2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems

Sigcon: Simplifying a Graph Based on Degree Correlation and Clustering Coefficient

Hojin Jung
School of Integrated Technology
Yonsei Institute Convergence Technology
Yonsei University
Incheon, Republic of Korea
hojin.jung@yonsei.ac.kr

Songkuk Kim
School of Integrated Technology
Yonsei Institute Convergence Technology
Yonsei University
Incheon, Republic of Korea
songkuk@yonsei.ac.kr

***Abstract*—Depicting a complex system like social networks as a graph helps understand its structure and relation. As advances in technology increase the amount of data, simplifying a large-scale graph has attracted interests. Simplification reduces the size of a graph while preserving its important properties.**

In this paper, we propose the summarization algorithm to simplify a graph focusing on degree correlation and clustering coefficient. The degree correlation is a measure to assess the influence of each vertex and their connections. The clustering coefficient estimates latent connections between two distinct vertices. To this end, we first separate a graph into communities. Looking at groups instead of a graph itself allows us to extract important vertices and edges more easily. We then categorize communities into four cases and simplify them in different ways to preserve the innate characteristics. The quantitative and qualitative evaluations demonstrate how effectively our algorithm serves the goal.

Overall, our contributions are as follows: (a) *unique pattern*: We found that hubs are connected indirectly via low-degree vertices. Sigcon preserves these vertices during simplification. (b) *efficient algorithm*: Sigcon identifies influential vertices and edges to simplify a graph on the basis of degree correlation and clustering coefficient connecting vertices effectively.

***Index Terms*—Graph Algorithm, Simplification, Degree Correlation, Clustering Coefficient**

I. Introduction

Many systems such as social networks and biological networks are modeled as graphs to depict their complex relation and structure. This enables to investigate important fractions more easily. As advances of technology generate a large amount of data, methods analyzing huge graphs as a summarization form have attracted interests [1]. Simplification is one of summarization techniques and aims to preserve several properties from a large graph. In this paper, we simplify a large-scale graph based on degree correlation and clustering coefficient.

Degree correlation is a measure to assess the importance of a vertex in the structure more deeply than the simple degree and to understand the structure of a graph. In a power-law graph [2], a few vertices have high-degree, and they exert powerful influence. This property only identifies the impact of a vertex itself in the entire graph. However, the influence of each vertex should be determined not only by the number of neighbors but also by who the neighbors are. For example, assortative graphs [3], hubs are connected to high-degree vertices, can describe why celebrities have married each other. On the other hand, disassortative graphs that high-degree vertices are linked to low-degree vertices can prove why people prefer listening unpopular music.

The other insight of simplifying a graph is clustering coefficient. While connectivity in a graph is primarily illustrated by physically existing edges, clustering coefficient is also an important factor in a graph since this predicts the latent connections among vertices. In sociological graphs, clustering coefficient explains why we tend to trust others more when they are friends of my friends. Clustering coefficient is categorized into two concepts, local clustering coefficient and global clustering coefficient. Local clustering coefficient focuses on the latent connections of a vertex, and global clustering coefficient estimates the latent connections in the entire graph. Here, global clustering coefficient is our main consideration.

To the best of our knowledge, simplifying a graph based on both degree correlation and clustering coefficient is not explored thoroughly. To deal with two properties in simplification together, we present the approach **Sigcon**, simplifying a graph based on connectivity in two stages. Connectivity refers to both degree correlation and clustering coefficient in our algorithm. First, we apply the link-clustering method [4] to extract communities. Crucial vertices and edges are detected more easily in communities because the sizes of communities are smaller than a graph. Link-clustering employs similarity between edges to find communities. Similarity depends on how many common neighbors end vertices of two edges have. After calculating each similarity, single linkage hierarchical clustering combines communities having close similarities into larger one. While combining communities that include many common vertices keeps their high density, merging communities having low similarities makes a less dense community.

The second step is coarsening communities and assembling them. Although the link-clustering method separates communities into highly connected groups and rarely connected ones, more detailed classification is required to maintain degree correlation and clustering coefficient simultaneously. We employ Watts-Strogatz clustering coefficient [5] to accomplish this

978-1-5386-2588-0/17 $31.00 © 2017 IEEE
DOI 10.1109/HPCC-SmartCity-DSS.2017.49

purpose. Watts-Strogatz clustering coefficient averages local clustering coefficients. Local clustering coefficient measures how many vertices are linked to the target practically among vertices connected to neighbors of a target vertex. As local clustering coefficient maintains the connectivity of a vertex itself as well as of its neighbors at the same time, the average local clustering coefficient categorizes communities preserving connections. We define the local clustering coefficient as the importance of a vertex and use the value in simplification.

Watts-Strogatz clustering coefficient is a criterion to separate communities into four cases: star, clique, various stars, and densely connected group. Simplification is employed regarding each community property. In star, since every vertex has the same importance, we preserve all vertices in summary. For clique, all vertices are adjacent one another, and this results that all of them have the same importance. Thus, we keep the formation like a star. In various stars, hubs are not connected directly, but they are interacted indirectly via small-degree vertices. We call the low-degree vertices bridges and the shape various stars. In this case, vertices that are neither hubs nor bridges are removed. In densely connected community, local clustering coefficients are ranked in descending order. Here, highly ranked vertices and their connections are preserved.

We evaluate our methodology quantitatively and qualitatively. In quantitative analysis we estimate simplification rates in 6 different graphs and compare global clustering coefficients between the original and simplified graphs. In qualitative analysis we test **Sigcon** in the aspects of scalability, and degree correlation. To describe the usefulness and effectiveness of our algorithm, we compare the visualized results of summarization with k-core and **Sigcon**. Datasets are obtained at [6] and the code is at [7].

Overall, the contributions of this paper are:

Pattern. One empirical pattern is observed where the community is various stars. Vertices linking to hubs are defined as bridges.

Efficient Algorithm. Our algorithm employs communities to simplify a graph by preserving degree correlation and clustering coefficient.

The remainder of this paper is organized as follows. In Section 2, we give some definitions and notations. In Section 3, we formulate the problem and present our approach. In Section 4, we organize our algorithm. After evaluating **Sigcon** in Section 5 and suggesting future work in Section 6, we conclude in Section 7.

II. Background and Related work

We begin by presenting notations and definitions frequently used in this paper. Before we describe our main schemes in next section, the correlation between degree and k-core is explored. The result is used to show the limitation of summarization with k-core.

A. Notations and Definitions

We consider an undirected and unweighted simple graph without multi-edge and self-loop, G. We define n as the number of vertices and m as the number of edges in G. For degree correlation of a graph, generally the measurement based on *Pearson coefficient* is used. *Degree correlation* is symbolized as r and varies from -1 to 1 both inclusive. Watts-Strogatz clustering coefficient, $\bar{c}$, is used to classify communities. Local clustering coefficient, c, determines the importance of each vertex. TABLE I lists all definitions and notations.

TABLE I
Definitions and Notations

Notations	Definitions
G	An undirected and unweighted graph
n	The number of vertices in G
m	The number of edges in G
r	Degree correlation
$\bar{c}$	Watts-Strogatz clustering coefficient
c	Local clustering coefficient

Degree correlation [8] measures the probability that vertices link to other ones with similar or dissimilar degree. That is, the similarity in *Degree correlation* implies that vertices and their neighbors have a similar amount of degree. If r is over 0, the graph is assortative. In the assortative graph, high-degree vertices avoid interacting to low-degree vertices, and vice versa. Otherwise, if r is less than 0, the graph is disassortative. In the disassortative graph, hubs are connected to neighbors with low-degree. The tendency of people to meet others who are different from them is one example. When r is equal to 0, the graph is neutral and vertices are connected randomly. The function of *Degree correlation* is:

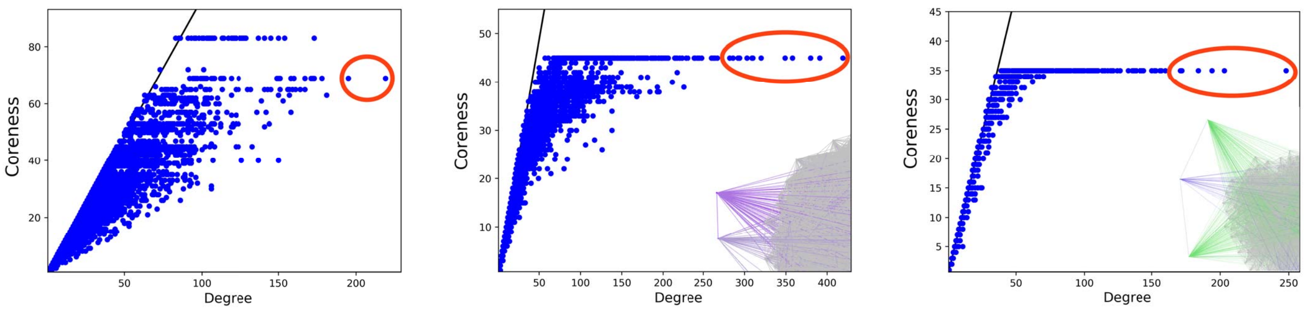
$$r = \frac{\sum_l (A_{il} - < A_i >) \sum_l (A_{jl} - < A_j >)}{\sqrt{\sum_l (A_{il} - < A_i >)^2} \sqrt{\sum_l (A_{jl} - < A_j >)^2}} \quad (1)$$

In this function, i, j, and l are vertices. Vertex i and vertex j are neighbors of l. A_{il} is the element of the lth row and the ith column of the adjacency matrix. $< A_i >$ denotes the mean $\sum_l (A_{il})/n$.

B. Correlation between k-core and Degree

There are many summarization approaches such as grouping [9], compression [10], and summarization with k-core. In this section we illustrate why the coarsening method with k-core [11] does not serve our goal maintaining both degree correlation and clustering coefficient. We first observe the correlation between degree and k-core. k-core is the collection of vertices having at least k degree to induce a subgraph. If the fraction with k-core is kept in a graph, vertices with degrees less than k are removed. Coreness is the maximum k-core of a vertex. Datasets used to describe the correlation between coreness and degree are listed in TABLE II.

The strong positive correlation between coreness and degree of each vertex is already studied in [12]. However, as the experiment is not tested targeting graphs sorted according



(a) Dublin (b) Mich67 (c) Caltech36

Fig. 1. **Correlation between coreness and degree in real graphs and the summarization with high k-core:** *Spearman's rank correlation coefficient* of three different graphs are strong positive, over 0.9. Red circles indicate $exceptions$. While Dublin and Mich67 are assortative, Caltech36 is disassortative. $Exceptions$ are shown in other graphs of TABLE II as well. In Mich 67 and Caltech 36, we visualized some portions of the summarized graphs with high k-core. In Mich 67, vertices with purple are hubs. In Caltech 36, small-degree vertices are colored with purple and high-degree ones are painted with green.

to their degree correlations, we examine the correlation in assortative and disassortative graphs again. *Spearman's rank correlation coefficient* is the correlation indicator. *Spearman's rank correlation coefficient* is close to 1 if coreness and degree have the strong positive correlation and the value is close to 0 if they are not correlated. As seen in Fig. 1, most vertices in all graphs show the positive correlation between coreness and degree, some vertices deviate this pattern, though. We call the vertices avoiding the strong positive correlation $exceptions$.

$Exceptions$ are vertices that have low coreness, but relatively high-degree. In order to see $exceptions$ formulated differently depending on degree correlation in detail, we depict them in a graph as seen in Fig. 1 (b) and (c). In Fig. 1 (b), the degree of vertices colored purple are high and they are connected directly each other. These High-degree vertices are important information to determine degree correlation of the assortative graph. However, if we raise the value k higher, the vertices with purple will not be remained. In Fig. 1 (c), the degree of the vertices with green are high and those which painted purple have low-degree. Although high-degree vertices are not connected directly, they are linked indirectly by low-degree vertices in this graph. This information is crucial to determine degree correlation of the disassortative graph. Nevertheless, if we raise the value k in the graph, vertices with green and purple are not preserved anymore. Therefore, the summarization with k-core shows the limitation in the aspect of degree correlation. Furthermore, losing the linkage information of vertices in assortative and disassortative graphs means losing the property of clustering coefficient after all.

III. Framework

The easiest way to simplify a large graph by preserving both degree correlation and clustering coefficient is finding fractions that follow the properties of the original graph empirically. However, there are some limitations in the approach. Since many real graphs are mixed with assortative and disassortative fractions, extracting vertices and edges required to preserve the same degree correlation of the original is difficult. Even though we find the perfect fractions having the same degree correlation to the first graph, it is controvertible whether they are reliable regarding clustering coefficient. In order to overcome these confinements, we developed the effective and useful simplification methodology, **Sigcon**. In this section, we propose our algorithm in two stages.

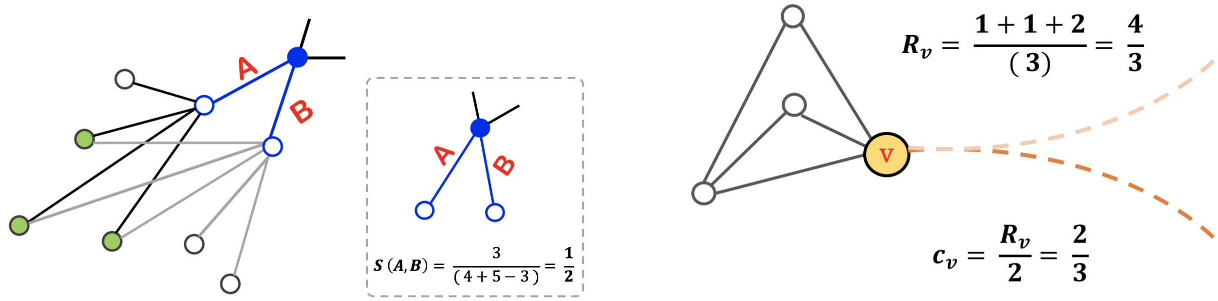
A. First Step: Decomposing a Graph into Communities

We adopt link-clustering [4] to find communities in a large graph. As finding fractions satisfying both degree correlation and clustering coefficient of the original has limitations, we decompose it into smaller units. Communities help understand the construction of a graph and find the important vertices and edges in better ways. Even though the traditional meaning of a community in the graph is a densely connected group [13], link-clustering defines it by computing the similarity between interrelated edges.

Similarity depends on how many vertices exist between two comparable edges. Suppose we compare two edges, **A** and **B**. The end vertex connected to **A** has 4 neighbors and the end vertex linked to **B** has 5 neighbors, but they have 3 common neighbors. The similarity between two edges is the value dividing 3 by 6 as shown in Fig. 2 (a). The approach demonstrates that edges hold the exclusive position and vertices hold the multiple relationship. This reflects the real world such that some students belong to soccer club and cello club simultaneously. After computing all similarities between edges, single linkage hierarchical clustering combines edges with the close similarity into one group and expresses them in a link-dendrogram. While edges with high similarities become a dense group, low similarity edges including less common vertices become a less dense group.

The link-clustering method obtains the best position to divide a graph into communities in link-dendrogram via partition density. D_i where i is the community number. Whereas modularity [14] usually used to extract communities only estimates densely connected groups, D_i holds the linkage property in high estimation. The density of D_i is:

$$D_i = \frac{m_i - (n_i - i)}{n_i(n_i-1)/2 - (n_i - 1)}. \tag{2}$$



(a) Similarity computation

(b) Redundancy of vertex v

Fig. 2. **Link-clustering and redundancy:** (a) shows how to compute similarity between edges and (b) shows the redundancy of vertex v.

The optimal partition density, D, is the mean value of all partition densities.

To sum up, merging similar communities together in link-clustering is beneficial to categorize them in two major groups: densely and sparsely connected communities. Moreover, decomposing a graph into communities depending on the partition density, D does not have resolution limits [15]. This allows us to preserve the connections between vertices and to classify more detailed groups more easily in the next step.

B. Second Step: Simplifying communities

In the previous section, we separate communities into densely connected and sparsely connected groups using link-clustering [4]. In this section, communities are classified into more detailed groups by Watts-Strogatz clustering coefficient [5] and are simplified based on their characteristic. Some confinements inspire this way. Given a community which is a perfect structure like a clique, its degree correlation is not determined. This produces a result that describing a community succinctly depending on degree correlation is difficult. In addition to that, as finding maximum clique is NP-complete [16], another simplification approach is needed.

Connections between vertices in a graph is the most important information to determine degree correlation and clustering coefficient. For connections between vertices in a community, we use Watts-Strogatz clustering coefficient. Watts-Strogatz clustering coefficient, $\bar{c}$, is the average of local clustering coefficients of a community. To compute a local clustering coefficient, we should know the redundancy [17] of a vertex first. If there is a vertex v, the redundancy of v divides the number of common neighbors between vertices linked to v by the degree of v as seen in Fig. 2 (b). Therefore, the function of the local clustering coefficient is:

$$c_v = \frac{R_v}{degree_v - 1}, \tag{3}$$

where R_v means the redundancy of v and the denominator is the maximum possible value of the redundancy. Since computing c maintains the connectivity information of a vertex itself and its neighbors, the Watts-Strogatz clustering coefficient, $\bar{c}$, can be a criterion for classification.

Watts-Strogatz clustering coefficient $\bar{c}$ refines two major community cases from link-clustering into four cases and we simplify a community depending on its property. Note that c is the importance of a vertex. If $\bar{c}$ of a community is 0, its shape is a perfect clique. If $\bar{c}$ is equal to 1, the community is a perfect star. Because all vertices in a clique and a star have the same structural role, our algorithm does not remove any vertices. On the other hand, when $\bar{c}$ is close to 0 or 1, insignificant vertices and connections will disappear.

1) Case 1 (Star): Every local clustering coefficient of vertices is 0. Accordingly, the average local clustering coefficient $\bar{c}$ is also 0. This indicates that all vertices have the same importance in a community. Therefore, a star which is a perfect structure does not miss any vertices in summarization.

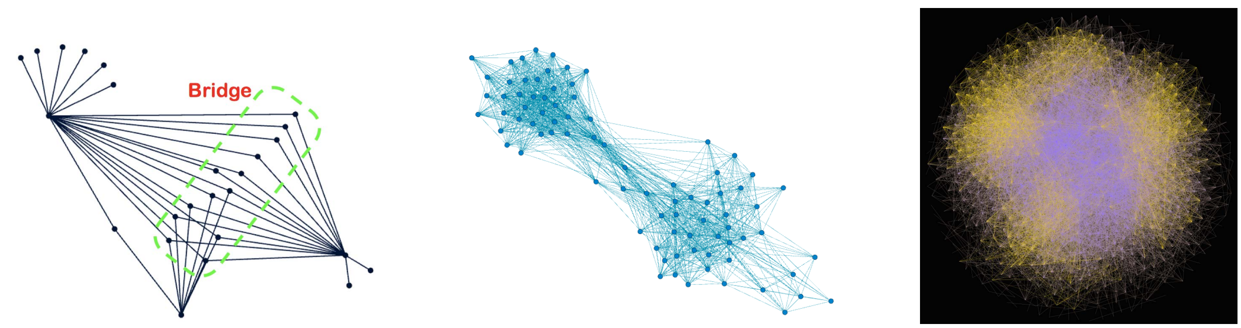
2) Case 2 (Clique): As all vertices link to each other with the same degree, they have the same local clustering coefficient in a community. In effect, Watts-Strogatz clustering coefficient $\bar{c}$ of a community is 1. In this case, since vertices have the same importance, any vertices will not disappear.

3) Case 3 (Various Stars): Although $\bar{c}$ is close to 0, each c of vertices differs. Because the difference between local clustering coefficients is little, we do not employ the value for simplification. Instead, our algorithm presents the other way to describe a community in simpler form. As seen in Fig. 3 (a), the unique pattern is discovered. While hubs are not connected directly, they interact indirectly by the vertices with as many degrees as the number of hubs. In this case we remain T hubs and bridges. The value, T, means the number of hubs.

4) Case 4 (Densely Connected Group): $\bar{c}$ is close to 1, and a community is dense as seen in Fig. 3 (b). In contrast to Case 3, local clustering coefficients are not small. In this case, we rank every c in descending order, and select top T vertices. Therefore, T vertices and their connections will be remained in summary.

IV. Implementation

The key contribution of our methodology is keeping degree correlation as well as clustering coefficient. Our algorithm has



(a) Case3 (b) Case4 (c) Caltech36

Fig. 3. **Cases and visualization:** Case 3 and Case 4 are the examples of communities in the graph, Mich67. (a) shows the unique pattern, various stars. (b) is the densely connected community. (c) is the visualization of Caltech36. The yellow fraction describes original graph and the simplified one is displayed with purple.

two main functions: dividing a graph into communities and simplifying them.

Sigcon employs the link-clustering method to find communities in the graph by computing each similarity between edges. In this scheme, edges with close similarities become a group. After the process, our methodology categorizes communities into four cases and makes each community simple. Although finding structures like a clique is useful to preserve the important information for connectivity, finding the maximal clique is NP-complete [16]. In effect, Watts-Strogatz clustering coefficient is used instead for classification. Simplification depends on each different property of communities.

As outlined in Algorithm 1 [18], **Sigcon** first extracts the communities via link-clustering. We then compute each Watts-Strogatz clustering coefficient of decomposed communities (line 3). If $\bar{c}$ is equal to 0 or 1, the shape is a star or a clique (line 4-5). If the rounded $\bar{c}$ is equal to 0, the form is various stars (line 6). In this case, T hubs and bridges will not be removed (line 7). When the value is close to 1, vertices are ranked based on local clustering coefficients in descending order and top T vertices will stay in simplification (line 8-9). In practice, the rate of summarization relies on T. The higher T, the more vertices and edges will remain. Low T gives more plain and simpler form.

Algorithm 1 Sigcon

Input: Graph G, T
Output: Summarized graph
1: Extract communities in G
2: **for each** community g such that $g \in G$ **do**
3: calculate $\bar{c}$ of g
4: **if** $\bar{c}$ = 0 or 1 **then**
5: pass
6: **else if** $\bar{c} \approx 0$ **then**
7: remain T hubs and bridges
8: **else**
9: remain top T vertices ranked in descending order based on c // $\bar{c} \approx 1$
10: **end if**
11: **end for**

The computation complexity of our methodology depends on link-clustering. The complexity of the link-clustering algorithm is $O(n^2)$ [4]. The complexity of computing Watts-Strogatz clustering coefficient depends on the number of degree of each vertex and less than $O(n)$. Selecting T hubs, bridges, and ranked T vertices has complexity $O(T)$ per each community. In summary, if we suppose that the total number of communities as x, the selection computation complexity is $O(xT)$.

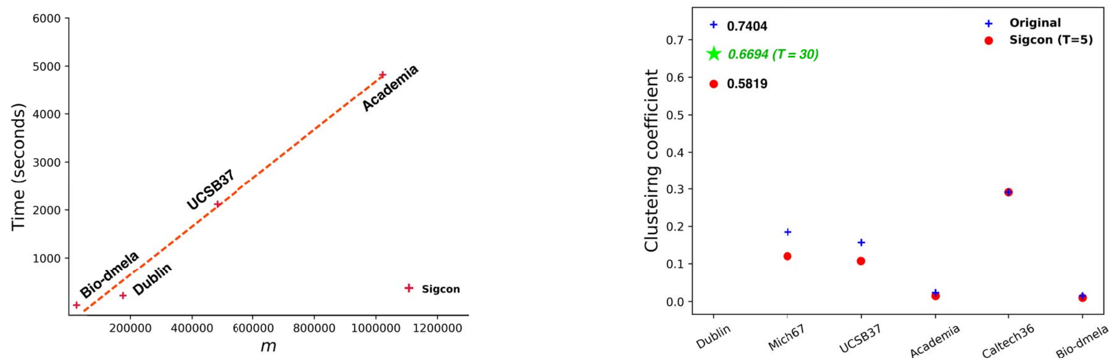
V. Evaluations

We measure the performance of our algorithm via quantitative and qualitative evaluations. Datasets used in this section are listed in TABLE II [6]. In the quantitative analysis, we proceed by comparing clustering coefficients between the original graphs and the simplified graphs using the summarization rates of 6 different graphs. In qualitative analysis, we demonstrate our algorithm effectiveness against the other summarization method with k-core. Furthermore, we observe that the correlation between the time complexity and edges is linear. Finally, We conclude by comparing degree correlations between the simplified graphs and the original.

A. Quantitative Evaluation

As shown in TABLE II, the graphs have various sizes and degree correlations. The value T is a criterion to affect the summarization ratio. The lower T, the more vertices and edges are removed in summary. We give T small values, 5 and 10, but it is adjustable whenever necessary. In addition, case 3 and case 4 can have different T. Although the rates are not quite large, simplification rates between 60% and 80% are general [24]. We visualize formation of Caltech36 is shown in Fig. 3 (c). Yellow fraction is the original graph and the purple one is the down-scaled form. While all datasets are categorized into two major types, assortative and disassortative graphs, each r value varies.

The latent connections among vertices can be measured by



(a) Scalability

(b) Clustering coefficient

Fig. 4. **Scalability and Clustering Coefficients:** As seen in (a), the running time in seconds of our algorithm increases linearly on the number of edges. (b) describes that the global clustering coefficients of summarized graphs are very close to the original ones.

TABLE II
SUMMARY OF GRAPHS

Graph	*n*	*m*	*r*	Type	*T*-5	*T*-10	Description
Dublin [19]	10972	175573	0.613	Assortative	18%	26%	Human contact graph
Mich67 [20]	3748	81903	0.142	Assortative	80%	83%	Social friendship graph
UCSB37 [20]	14917	482215	0.183	Assortative	84%	87%	Facebook social graph
Academia [21]	200169	1022441	-0.02	Disassortative	80%	85%	Social graph
Caltech36 [22]	769	16656	-0.065	Disassortative	68%	70%	Facebook social graph
Bio-dmela [23]	7393	25569	-0.047	Disassortative	68%	71%	Biological graph

clustering coefficient. Clustering coefficient is categorized into global clustering coefficient and local clustering coefficient. In this evaluation, we use the global clustering coefficient and it is the value dividing the number of existing triangles by the number of triple paths in a graph. Note that global clustering coefficient differs from Watts-Strogatz clustering coefficient. Fig. 4 (b) shows six simplified graphs have alike clustering coefficients to the original ones. However, in the dataset, Dublin, the difference of clustering coefficient between the original graph and the summarized one is a little bit more. To understand this difference, we look at the summarization rates of Dublin in detail.

Comparing the three assortative graphs, we see that the summarizaiton ratio of Dublin makes much of a difference against other graphs, Mich67 and UCSB37. We investigate the reason via the shape of Dublin and the value r. Dublin is composed of many separated small subgraphs and a big subgraph. Also, the higher r indicates that vertices in the subgraphs having similar degree are linked to each other. Since not only low-degree vertices but also high-degree vertices are linked to each other densely, many communities belong to densely connected groups. In this case, if we adopt low T, subgraphs with small nodes and edges cannot be kept in summary. Furthermore, our algorithm ranks vertices in descending order depending on their local clustering coefficients and does not consider the same local clustering coefficients as the same rank. This removes vertices with high and identical local clustering coefficients in a community. As the result, a lot of vertices with similar c and edges connecting them are deleted in relatively big-sized community and the size of the graph Dublin is also dwindled a lot.

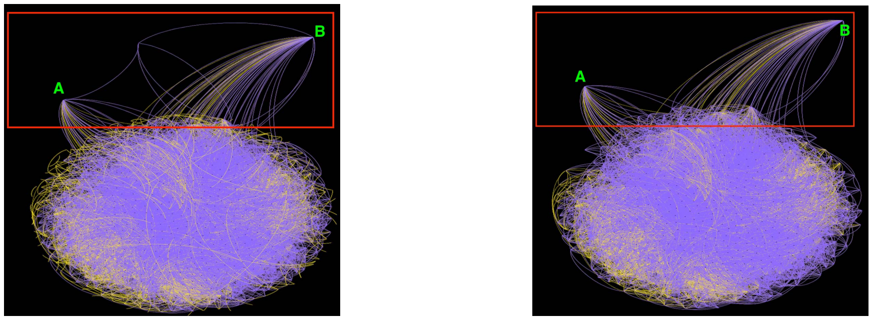
In order to show that this problem is adjustable, we increase the value T to 30 targeting the graph, Dublin. The summarization rate of Dublin with T-30 is 56 percentages and illustrates that the removed fractions in simplification with T-5 are preserved by adjusting T simply. In addition to that, as seen in Fig. 4 (b), the difference of global clustering coefficient between the original graph and the simplified one decreases. In Fig. 4 (b), while the simplified graph with T-5 is close to 0.58, the summarization with T-30 is close to 0.67. This result describes that the error regarding clustering coefficient in Dublin is also reducible by increasing the value T easily.

B. Qualitative Evaluation

Next, visual comparison between our algorithm and the representative coarsening algorithm with k-core is explored. We propose **Sigcon** has linear correlation between the runtime and edges in scalability. To properly calculate time complexity of the graphs in TABLE II, all datasets have different sizes. We also show the simplified graphs have the same degree correlation.

1) Visual Comparison: Our algorithm simplifies a large graph by capturing $exceptions$. Fig. 5 (a) shows **Sigcon** preserves $exceptions$ in simplification differently from the summarization with k-core. For comparison, we adopt a diassortative graph, Caltech36.

In this evaluation, we suppose that there are vertex **A** and vertex **B** which are hubs and they are linked via the other vertex indirectly. As seen in Fig. 5 (b), the coarsening



(a) **Sigcon** (T = 5) (b) k-core (k = 10)

Fig. 5. **Simplification with Sigcon and with k-core:** While the yellow shows the original graph shape, the purple indicates the simplified fraction. Although we summarize Caltech36 with low T in **Sigcon**, vertices linking to hubs indirectly are not deleted. On the other hand, although we remove the small portion in (b), the important information is not kept in summary.

algorithm with k-core misses $exceptions$. Even though we prefer low k-core in this graph, $exceptions$ are removed after all. Losing $exceptions$ as hubs in the simplified graph corresponds to losing the crucial information for the degree correlation of the original graph. However, our algorithm makes a simpler graph without omitting $exceptions$.

2) Scalability: The correlation between running time and the number of edges is linear as seen in Fig. 4 (a). Four graphs having different sizes are selected. The dotted red line indicates that time complexity of **Sigcon** increases linearly depending on the number of edges. We ran this experiment on Intel(R) Core i7-5820K CPU at 3.3GHz With 64GB memory.

3) Degree Correlation: The goal of this paper is to make graphs in simpler shape depending on degree correlation and clustering coefficient. For degree correlation, we observe the average degree correlation of vertices with degree k and plot it as seen in Fig. 6. K_{nn} [25] is to compute the average degree of neighbors of a vertex. The function of K_{nn} is:

$$K_{nn}(k_i) = \frac{1}{k_i} \sum_{j=1} A_{ij} k_j, \qquad (4)$$

where k_i is the degree of vertex i and k_j is the degree of vertex j. A_{ij} is 1, when vertex i and vertex j are adjacent, otherwise, 0. As seen in Fig. 6, in assortative graphs every K_{nn} increases with the number of degrees. By contrast, in disassortative graphs each K_{nn} decreases with degrees. The dotted green line is to show that the K_{nn} in simplified graph follows the distribution of the original graph. This result suggests that all graphs follow similar degree correlations to the original.

VI. Discussion and Future Work

In this work, we present a simplification method based on the degree correlation and the clustering coefficient. The approach allows us to summarize a large graph efficiently satisfying our goal. Besides, the substantive and latent properties offer a chance to understand a graph in various aspects. However, some future work still remains for better performance.

First, we cannot change the computational complexity, $O(n^2)$, of the link-clustering method. Although discovering communities based on linkage property can categorizes a graph into smaller units, coarsening a trillion vertices and edges simply in a single machine is quite limiting. Second, we do not analyze the betweenness centrality of bridges linking two hubs indirectly. Betweenness centrality is a measurement to find the shortest path between vertices. In graphs, we assume that the bridges have high betweenness centrality and expect they can be used for simplification. In effect, small-world effect [26] that every distance between vertices is small is also kept in summary.

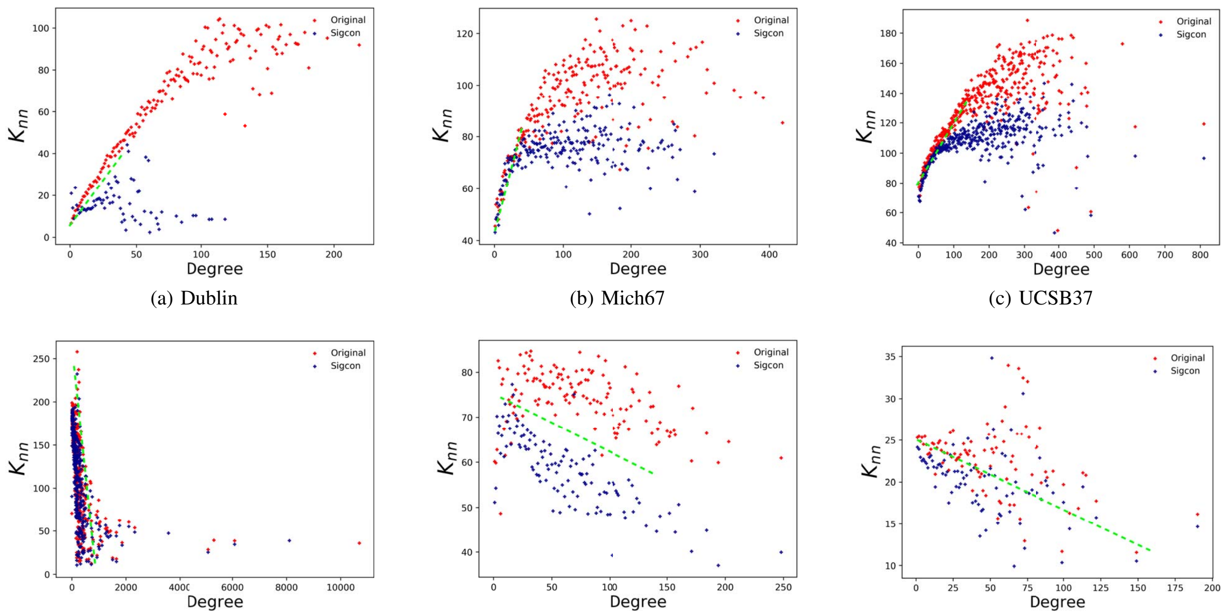
In summary, the future work is reducing the computation complexity for much larger sized graphs and proving the effectiveness of bridges in summarization.

VII. Conclusions

We developed a well-founded simplification algorithm based on the leverage of a vertex. Preserving degree correlation and clustering coefficient allows us to understand the substantive and latent connections among vertices in simplification. To this end, we designed our algorithm in two steps. **Sigcon** separates a large real graph into communities since extracting small units with the link-clustering method maintains connections between vertices. Furthermore, crucial vertices and edges are detected more easily in a community than in a graph. **Sigcon** then categorizes communities into four cases with Watts-Strogatz clustering coefficient. The value averages local clustering coefficients in a community, thus, classification depending on the mean of local clustering coefficients can maintain its unique connectivity property between vertices. This also leads to simplify a community in distinct ways. Quantitative and qualitative experimental evaluations demonstrate the effectiveness of our algorithm.

Acknowledgment

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ICT Consilience Creative program(IITP-2017-2017-0-01015) supervised by the IITP(Institute for Information communications Technology Promotion)



(a) Dublin (b) Mich67 (c) UCSB37

(d) Academia (e) Caltech36 (f) Bio-dmela

Fig. 6. **Degree Correlations:** We compare degree correlations between simplified graphs and the original ones. The dotted green line is showing whether the simplified graph has similar distribution of K_{nn} to the original graph. The red and blue points are K_{nn} values of the original graph and **Sigcon**.

References

[1] Y. Liu, A. Dighe, T. Safavi, and D. Koutra, "A Graph Summarization: A Survey," vol. V, 2017.

[2] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *ACM SIGCOMM computer communication review*, vol. 29, no. 4. ACM, 1999, pp. 251–262.

[3] M. E. Newman, "Assortative mixing in networks," *Physical review letters*, vol. 89, no. 20, p. 208701, 2002.

[4] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, no. 7307, pp. 761–764, 2010.

[5] M. E. Newman and D. J. Watts, "Scaling and percolation in the small-world network model," *Physical Review E*, vol. 60, no. 6, p. 7332, 1999.

[6] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. [Online]. Available: http://networkrepository.com

[7] H. Jung. (2017, jul) Sigcon: Simplifying a graph based on degree correlation and clustering coefficient. [Online]. Available: https://zenodo.org/badge/latestdoi/95521975

[8] M. E. Newman, "Mixing patterns in networks," *Physical Review E*, vol. 67, no. 2, p. 026126, 2003.

[9] K. LeFevre and E. Terzi, "Grass: Graph structure summarization," in *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM, 2010, pp. 454–465.

[10] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos, "Vog: summarizing and understanding large graphs," in *Proceedings of the 2014 SIAM international conference on data mining*. SIAM, 2014, pp. 91–99.

[11] W. Khaouid, M. Barsky, V. Srinivasan, and A. Thomo, "K-core decomposition of large networks on a single pc," *Proceedings of the VLDB Endowment*, vol. 9, no. 1, pp. 13–23, 2015.

[12] K. Shin, T. Eliassi-Rad, and C. Faloutsos, "Corescope: Graph mining using k-core analysis-patterns, anomalies and algorithms." ICDM, 2016.

[13] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 9, pp. 2658–2663, 2004.

[14] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical review E*, vol. 69, no. 2, p. 026113, 2004.

[15] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *Proceedings of the National Academy of Sciences*, vol. 104, no. 1, pp. 36–41, 2007.

[16] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy, "Approximating clique is almost np-complete," in *Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on*. IEEE, 1991, pp. 2–12.

[17] R. S. Burt, "The social structure of competition," *Explorations in economic sociology*, vol. 65, p. 103, 1993.

[18] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of the 7th Python in Science Conference (SciPy2008)*, Pasadena, CA USA, Aug. 2008, pp. 11–15.

[19] R. A. Rossi, D. F. Gleich, A. H. Gebremedhin, and M. A. Patwary, "What if clique were fast? maximum cliques in information networks and strong components in temporal networks," *arXiv preprint arXiv:1210.5802*, pp. 1–11, 2012.

[20] A. L. Traud, E. D. Kelsic, P. J. Mucha, and M. A. Porter, "Comparing community structure to characteristics in online collegiate social networks," *SIAM Rev.*, vol. 53, no. 3, pp. 526–543, 2011.

[21] M. Fire, L. Tenenboim, O. Lesser, R. Puzis, L. Rokach, and Y. Elovici, "Link prediction in social networks using computationally efficient topological features," in *IEEE Third International Confernece on Social Computing (SocialCom)*. IEEE, 2011, pp. 73–80.

[22] A. L. Traud, P. J. Mucha, and M. A. Porter, "Social structure of Facebook networks," *Phys. A*, vol. 391, no. 16, pp. 4165–4180, Aug 2012.

[23] R. Singh, J. Xu, and B. Berger, "Global alignment of multiple protein interaction networks with application to functional orthology detection," *PNAS*, vol. 105, no. 35, pp. 12 763–12 768, 2008.

[24] C. Dunne and B. Shneiderman, "Motif simplification: improving network visualization readability with fan, connector, and clique glyphs," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2013, pp. 3247–3256.

[25] A. Barrat, M. Barthelemy, R. Pastor-Satorras, and A. Vespignani, "The architecture of complex weighted networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 11, pp. 3747–3752, 2004.

[26] D. J. Watts and S. H. Strogatz, "Collective dynamics of small-worldnetworks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.