC_21	563	77.52	198.32	877	1.39e-03	2.01e-02	1.56e+00	19	25	3679.24	58.98
COIL-DEL	3900	21.54	54.24	4155	4.05e-03	4.95e-02	1.07e+00	11	17	2242.05	43.00

Table 1: Graph data of different sizes selected from [11]. Disconnected graphs are removed from the original datasets. The columns regarding the approximation quality denote the following. *Error (absolute)* denotes the overall error on the dataset, *Error (relative)* denotes the relative error regarding the depths, *Error per Node* denotes the average error per node, *Error per Graph* denotes the average error per graph, *Max. Node Error* denotes the maximum error for a node and *Max. Graph Error* denotes the maximum error on a graph. The last two columns show the runtimes of the exact and approximation algorithm in seconds.

In summary, the results of the evaluation of Alg. 1 clearly show that our heuristic performs well in approximating the graph Tukey depth. It is faster (up to 200 times) than the exact algorithm, even on these small graph datasets. Regarding larger graphs, this gap in runtime will increase because of the exponential runtime of the exact algorithm. Additionally, the very small relative errors (at most $4 \cdot 10^{-3}$), the average errors (at most 1.65 per graph), and also the worst case errors show that the algorithm can be used for further applications based on the Tukey depth (see Sec. 4).

4 Applications to Mining and Learning in Graphs

This section deals with the connection of *Tukey depth* to *node separability* and to *geodesic core-periphery decompositions*. We first state three important properties of Tukey depth. In particular, Proposition 3 clarifies the role of Tukey depth in the context of geodesic closed sets. Propositions 4 and 5 are from [2].

Proposition 3 *Let G be a graph, $v \in V(G)$ with $\mathrm{td}(v) = n - c$, and $C \subseteq V(G)$ a geodesically closed node set with $|C| > c$. Then $v \in C$.*

Proposition 4 *Let G be a graph, $X \subseteq V(G)$, and C be the geodesic closure of X. Then the Tukey depth is a quasi-concave function, i.e., for all $c \in C$ we have $td(c) \geq \min\{td(x) : x \in X\}$.*

Proposition 5 *Let G be a graph, $k \in \mathbb{N}$, and $X = \{v \in V(G) : \mathrm{td}(v) \geq k\}$. Then X is geodesically closed.*

To underline the importance of these three statements, we give two examples that show how they influence existing concepts based on geodesic closures.

Example 1: Node Classification and Active Learning In [1,17,18,20], disjoint half-spaces and closed sets are used for binary classification in closure systems, for node classification, and active learning in graphs using geodesic convexity. Given the Tukey depth $\mathrm{td}(v)$ of a node v, Proposition 3 immediately implies that a separating half-space or closed set not containing v cannot have a cardinality greater than $n - \mathrm{td}(v)$. Thus, for nodes of *high* Tukey depth there is *no* large geodesic closed set *not* containing them. Hence, Proposition 3 implies a nice theoretical connection between Tukey depth and the maximum size of separating half-spaces and closed sets. Using approximate Tukey depths, the predictive performance of all the above methods can possibly be improved.

Example 2: Geodesic core-periphery decomposition The geodesic core-periphery decomposition of graphs was analyzed in [19,21,23]. In particular, it was found in [21,23] that many social networks consist of a dense geodesic core "surrounded" by a periphery (see Fig. 3 for an example). While some graphs, especially tree-like graphs, seem to have no core, others, such as graphs sampled from random models like Erdős-Rényi, Barabási-Albert and Watts-Strogatz seem to have no

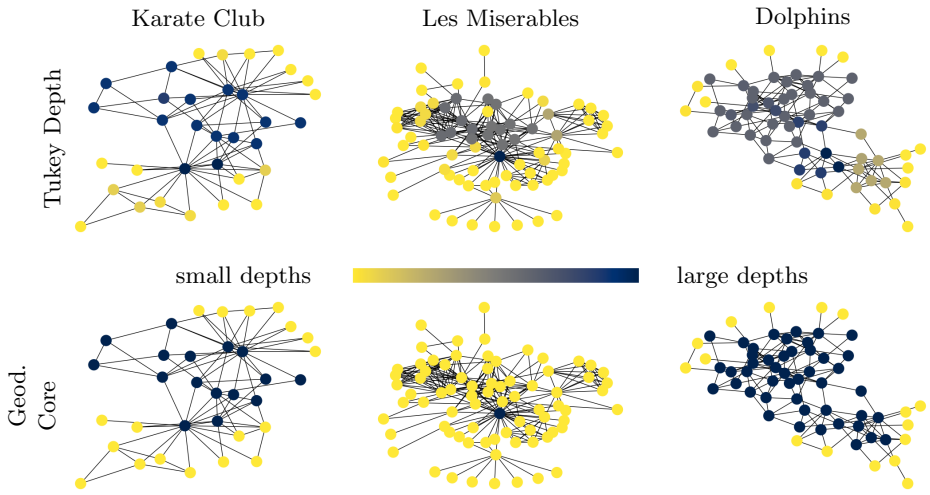


Fig. 4: Tukey depth (top) vs. geodesic core-periphery decomposition (bottom) for the Karate Club [25], Les Miserables character [8], and Dolphins social networks [10]. For the different Tukey depths we use sequential colors. Core and periphery nodes are denoted by blue and yellow, respectively.

periphery. Moreover, the closure of a small number of randomly chosen graph nodes (≈ 10) always contains the geodesic core (if it exists). Furthermore, if the nodes are sampled from the geodesic core only, then the closure of the nodes is the geodesic core itself. If we compute the closure of, say, 10 randomly chosen nodes from the entire network (Fig. 3a), then the closure always contains the core (orange nodes in Fig. 3b). If all random nodes belong to the core (orange nodes in Fig. 3a), then their closure is the core itself. The above statements explain this behavior. Using that the core is always contained in the closure of a small number of randomly chosen nodes, from Proposition 4 it follows that the nodes in the core are those with the highest Tukey depths. Moreover, the quasi-concaveness implies that if the core is generated by a few nodes from the core, then the core nodes must have a very close Tukey depth. Finally, using Proposition 5, we have that the set of nodes in a graph with a Tukey depth above some threshold is always closed; cores arise as a special case of this property. These three properties motivate the following *deterministic* definition of geodesic cores:

Definition 1 *The k-geodesic core of a graph G is defined by*

$$C_k := \{v \in V(G) : \mathrm{td}(v) \geq k\} \ .$$

To empirically confirm our claim that the core contains the nodes with the highest Tukey depths, consider the three graphs in Fig. 4. For each graph, we computed the exact Tukey depths (top) and their geodesic cores (bottom). For the Karate Club network (left) considered also in [2], the core exactly matches

the nodes of the highest Tukey depth. Furthermore, there is not much fluctuation in the depths of the core nodes. In fact, all nodes of Tukey depth of at most 3 belong to the periphery and all nodes of Tukey depth 19 or 21 to the core. In the case of the Les Miserables character network (middle), there is only a single node with a very high Tukey depth of 57, surrounded by nodes of depth less than 35. In this case, the core algorithm returns only the node with the highest Tukey depth, showing that the graph Tukey depth can possibly be used to improve core-periphery decomposition. This is also the case for the Dolphin community graph (right), where the core consists of nodes with Tukey depth greater than 2, while all nodes in the periphery have a Tukey depth of at most 2.

5 Concluding Remarks

Our results indicate that graph Tukey depth is an interesting and promising concept for mining and learning with graphs. The study of the relationship of graph Tukey depth to other node centrality measures is an interesting question for further research (see Fig. 1). For example, while the centroid(s) in trees [15,16] are exactly the nodes with the highest Tukey depth, this is not necessarily the case for more general graphs beyond trees.

Another important issue is that the semantics of graph Tukey depth must be understood better. For example what are the properties of the nodes with the highest depth (cf. the def. of Tukey-median in $\mathbb{R}^d$)? We have empirically demonstrated that graph Tukey depth can closely be approximated for small graphs. It is a question of whether this result holds for (very) large graphs as well. To answer this question, the scalability of our approximation algorithm should be improved on the one hand. On the other hand, one needs (possibly tight) theoretical upper bounds on graph Tukey depths. Another interesting question is to identify graph classes for which our approximation is always exact. While this is the case for trees, it is unclear whether it holds for outerplanar graphs as well. We believe that this question can be answered affirmatively by using the techniques from [19]. As shown in the paper, graph Tukey depth "naturally" connects different existing concepts based on geodesically closed node sets; examples include the deterministic definition of k-geodesic cores. This implies that using our fast core approximation algorithm [19], we can closely approximate the set of nodes with the highest Tukey depth.

References

1. de Araújo, P.H.M., Campêlo, M.B., Corrêa, R.C., Labbé, M.: The geodesic classification problem on graphs. In: Proceedings of the tenth Latin and American Algorithms, Graphs and Optimization Symposium. Electronic Notes in Theoretical Computer Science, vol. 346, pp. 65–76. Elsevier (2019)
2. Cerdeira, J.O., Silva, P.C.: A centrality notion for graphs based on tukey depth. Appl. Math. Comput. **409**, 126409 (2021)
3. Chepoi, V.: Separation of two convex sets in convexity structures. J. of Geometry **50**(1), 30–51 (1994)

4. Dai, X., Lopez-Pintado, S., for the Alzheimer's Disease Neuroimaging Initiative: Tukey's depth for object data. Journal of the American Statistical Association pp. 1–13 (2022)
5. Donoho, D.L., Gasko, M.: Breakdown Properties of Location Estimates Based on Halfspace Depth and Projected Outlyingness. The Annals of Statistics **20**(4), 1803 – 1827 (1992)
6. Gilad-Bachrach, R., Navot, A., Tishby, N.: Bayes and tukey meet at the center point. In: Learning Theory. pp. 549–563. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
7. Johnson, D., Preparata, F.: The densest hemisphere problem. Theoretical Computer Science **6**(1), 93–107 (1978)
8. Knuth, D.E.: The Stanford GraphBase: A Platform for Combinatorial Computing. Association for Computing Machinery (1993)
9. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data` (Jun 2014)
10. Lusseau, D.: The emergent properties of a dolphin social network. Proceedings of the Royal Society of London. Series B: Biological Sciences **270** (2003)
11. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: Tu-dataset: A collection of benchmark datasets for learning with graphs. In: ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020) (2020), `www.graphlearning.io`
12. Mozharovskyi, P.: Contributions to depth-based classification and computation of the Tukey depth. Ph.D. thesis, University of Cologne (2015)
13. Newman, M.: Networks: An Introduction. Oxford University Press, Inc., USA (2010)
14. Pelayo, I.M.: Geodesic Convexity in Graphs. Springer New York (2013)
15. Piotrowski, W.: A generalization of branch weight centroids. Applicationes Mathematicae **19**(3-4), 541–545 (1987)
16. Piotrowski, W., Syslo, M.M.: Some properties of graph centroids. Ann. Oper. Res. **33**(3), 227–236 (1991)
17. Seiffarth, F., Horváth, T., Wrobel, S.: Maximal closed set and half-space separations in finite closure systems. In: ECML PKDD 2019. LNCS, vol. 11906, pp. 21–37. Springer (2019)
18. Seiffarth, F., Horváth, T., Wrobel, S.: Maximum margin separations in finite closure systems. In: ECML PKDD 2020. Lecture Notes in Computer Science, vol. 12457, pp. 3–18. Springer (2020)
19. Seiffarth, F., Horváth, T., Wrobel, S.: A fast heuristic for computing geodesic cores in large networks (2022), `https://arxiv.org/abs/2206.07350`
20. Thiessen, M., Gärtner, T.: Active learning of convex halfspaces on graphs. In: Advances in Neural Information Processing Systems (2021)
21. Tilen, M., Šubelj, L.: Convexity in complex networks. Network Science **6**(2), 176–203 (2018)
22. Tukey, J.W.: Mathematics and the picturing of data. Proceedings of the International Congress of Mathematicians, Vancouver, 1975 **2**, 523–531 (1975)
23. Šubelj, L.: Convex skeletons of complex networks. Journal of The Royal Society Interface **15**(145), 20180422 (2018)
24. Šubelj, L., Fiala, D., Ciglaric, T., Kronegger, L.: Convexity in scientific collaboration networks. Journal of Informetrics **13**(1), 10–31 (2019)
25. Zachary, W.W.: An information flow model for conflict and fission in small groups. Journal of Anthropological Research **33**(4), 452–473 (1977)