
Book Section:
Almulhim, F, Thwaites, PA orcid.org/0000-0001-9700-2245 and Taylor, CC orcid.org/0000-0003-0181-1094 (2019) A New Approach to Measuring Distances in Dense Graphs. In: Machine Learning, Optimization and Data Science. Lecture Notes in Computer Science . Springer, Cham , pp. 204-216. ISBN 978-3-030-13708-3

A new approach to measuring distances in dense graphs*

Fatimah A Almulhim[1,2], Peter A Thwaites[1], and Charles C Taylor[1]

[1] University of Leeds, Leeds LS2 9JT, UK mmfa@leeds.ac.uk
[2] Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia

Abstract. The problem of computing distances and shortest paths between vertices in graphs is one of the fundamental issues in graph theory. It is of great importance in many different applications, for example, transportation, and social network analysis. However, efficient shortest distance algorithms are still desired in many disciplines. Basically, the majority of dense graphs have ties between the shortest distances. Therefore, we consider a different approach and introduce a new measure to solve all-pairs shortest paths for undirected and unweighted graphs. This measures the shortest distance between any two vertices by considering the length and the number of all possible paths between them. The main aim of this new approach is to break the ties between equal shortest paths SP, which can be obtained by the Breadth-first search algorithm (BFS), and distinguish meaningfully between these equal distances. Moreover, using the new measure in clustering produces higher quality results compared with SP. In our study, we apply two different clustering techniques: hierarchical clustering and K-means clustering, with four different graph models, and for a various number of clusters. We compare the results using a modularity function to check the quality of our clustering results.

Keywords: Network, adjacency matrix, K-means clustering, hierarchical clustering, modularity function.

1 Introduction

The problem of computing distances and shortest paths between vertices in graphs is one of the most fundamental and well-studied problems in graph theory. The shortest path means the minimum path length between any pair of vertices in a graph, and in the case of directed graphs, there are source and destination vertices which determine the direction of the path. The shortest path is of great importance in many different applications, which has induced researchers to produce different measures to match their applications' purposes and graph types, for example, social network analysis, transportation, and computer science. And due to this variety of applications, there are many studies on graph types. For example, the graph can be either static with a fixed number of vertices and edges or dynamic which updates in the graph's structure by adding

* Supported by Saudi Arabian Cultural Bureau in London.

or deleting vertices and edges or changing the position of edges. The edges of a graph can be directed or undirected and can have either a positive or negative weight. There is no one universal approach to solve the shortest path that could be suitable for all different graph models.

The survey by [9] classifies the shortest-path algorithms into two groups: (1) single source shortest-path (SSSP) algorithms, which calculate the shortest path from a source vertex to other vertices in the graph based on the adjacency list representation, and (2) all-pair shortest path (APSP) algorithms, which calculate the shortest path between all pairs of vertices in the graph based on the adjacency matrix representation. In their survey, they presented a taxonomy of multiple levels of shortest path algorithms as a useful tool for the researcher to understand the shortest path categories, and to guide them to suitable techniques, which depend on the application. They also mention some challenges and solutions in each group of algorithms.

The survey by [12] reviews the shortest path algorithms on static graphs that produce exact results for the APSP problems for both weighted and unweighted graphs as well as dense and sparse graphs. He also represented some studies on APSP for restricted families of graphs, such as interval graphs that determine an interval for each vertex with its neighbours.

[13] presented a survey of APSP and SSSP for weighted and unweighted graphs as well as directed and undirected graphs.

The different studies in the literature present different methods to capture the SP in graphs. Although SP results are consistent, the difference is in the methods or in the time taken. However, an efficient shortest distance algorithm is still desired in many disciplines, we think about it differently, and introduce a new measure to solve APSP for undirected and unweighted graphs. The new algorithm has a unique feature: the ability to distinguish between equal SP distances. More precisely, whereas SP is a positive integer, the new measure breaks up the equal integers into rational number. This idea and its framework is presented in Section 2. In Section 3, we have a short discussion about graph clustering. In particular, we explain the hierarchical clustering method (HC) and K-means clustering algorithm. We also describe the modularity function as a useful measure of clusters quality. A simulation study with four different graph models is described in Section 4 to compare the clustering results between the proposed distance and SP. In Section 5, we consider a real data example from the Facebook network and we conclude in Section 6 with a discussion of our approach.

2 A new distance in graphs (Breaking Ties Distance - BTD)

Most of the straightforward and essential approaches that solve the shortest path problem use the tree search idea. They start from the source vertex (root) and pass along the branches to reach the destination. In the case of undirected, unweighted graphs, this may be done by following the Breadth-First Search (BFS)

algorithm from each vertex to the rest of vertices by counting the number of edges of the shortest path. In recent years, there have been many improvements in shortest paths algorithms, which usually focus on decreasing the time complexity of the essential algorithms [9].

In this study, we focus on undirected, unweighted dense graph models. As the majority of dense graphs have ties between the shortest distances, these distances appear as equal shortest distances. However, these ties are real obstacles in several applications, for example, in clustering of vertices; when we allocate a vertex to the nearest cluster's centroid and then find a tie in the shortest distances between the vertex and a couple of centroids. Therefore, due to this issue, we consider a new way of measuring distances in graphs, which allows for breaking of ties. Whereas the majority of shortest path measures produce integer SP lengths, our distance metric produces real values for the distances. The proposed distance measure is not a transient thought; rather, it is a result of cumulative work and trials of various updated distance metrics until the aim is achieved.

During several experiments with these measures, we determined the important and effective parameters that cause these ties in graphs: the vertex degree, graph diameter and the plurality of shortest paths between a pair of nodes. Our concept is inspired by the general principle: 'more relations, more strength'. We believe that a pair of nodes joined together by multiple equal shortest paths should be thought of as closer than a pair of nodes joined by a single path of the same length. As a simple example from social networks, any two vertices (two persons) connected by three relations (such as gender, neighbourhood, and school) appear to be more strongly linked (closer) than two vertices connected by one relation. Based on this assumption, we have carried out a lot of experiments to find a distance metric formula that combines all available paths between any pair of vertices. However, the resulting distances do not refer to existing routes in the graph, in contrast, the new distance measure produces distances between vertices which often distinguish between equal shortest paths. Our measure starts with a simple version of distance and then undergoes several updates until reaching its final version. The breaking-ties distance by BTD satisfies some conditions which make it suitable to measure distances in all graph models in this study:

Let $G(V, E)$ be an undirected, unweighted graph with vertices V (nodes) and edges E , then:

- BTD is symmetric, i.e. $d_{\mathrm{BTD}}(v_i, v_j) = d_{\mathrm{BTD}}(v_j, v_i)$ for $v_i, v_j \in V$.
- BTD preserves the shortest paths order, i.e. if $d_{\mathrm{SP}}(v_i, v_j) < d_{\mathrm{SP}}(v_k, v_s)$, then BTD satisfies $d_{\mathrm{BTD}}(v_i, v_j) < d_{\mathrm{BTD}}(v_k, v_s)$. This is evident from the examples in Figure1.
- BTD satisfies the triangle inequality $d_{\mathrm{BTD}}(v_i, v_j) \leq d_{\mathrm{BTD}}(v_i, v_s) + d_{\mathrm{BTD}}(v_s, v_j)$ as does SP.

We define a similarity matrix S_{ij}:

$$S_{ij} = \sum_{r=1}^{diam} \frac{(A^r)_{ij}}{(2\max(A^r))^r}, \tag{1}$$

where:

- $A_{(n\times n)}$ is the adjacency matrix, which has entries $0, 1$ with 0 in the diagonal, and n is the number of nodes.
- $diam$ is the diameter, the longest SP in the graph.
- $\max(A^r)$ is the largest element in A^r.
- A^r is the usual matrix multiplication.

Since $(A^r)_{ij} \leq \max(A^r)^r_{ij}$, with equality when $r = 1$ and in the case of $r > 1$, S_{ij} is always less that < 1.

Now define the dissimilarity matrix by:

$$D_{ij} = -\log(S_{ij}), \quad i \neq j \quad \text{and} \quad D_{ii} = 0, \tag{2}$$

We carry out many experiments which compare BTD and SP on different random graphs according to four different graph models from [4]. Model ER is the Erdős-Rényi graph model, model WS is Watts-Strogatz graph model, model BA is Barabási-Albert graph model, and model FF is Forest-fire graph model. The experiments show that BTD produces distances which follows the SP order without any overlap between distances.

In Figure 1 (first row), all graph models show a spreading in the BTD in each SP group. In the BA model, the spreading is limited compared with other models due to the tree graph structure which has a lack of ties between nodes (the range of the distances which correspond to SP= 2 is equal 10^{-3}).

From this experiment, we believe that BTD succeeds in breaking ties between equal SP especially in dense graphs. In the next section, we compare the performance of BTD with SP in graph clustering and show that BTD produces higher quality results compared with SP.

3 Graph clustering

In graph theory, the clustering idea is to divide the nodes V into different groups, each group's nodes share similar features. These groups are called clusters, and in some cases, there are some constraints on the number or size of the clusters.

In real networks, these groups are sometimes called communities and appear naturally due to the real network's structure. This is because, first, the general feature of a real graph structure is the inhomogeneities in the edges distributions, and second, the nodes degree distribution (node degree is the number of edges which connect the node with the other nodes in a graph) often follows the power law distribution. This gives the majority of the low degree nodes a higher tendency to connect with large degree nodes which lead to the creation of some communities in real networks. These communities may be visible in some small real networks.

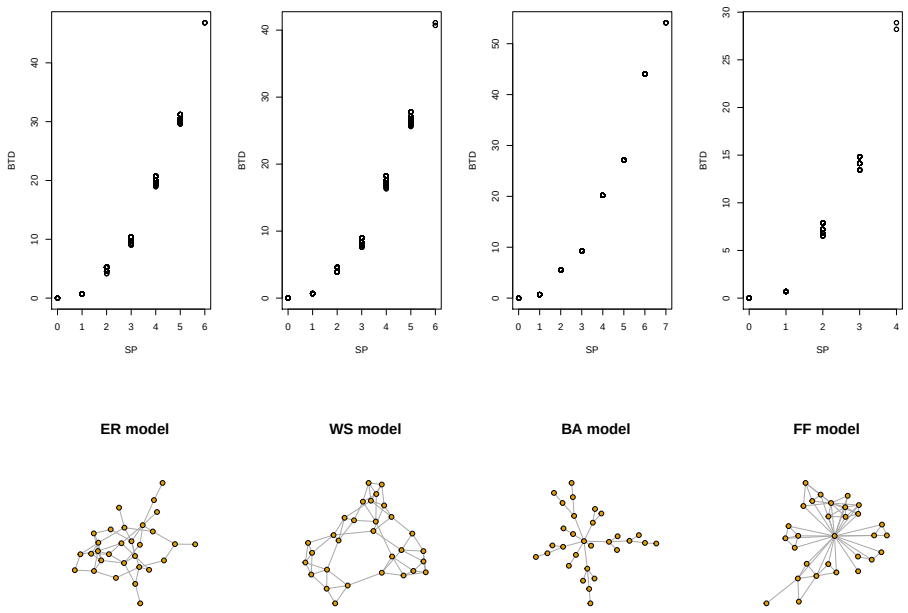


Fig. 1: Examples of graph models with $n = 30$, and the corresponding plots illustrate the breaking of ties SP distance.

In graph theory, there are different methods designed to cluster graphs, called graph partitioning methods. The main idea of these methods is to divide the graph's nodes into K clusters of predefined equal size and keep the number of edges between clusters minimal. These methods grew in pure mathematics among researchers who were interested in graph clustering [6]. Although these methods are simple and fast, they are still not a preferred tool to detect communities in graphs because of the preliminary assumption: equal groups size, which is considered as a drawback of this tool. Therefore, most of the traditional clustering methods which relax this condition are accepted in graph theory, such as hierarchical clustering, partitional clustering, and spectral clustering [5]. In the next sections, we apply BTD in hierarchical clustering and K-means algorithm.

3.1 Hierarchical clustering- HC

The Hierarchical clustering (HC) method is one of the most popular methods in graph theory, it represents the clusters usually as a dendrogram. The leaves correspond to the graph nodes, and the root joins all nodes. It can be done by

the agglomerative technique, which assigns each node in a separate cluster, and merges clusters to end with all nodes in one cluster. Alternatively, by the divisive technique, which initially puts all the nodes into one cluster, then divides it into sub-clusters. It continues division until finding a desirable structure.

The HC method has a special feature distinguishing it from the rest of clustering methods in that it produces multi-level clustering. Each level produces different clusters and each higher level cut produces of a subset of clusters from the lower level structure. This feature makes HC common in graph clustering of real networks as they have a hierarchical structure of communities, for example, social, biology, marketing networks, etc [1].

We use one of the HC techniques in association with our BTD metric, and as most of the work in literature is based on the agglomerative strategy, we choose it in our study. However, as the clusters merge from the lower level, there are different criteria to merge the clusters, each one estimates the similarity between clusters in a different way, for example, single, complete, and average linkage.

In this study, we apply agglomerative hierarchical clustering with complete linkage on four different graph models and show a comparison between both distances: SP and BTD. The results are discussed in Section 4.

3.2 K-means clustering

An alternative clustering method is K-means clustering algorithm, which is one of the oldest methods in cluster analysis. Although its first appearance was in the 1950's, it is still one of the most commonly used methods. Also, it has a rich history in the literature as it is applied in various scientific areas. Three essential ingredients are required in K-means algorithm: the number of clusters K, initial centroids $\{c_1, c_2, \ldots, c_K\}$ and a distance metric [7]. Given a graph $G(V, E)$, algorithm 1 below summarizes the steps:

Algorithm 1 K-means clustering $G(V, E)$

1: Compute the $(n \times n)$ distance matrix based on BTD.
2: Select K nodes randomly as initial centroids $v_1^*, \ldots, v_K^*$.
3: Allocate each node v_i to cluster C_k, where $k = arg\min_j d(v_i, v_j^*)$
4: **repeat**
5: Allocate each node v_i to C_k if $k = arg\min_l \frac{\sum_{v_j \in C_l} d(v_i, v_j)}{|C_l|}$
6: **until** convergence criterion is met no change in clusters members.

Starting the algorithm by computing the distance matrix BTD, then choose K initial centroids randomly. To form the clusters, we assign each node v_i to cluster k which satisfies the condition in 3. After the first allocation, we repeat step 5 until allocating all nodes to their fitted clusters. This differs from the original K-means algorithm where the points have coordinates and it is possible to calculate centroids in each iteration. In graphs, the nodes do not have coordinates, and in this case, we can not recompute the centroids in each iteration.

The most crucial point in K-means algorithm that can affect its performance is the selection of the initial centroids. It is noticeable that the random selection

of initial starts often leads to very different clustering solutions. Therefore, K-means can only converge to a local minima, and this problem increases if the dataset structure does not have natural clusters. At most, we can repeat the algorithm for different sets of initial centroids, and either evaluate the results by subjective choice (in small data sets it may be possible to choose the solution which has obvious clusters by eye) or choose the cluster's solution which has a minimum squared error between the nodes. Given a graph of size n, $V = \{v_1, v_2, \ldots, v_n\}$, and K-means clustering result $C = \{C_1, C_2, \ldots, C_K\}$, the sum of squared errors (SSE) is given by:

$$SSE(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{v_i, v_j \in C_k} d(v_i, v_j)^2, \tag{3}$$

where K is the number of clusters.

In our study, we did a simulation study of clustering four different graph models using the K-means clustering algorithm, the goal of this simulation was to compare the performance of both distances BTD and SP in graphs. The results are presented in section 4.

3.3 Modularity function

The evaluation of the quality of a cluster is one of the most critical tasks in cluster analysis. As one of the clustering goals is exploring the latent structure of a graph, high-quality clustering result could describe the communities in the underlying graph. However, [2] argued that there is no single unique measurement to check the clusters quality, and in case of graphs which can be easily visualized by the researcher, the evaluation could be subjective.

One of the most popular quality functions for measuring the goodness of network partitions is the modularity function, introduced by [10]. The idea of this approach is that most random networks do not have clear communities in the graph structure. So, It assesses the partitions quality based on the difference between the arrangement of the edges within clusters in graphs, and the random distribution of these edges between nodes in case of no community structure. It can be either positive or negative, and we are looking for divisions with high modularity as a sign of proper partitions. It can be written as

$$Q = \frac{1}{4m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j),$$

where k_i is the degree of the vertex v_i, $m = |E|$ is the total number of edges in the graph, $\frac{k_i k_j}{2m}$ is the expected number of edges between vertices i and j if edges are placed at random (null model), and the function $\delta(c_i, c_j)$ is equal to 1 if i and j are in the same group and 0 otherwise.

In our study, we use the modularity measure as an optimization function given K of the quality of our clustering results. The reasons behind our choice are that the modularity function is widely used by most of the academic researchers

in cluster analysis as the best measure of the goodness of partitions [14, 3, 6]. Also, the modularity is quite a simple tool and is faster than most of the available quality measures even for large and sparse networks [11].

4 Simulation study

The first simulation study is HC of four different graph models, the goal of this simulation is to compare the clustering results between the SP and BTD. For each model, we simulate 1000 graphs; each graph has size 100. Then, apply HC (complete link) with the number of clusters $K = 2, 3, \ldots, 10$. The Q results in Figure 2 show that BTD always exceeds SP results in all graph models, and this is an evidence of the efficiency of BTD in hierarchical graph clustering.

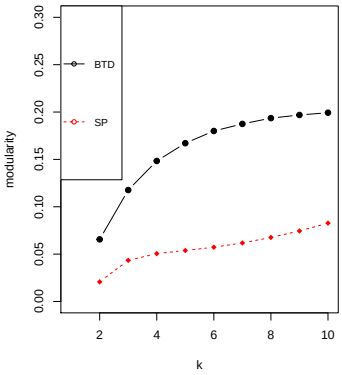
The second simulation study is K-means clustering with the same graph models to compare the performance of BTD and SP by the modularity function Q. For each model, we simulate 50 graphs of size 200. For each graph, we choose 100 different sets of random initial centroids in each of cases $k = 5, 7$. Table 1 shows the simulation results for the four models. In each simulation, we calculate the modularity measure for each of the clustering results for all 100 random initial centroids and choose the maximum modularity over these 100 modularities. So, over 50 simulations, we obtain 50 modularity measures. In Table 1, max, min, and avg correspond to maximum, minimum and average values over the 50 modularity measures. Avg Itr is the average iteration number of the K-means algorithm over 50 simulations. The avg time is the average time taken over all 50 simulations. All these measures are calculated for both cases of distances: BTD and SP. Length 1 is the average number of BTD clustering results which have modularity measures bigger than the maximum modularity measure of SP clustering results for all 50 simulations. Length 2 is the average number of SP clustering results which have modularity measures less than the minimum modularity measure of BTD results over all 50 simulations. Table 1 shows that for the ER model, WS model, and FF model, the BTD produces slightly higher quality results when compared with SP using the modularity function. In the ER model, length 1 and length 2 are higher than the other models due to lots of ties in the graph structures. In the BA model, the results look similar between BTD and SP because the graph model has a tree structure which has a lack of the ties between nodes.

Also, we apply a paired t-test on the simulation results to check if the mean difference between BTD and SP results is zero:

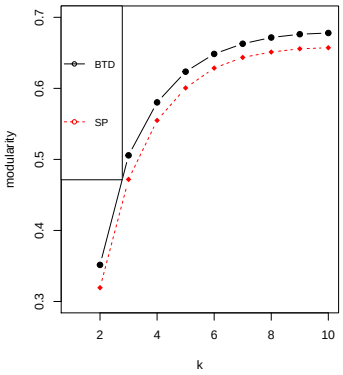
$$H_0 : \max(Q_{\mathrm{BTD}}) = \max(Q_{\mathrm{SP}})$$

$$H_1 : \max(Q_{\mathrm{BTD}}) \neq \max(Q_{\mathrm{SP}}),$$

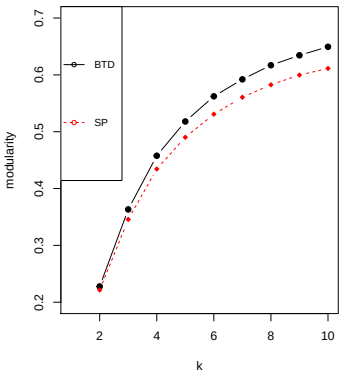
where the max is taken over the 100 random starts. The p-values of the paired test are less than 0.05 for ER, WS, and FF models, which is a significant sign of the difference between both distances results. Figure 3 illustrates the box plots of the differences between maximum/minimum modularity values between



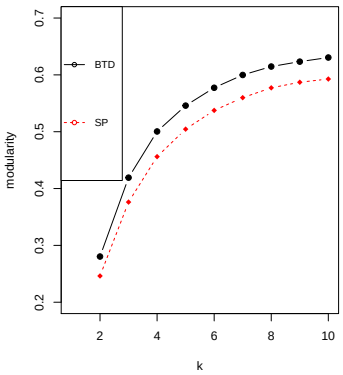
(a) ER model



(b) WS model



(c) BA model



(d) FF model

Fig. 2: Scatter plot of the mean of the modularity Q (vertical axes) of HC (complete linkage) using BTD (black line) and SP (red line) over 1000 simulations of each graph model, with different number of clusters $K = 2, 3, \ldots, 10$ (horizontal axes).

BTD and SP over 50 simulations; each figure corresponds to one model and one number of clusters. We can see that for all graph models except the BA model, most of the differences appeared higher than zero in most simulations. This is evidence that BTD generally produces higher cluster quality than SP.

Table 1: Table of statistics measurements to compare between the efficiency of BTD and SP over four different graph models with K-means algorithm for $K = 5, 7$.

	ER model				WS model			
	$k = 5$		$k = 7$		$k = 5$		$k = 7$	
Distance	BTD	SP	BTD	SP	BTD	SP	BTD	SP
Max	0.35	0.33	0.35	0.33	0.70	0.70	0.75	0.73
Min	0.31	0.29	0.31	0.30	0.64	0.63	0.68	0.66
Avg	0.33	0.32	0.33	0.31	0.67	0.66	0.70	0.69
Avg Itr	13.6	12	13.06	13	12.24	13.24	13.24	12.74
Avg time	13.3	12	13.11	11.91	13.81	13.14	14.39	13.28
length 1	13.4		21.46		3.24		6.24	
length 2	19.16		35.02		4.46		6.76	

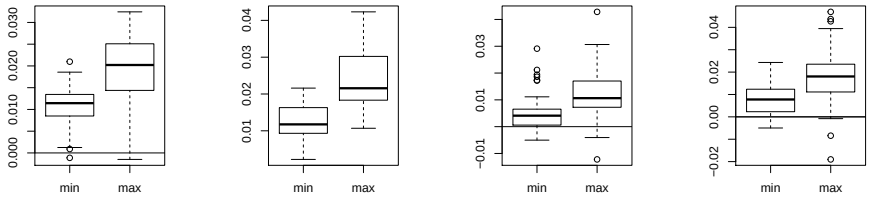
	BA model				FF model			
	$K = 5$		$K = 7$		$K = 5$		$K = 7$	
Distance	BTD	SP	BTD	SP	BTD	SP	BTD	SP
Max	0.78	0.78	0.82	0.82	0.57	0.55	0.58	0.56
Min	0.67	0.68	0.77	0.77	0.32	0.32	0.26	0.27
Avg	0.75	0.75	0.80	0.80	0.46	0.44	0.45	0.43
Avg Itr	6.5	6.4	6.7	6.6	10.44	8.72	12.04	8.74
Avg time	7.6	7.3	8.7	8.3	1.35	1.39	1.57	1.38
length 1	0.42		0.38		3.34		3.38	
length 2	0.54		0.48		2.96		3.38	

From Table 1 and Figure 3, we conclude that for dense graphs, the BTD produces higher quality clusters than the SP when assessed by the modularity function Q. This makes the BTD a more preferable distance measure in dense graphs than the SP. Note that because the difficulties in covering all parameter settings and all simulation parameters: number of simulations, graph size, the number of clusters and the models parameters are subjective choices.

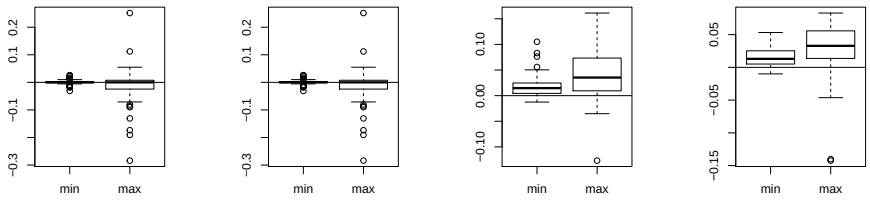
5 Facebook example [8]

Our Facebook Network dataset is a combination of 10 ego-networks which consist of 4,039 users (nodes) and 88,234 edges. Each ego user has connections with all nodes in his/her network. In this example, we compare the performance of BTD with SP in hierarchical and K-means clustering algorithms on one connected component consisting of both the first and second ego networks excluding the ego nodes. It has 547 nodes and 5706 edges.

In Figure 4, we show the proposed Facebook network with $K = 3$ clusters produced by K-means algorithm and BTD. As well as the HC results for BTD and SP for $K = 2, \ldots, 10$. The results show that BTD produces higher modularity than SP for the more important K values: in this network it appears that there are three natural communities, and for $K = 3, 4, 5$, HC produces a higher modularity score for BTD compared with SP. For $K > 8$, both distance



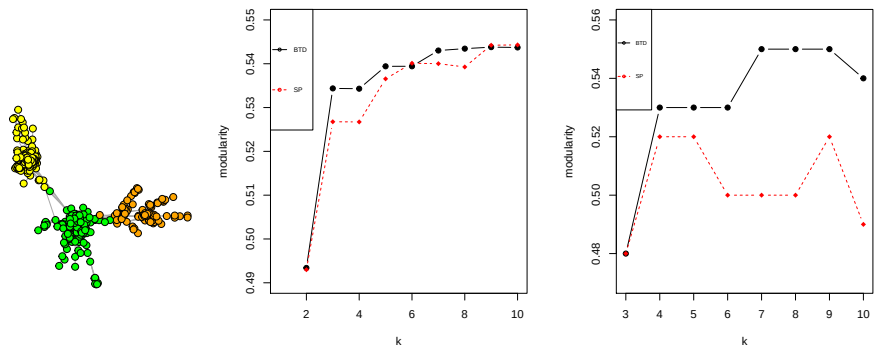
(a) ER model, $k = 5$ (b) ER model, $k = 7$ (c) WS model, $k = 5$ (d) WS model, $k = 7$



(e) BA model, $k = 5$ (f) BA model, $k = 7$ (g) FF model, $k = 5$ (h) FF model, $k = 7$

Fig. 3: Box plots of the differences of maximum/minimum modularity values between BTD and SP over 50 simulations of all proposed graph models of size $n = 200$, $K = 5, 7$ and 100 initial starts group in each simulation. The horizontal line crosses at zero to show the positive differences.

measures produce equal results, but the structural features have by this point disappeared. Even though Q reaches higher scores for large K, this is not a sign for a better number of clusters or communities, as the Q function is constructed to assess the cluster quality but not to choose the number of clusters [6]. We apply K-means algorithm with a different number of clusters $K = 3, \ldots, 10$, in each K we choose 10 different random initial centroids sets and check the maximum and minimum over these 10 sets for each K. We compare the performance of BTD with SP in this experiment by running the same 10 sets with each distance. Figure 4 illustrates the differences between BTD and SP by the minimum modularity scores over K. The results show a significant difference in favour of the BTD; this means the arbitrary choice of initial centroids always has a lower modularity score limit by SP than BTD. In the maximum comparison, both clustering methods behave similarly and produce similar modularity score. From HC and K-means algorithms experiments, we conclude that BTD produces higher quality results compared with SP in the Facebook network.



(a) Three clusters (b) HC results (c) Minimum K-means results

Fig. 4: Facebook graph with 3 clusters, comparison between BTD and SP in HC, and K-means clustering results.

6 Discussion

This paper presents a novel approach to measure distances in undirected, unweighted graphs. Its main idea is to break the ties between similar distances in dense graphs. In the experimental study, we examined the proposed distance BTD with four different graph models and concluded that BTD breaks the ties in SP and keeps the same SP order. Also, BTD has effective results compared with SP in graph clustering. In all simulation experiments, the results give evidence for the superiority of BTD compared with SP. This result is a significant finding in graph theory especially for graph clustering as most of the literature depends on SP.

Moreover, we have introduced a new way of assigning the nodes to the clusters in K-means algorithm based on the dissimilarity matrix, which differs from the standard K-means algorithm which is co-ordinate based.

Finally, we reaffirm our results by considering real data from the Facebook network.

Currently, we think over further study to reduce the time taken in repeating K-means algorithm for different initial centroids sets by considering deterministic choices. We intend to look for other real data which has a different structure from Facebook and compare the results. Moreover, we will check the validity of BTD in different statistical learning topics, for example, classification and regression.

Acknowledgments

The first author of this manuscript is grateful to the Saudi Arabian Cultural Bureau in London for financial support.

References

1. Aggarwal, C. C., Reddy, C. K.: DATA CLUSTERING: Algorithm and Applications. Taylor and Francis Group, London (2014)
2. Bonner, R.: On some clustering techniques. IBM Journal of Research and Development **8**, 22–32 (1964)
3. Clauset, A., Newman, M. E. J., Moore, C.: Finding community structure in very large networks. Physical review E 70:6, (2004)
4. Csardi, G., Nepusz, T.: The igraph software package for complex network research. InterJournal Complex System (2006) http://igraph.org
5. Everitt, S., Landau, B. S., Leese, M., Stahl, D.: Cluster Analysis. Wiley, Fifth edition, UK (2011)
6. Fortunato, S.: Community detection in graphs. Physics Reports **486**(3), 75–174 (2010)
7. Jain, A. K.: Data clustering: 50 years beyond K-means. Pattern Recognition **31**(8), 651–666 (2010)
8. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection. (2014) http://snap.stanford.edu/data/index.html
9. Madkour, A., Aref, G., Rehman, F., Abdur Rahman, M., Basalamah, S.: A survey of shortest-path algorithm. CoRR, abs/1705.02044 (2017) http://arxiv.org/abs/1705.02044
10. Newman, M. E. J.: Modularity and community structure in networks. Proceedings of the National Academy of Sciences of the United States of America **103**(23), 8577–8582 (2006)
11. Song, S., Zhao, J.: Survey of graph clustering algorithms using amazon reviews. (2014) http://snap.stanford.edu/class/cs224w-2014/projects2014
12. Udaya Kumar Reddy, K. R.: A survey of the all-pairs shortest paths problem and its variants in graphs. Acta Universitatis Sapientiae, Informatica **8**(1), 16–40 (2016) http://doi.acm.org/10. 1515/ausi-2016-0002
13. Zwick, U.: Exact and Approximate Distances in Graphs - A Survey. Springer, 33–48 (2001)
14. Blondel, V. D., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment **2008**(10), P10008 (2008)