JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 16, No. 4, December 2022, pp. 233-243

Power System Connectivity Visualization Using an Orthogonal Graph Layout Algorithm Based on the Space-Filling Technique

San Hong, Sangjun Park, Chanil Kim, and Hyunjoo Song*
School of Computer Science and Engineering, Soongsil University, Seoul, Korea
sanhong@vis.ssu.ac.kr, sjpark1208@vis.ssu.ac.kr, chanil@vis.ssu.ac.kr, hsong@ssu.ac.kr

Abstract
In this study, a novel orthogonal graph layout algorithm was proposed to efficiently represent the connection relationship between each node in power system data. The proposed layout algorithm is a four-stage algorithm. First, clustering is performed based on the connection relationship between nodes, and the obtained clusters are placed close to squares by using the squarified treemap technique to fully utilize the given space. Next, adjacent nodes were arranged in a snake-like order in each cluster according to the characteristics of the IEEE test system data. The links were then arranged according to the positional relationship of the pairs of connected nodes. Each node had several ports so that links could be distributed evenly according to the direction of the links. A case in which all nodes were arranged orthogonally in an arbitrary manner without performing clustering; a case in which adjacent nodes within each cluster were arranged in an arbitrary order after only performing clustering; and a case where adjacent nodes within each cluster were arranged in row-major order after only performing clustering were compared. The results verified that the proposed method improved edge crossing, edge bending, and edge length considerably.

Category: Human-Computer Interaction

Keywords: Graph layout algorithm; Orthogonal layout; Space-filling technique; Power system visualization; Visualization; Human-computer interaction

I. INTRODUCTION

The efficiency of the visual analysis of the connection relationship between nodes is considerably affected by the method of arranging nodes and links. For example, in terms of visualization scalability, if the number of nodes or links increases, links may pass over nodes or links may overlap with one another, which can result in visual clutter. When visualizing multi-dimensional data in multiple-coordinated views by organically linking various visualizations, the area of the screen in which the connection relationship is expressed has limitations. Therefore, the development of an efficient method is critical for expressing the connection structure in exploratory analysis of multi-dimensional data in which connection relationship information exists.

Power system data have the aforementioned connection relationship. Studies have focused on the transmission and distribution of electric power between the nodes in a system. The verification of newly proposed algorithms or prediction models for power demand and generation is a critical topic of research. In these studies, other variates along with the connection relationship between nodes are measured and verified. Power system researchers commonly

Open Access http://dx.doi.org/10.5626/JCSE.2022.16.4.233 http://jcse.kiise.org

Received 14 September 2022; **Accepted** 20 December 2022
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

use IEEE test case data [1] with various numbers of nodes to compare the results of various studies efficiently. In these data, a system may consist of as few as 14 nodes and as many as 1062 nodes. For a small-size system, the user can directly configure the system using commercial software such as PowerWorld [2] and incorporate a visual analysis of the system. However, if a system consists of hundreds of nodes, the connection relationships of the system should be entered manually. Thus, scalability limitation exists.

The orthogonal layout method is widely used to express connection relationships of power systems. In the orthogonal layout method, nodes are arranged on a grid and subsequently connected with links bent at right angles. However, in the orthogonal layout method, the length or the number of bends is higher than that of the force-directed layout method, in which nodes with straight links are unconditionally connected. The orthogonal layout method has been extensively studied and its application has been extended to other fields. Therefore, automated layout methods have attracted considerable research attention. However, because studies have focused on limited number of nodes in systems, manual static layouts are used instead of automated layouts. However, this approach achieves poor performance in systems with hundreds to thousands of nodes. Unlike power systems developed to perform simulations in studies, in practice, power systems consist of many nodes. Therefore, an automated approach is necessary.

The rest of the paper is organized as follows: Section II provides a review of existing layout algorithms. Section III describes the representative connection data of power systems that are the focus of this study. Section IV explains the proposed automated layout algorithm for data with the characteristics described in previous sections. Section V quantitatively compares the efficiency of the layout results for networks that were randomly constructed with the standard power system data. Finally, Section VI presents conclusions and future research direction.

II. RELATED WORK

The force-directed layout is widely used for arranging nodes and links and in various visualization libraries [3, 4]. However, because the position of nodes is gradually calculated in the force-directed algorithm, considerable computation is required to obtain the final position. Therefore, in this algorithm, visualization of a large-scale connection structure with many nodes becomes time consuming. Moreover, large gaps are created between the nodes during the layout process because of the repulsive force among the nodes, which results in an inefficient use of the given space. In this study, the position of nodes was derived after calculating the clusters of nodes to reduce excessive computations in calculating the position of nodes. In this study, the layout based on space-filling technique, which can maximize the given space, was proposed to increase space efficiency. Studies [5, 6] have focused on using machine learning in the layout process. However, these studies are not specialized for the orthogonal layout. Therefore, the training data for the orthogonal layout should be prepared to perform separate training. Furthermore, these studies did not maximize the use of space.

The space-filling technique, which is used to utilize space effectively in data representation, was used in the visualization of the graph and hierarchical structure. In a study focusing on treemaps [7], the space was first divided horizontally according to the size of the lower-level structure. Next, the space was subdivided vertically. Thus, the space was alternately divided horizontally and vertically. However, rectangles that are excessively long on one side can occur. In this study, the squarified treemap [8], which is an improved version of treemap, was used so that the space in which a cluster of nodes are arranged appear as close to a square as possible. Such layout methods in which a limited space is divided have been studied. Wu et al. [9] used a Voronoi tessellation to allocate space for each cluster in a graph in which clusters of nodes exist. However, the arrangements of nodes or link connections within each cluster were not orthogonal.

Kieffer et al. [10] conducted a user study to investigate the connection method that people aesthetically prefer when nodes are connected orthogonally and subsequently performed the layout based on the user study results. The process consisted of the following steps: major nodes were derived through topological decomposition; first, the major nodes were arranged; the remaining nodes were then connected, and finally, the layout was refined. In their study, link connections were concise; however, the available space on the screen could not be fully utilized when refining the layout. Wu et al. [11] used the treemap technique to divide the space and arrange nodes orthogonally within each section when visualizing a biological pathway. However, because the objective of their study was to distribute the links evenly throughout the screen, many gaps existed. Rüegg et al. [12] placed several ports on the nodes to connect links to achieve a compact layout through stress minimization. In this study, the treemap technique was used to divide the space to fully utilize the available space. Next, the clusters were arranged. The nodes within clusters were then arranged. In this process, the nodes were packed within the space to produce orthogonal results. Twelve ports were then placed in a node in four directions so that the links could be distributed evenly among the ports according to the connection direction.

Various criteria have been used to evaluate the layout of nodes and links in each algorithm. Gibson et al. [13] emphasized the importance of link crossing in the evaluation

of layout algorithms and revealed that symmetry is reported differently in various studies. Furthermore, they mentioned that the criteria that are mainly considered in the graph layout are the uniform length of links, uniform distribution of nodes, separation of nonadjacent nodes, and overlap between nodes and links. In this study, the space-filling approach was used, so the nodes were uniformly distributed. Therefore, evaluation was not performed based on the uniform distribution of nodes criterion. Unlike the force-directed approach, maintaining uniform length of the links in an orthogonal layout is difficult. Therefore, this criterion was excluded. Furthermore, the layout algorithm was designed such that among the remaining evaluation criteria, the separation of nodes and the overlap between the nodes and links did not occur. Rüegg et al. [12] and Kieffer et al. [10] investigated the indicators for the number of link bend points and the link length in addition to the above criteria. In this study, we conducted a quantitative evaluation by examining the number of link crossings, the number of link bend points, and the link length.

III. IEEE TEST CASE DATA

IEEE test case data were used as the representative data for verification and comparison with previous studies [14, 15]. These data were compiled by using parts of systems at a particular time to easily verify various simulation results based on existing systems. In power system studies, these data are used to compare the study results with other research results. In these data, a system may have as few as 14 nodes and as many as 1062 nodes. Table 1 presents a summary of the systems in these data. These systems are multi-graphs in which one or more links exist between a pair of nodes, and each system has a distinct number of links and density of links (calculated by postulating these are simple directed graphs). However, in this study, the layout was performed by excluding duplicate links between the same pair of nodes. We applied Louvain clustering algorithm [16] to each system based on the connection relationship of the links. The results revealed that the number of clusters in each data ranged from one to nine.

Table 1. Network characteristics of the IEEE test case data with different number of nodes

	Nodes	Links	Density	Clusters
1	14	20	0.1099	1
2	30	41	0.0471	1
3	57	80	0.0244	1
4	118	186	0.0130	2
5	300	411	0.0046	5
6	1062	1683	0.0014	9

IV. GRAPH LAYOUT ALGORITHM

The objective of the proposed algorithm was to reduce visual clutter when arranging the nodes and links in a limited space. Moreover, the sum of the link lengths in the layout, the number of times the links overlap with one another, and the number of times the links bend, as used in previous studies, were selected as the quantitative indicators for visual clutter. Therefore, the algorithm was designed to yield small values for these indicators when arranging the nodes and links.

A. Node Clustering

In the node clustering step, closely connected clusters were extracted before the nodes are placed in their respective clusters. The clustering process increased the number of links in each cluster and minimized link connections between the nodes that belong to different clusters. The objective of clustering was to reduce the number of connections between distant nodes by placing closely connected nodes near each other in a cluster. Therefore, in this study, we used the Louvain community detection algorithm. This algorithm starts from the stage in which each node comprises a cluster and grows clusters such that connections within a cluster are maximized and connections between clusters are minimized when clusters are merged.

B. Cluster Layout with Space-filling Technique

Each cluster was arranged using a space-filling technique (i.e., treemap technique) to fully utilize the given space. In the treemap technique, the entire given space is divided according to the size of the attributes of choice in each layer of the hierarchical structure. The space is recursively divided according to the lower-level items in the hierarchical structure. In this study, the space was divided once (i.e., without recursion to the lower level) according to the clusters resulting from Louvain algorithm.

The space was allocated according to squarified treemaps [8]. In this technique, the method proposed in the first treemap study was extended, and the space was divided so that the outcomes are close to squares. We prioritized dividing the space into square-like shapes because when the nodes were arranged in a space shaped like a vertically or horizontally long rectangle, the adjacent nodes could be placed on the opposite ends of the rectangle depending on the order or direction in which these nodes were arranged. In this case, the length

of the links was longer than when the nodes were arranged in a square-shaped space.

C. Arranging Nodes within Each Cluster

After cluster layout was completed, the nodes were arranged in the clusters. First, the space allocated to each cluster in the previous section was again divided uniformly according to the number of nodes. Thus, the width and length of the entire space were divided equally by the rounded-up value of the square root of the number of nodes in the clusters to obtain a grid with a uniform size. The nodes were then placed in the center of the areas separated by the grid to create a uniform gap between the nodes. However, a large empty space could occur in the last row if the nodes were arranged from the first row of the grid, depending on the number of nodes. Nevertheless, in this method, the potential positions at which the nodes may be placed can be determined only using the square root operation. Therefore, we did not adopt additional steps to minimize the empty space considering scalability in terms of the computation time based on the number of nodes.

Next, several orders may exist in which the nodes are placed at each derived position. However, in this study, we considered the characteristics of the data and assumed that a high probability exists that a connection is present between nodes with adjacent unique IDs. This assumption is revisited in the quantitative analysis process in Section 5. Based on this assumption, the nodes were placed in the ascending order of the unique numbers in the layout process, starting from the left side of the top row, going from left to right, and then from right to left in the next row to place the nodes with adjacent unique IDs close to each other. This adjustment ensured arrangement of the nodes with adjacent unique IDs close to each other even when the nodes were placed on adjacent rows (Fig. 1).

In this study, we assumed that adjacent nodes have adjacent unique IDs. Therefore, we did not reset the order of the nodes. However, if this assumption does not hold, the results related to reordering the adjacency matrix can be utilized [17]. When the data are represented in the form of an adjacency matrix, the sorting order determines generation of a meaningful pattern (i.e., clique), even for identical data. Therefore, sorting algorithms have received considerable research attention. Using these studies, the nodes can be rearranged so that a pattern called clique can be verified. Furthermore, if the nodes are arranged in this order, adjacent nodes can be processed in sequence, as assumed earlier.

To evaluate the differences with the node layout method, Section 5 presents a comparison of the snake-order layout method proposed in this study with the following methods: the nodes are arranged randomly without performing clustering; the nodes are arranged randomly within the cluster; the nodes are arranged in order within the cluster. A comparison of these methods verified the differences in quantitative indicators based on the node layout method.

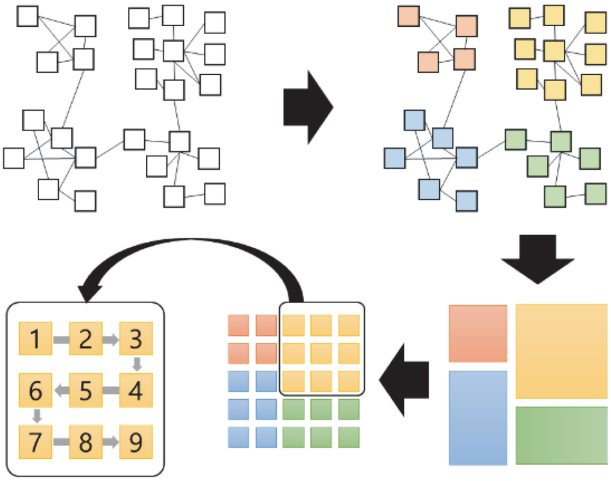

Fig. 1. Node layout process. (Step 1) Apply Louvain clustering based on the connection relationship between the nodes. (Step 2) Arrange the clusters, which are represented with distinct colors, by applying squarified treemaps to the clusters. (Step 3) Derive candidate node positions in each cluster. (Step 4) Arrange the nodes in snake-like order in each cluster.

D. Arranging Links Between the Nodes

Links between nodes are arranged differently depending on the positional relationship between the start and end nodes of the links. Because directed graphs were

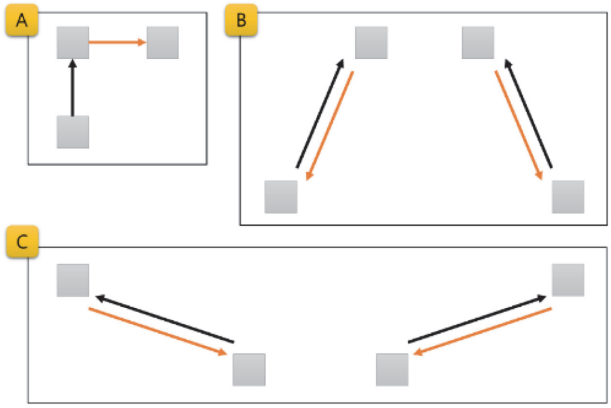

Fig. 2. Classification based on the positional relationship between the start and end nodes (A) The two nodes are located on the same row (orange) or same column (black) (B) The two nodes are located on the opposite ends of a long diagonal line that runs from top to bottom (C) The two nodes are located on the opposite ends of a long diagonal line that runs from left to right.

considered in this study, the positional relationship between the start and end nodes of the links can be classified into ten cases (Fig. 2). The start and end nodes of a link can be located on the same row or column. The start and end nodes of a link can be located on the opposite ends of a long diagonal line that runs from top to bottom or from left to right. Furthermore, each case can be divided into two cases according to the direction of the link (orange and black links in Fig. 2).

Next, the positional relationship of the nodes that exist between the two connected nodes is considered when the links are arranged. Twelve connection points are arranged on each node in four directions (up, down, left, and right), with three connection points in each direction. The links are arranged using these connection points as the two endpoints. The start and end positions of the link are summarized in Table 2 based on the connection point numbers displayed in Fig. 3A. The simplest case is when two nodes are adjacent to each other. If these nodes are directly adjacent to each other on the same row or column, the connection is made to the middle connection point on the adjacent side of each rectangle representing the node (connection points 1, 4, 7, and 10 in Fig. 3A). However, even if the two nodes are on the same row or column, if another node (orange squares in Figs. 3C and 3D) exists between them, they cannot be connected directly. Instead, the connection should bypass the node between the two nodes. For a vertical node connection, the connection goes around in the left or right direction (Fig. 3C). For a horizontal node connection, the connection goes around in the upward or downward direction (Fig. 3D). However, if connections are made in the same direction only (e.g., when connecting to a node on the right, the connection starts from 2 and ends at 0), links existing in the same row can easily overlap. Therefore, the connection was made in the opposite direction,

Table 2. The start and end points of the link according to the direction of the link, the positional relationship of the nodes, and whether the nodes are adjacent

Direction	Dominant direction	Obstacle	Start	End	# of bending points	Example
Left	-	No	10	4	0	Fig. 3B
		Yes	0 or 8	2 or 6	2	Fig. 3D
Right	-	No	4	10	0	Fig. 3B
		Yes	2 or 6	0 or 8	2	Fig. 3D
Top	-	No	1	7	0	-
		Yes	3 or 11	5 or 9	2	Fig. 3C
Bottom	-	No	7	1	0	-
		Yes	5 or 9	3 or 11	2	Fig. 3C
Right Bottom	Vertical	No	5	11	2	Fig. 3E (left)
		Yes	5	0	3	Fig. 3E (right)
	Horizontal	No	6	0	2	Fig. 3F (top)
		Yes	6	11	3	Fig. 3F (bottom)
Left Top	Vertical	No	11	5	2	Fig. 3E (left)
		Yes	11	6	3	-
	Horizontal	No	0	6	2	Fig. 3F (top)
		Yes	0	5	3	-
Right Top	Vertical	No	3	9	2	-
		Yes	3	8	3	-
	Horizontal	No	2	8	2	-
		Yes	2	9	3	-
Left Bottom	Vertical	No	9	3	2	-
		Yes	9	2	3	-
	Horizontal	No	8	2	2	-
		Yes	8	3	3	-

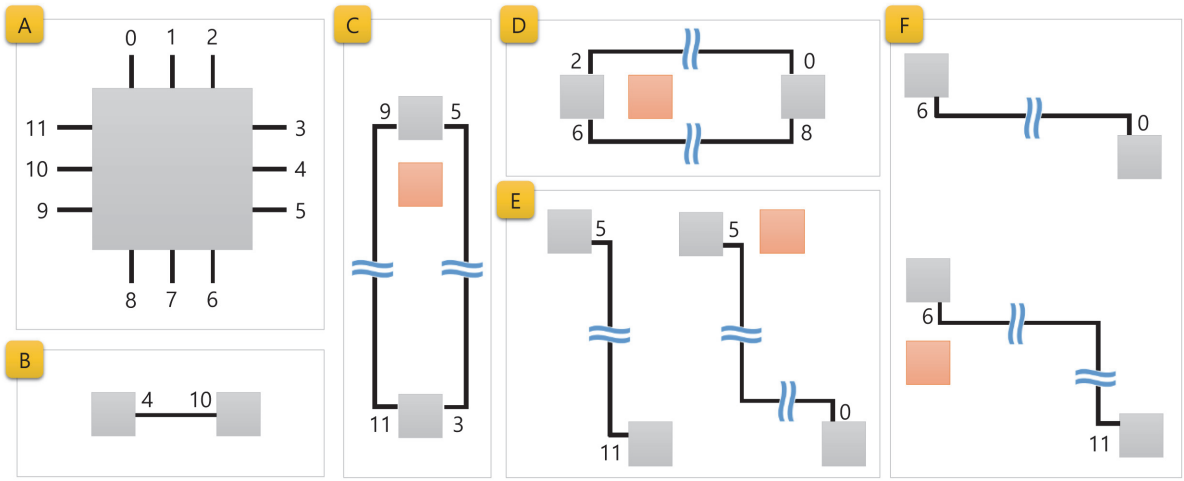

Fig. 3. Link connection based on the positional relationship between two nodes (A) Three-link connection points are arranged on a node in each of four directions, namely up, down, left, and right. (connection point numbers are assigned from 0 to 11) (B) The two connected nodes exist on the same row, and no other node exists between them. The two nodes are connected at connection points 4 and 10, which is the shortest distance. (C) The two connected nodes exist on the same column, but another node (orange square) exists between them. In this case, the link is arranged randomly to connect to either the connection point on the left or right. (D) The two connected nodes exist on the same row, but another node (orange square) exists between them. In this case, the link is arranged randomly to connect to either the connection point on the top or bottom. (E) Two connected nodes are located far apart in the vertical direction. If the two nodes exist on the adjacent columns, the link connecting them is bent twice, as displayed on the left diagram. However, if more than one column exist between the two nodes, the link connecting them is bent three times. (F) Two connected nodes are located far apart in the horizontal direction. If the two nodes exist on the adjacent rows, the link connecting them is bent twice, as displayed in the top diagram. However, if more than one row exist between the two nodes, the link connecting them is bent three times, as displayed in the bottom diagram.

depending on whether the connection point on the start node is an even or odd number (e.g., when connecting to a node on the right, the connection starts from 8 to ends at 6).

The case in which the nodes were located in different rows or columns can be categorized into two cases. The first case is when the nodes are not directly adjacent to each other but are located in adjacent rows or columns (the left diagram in Fig. 3E, the top diagram in Fig. 3F). In this case, the link is arranged in the space between the adjacent rows or columns. This approach shortens the overall length of the link, even if link bending occurs two times. The second case is when the nodes are located on various nonadjacent rows and columns (i.e., an obstacle exists in the middle of the path). In this case, the connection was ensured by bending the link three times (the right diagram in Fig. 3E, the bottom diagram in Fig. 3F). The first bending point was right next to the start node; the link was bent in the direction of the target node right next to the start node. The second bending point occurred at a location two-third of the width of the node apart from the coordinates of the target node. The two-third width was selected because the value was heuristically determined to be the value with the least overlap when connections were arranged to pass through various points between the nodes. Algorithm 1 summarizes the process described in this section.

Algorithm 1 Orthogonal link path algorithm

```
1: procedure LinkPath (a, b)          Find a path including bending points between node a and b
2:     c ← determine the case from locations of a and b for referencing Table 2
3:     s ← starting connection point of node a from Table 2
4:     t ← target connection point of node b from Table 2
5:     bendingPoints ← empty array
6:     width ← width of a drawn node in pixels
7:     m ← width * 2 / 3              Margin for additional bending point
```

```
8:  if a and b is not adjacent
9:      if dominant direction is vertical
10:         if a and b have no obstacles between them
11:             bendingPoints[0] ← ((a.x + b.x) / 2, s.y)
12:             bendingPoints[1] ← ((a.x + b.x) / 2, t.y)
13:         else
14:             o ← obstacle node between node a and node b
15:             bendingPoints[0] ← ((a.x + o.x) / 2, s.y)
16:             bendingPoints[1] ← ((a.x + o.x) / 2, (b.y + m) or (b.y – m) depending on direction)
17:             bendingPoints[2] ← (t.x, (b.y + m) or (b.y – m) depending on direction)
18:         end if
19:     else
20:         if a and b have no obstacles between them
21:             bendingPoints[0] ← (s.x, (a.y + b.y) / 2)
22:             bendingPoints[1] ← (t.x, (a.y + b.y) / 2)
23:         else
24:             o ← obstacle node between node a and node b
25:             bendingPoints[0] ← (s.x, (a.y + o.y) / 2)
26:             bendingPoints[1] ← ((b.x + m) or (b.x – m) depending on direction, (a.y + o.y) / 2,)
27:             bendingPoints[2] ← ((b.y + m) or (b.y – m) depending on direction, t.y)
28:     end if
29: end if
30: return concatenate points from s, bendingPoints, and t
31: end procedure
```

V. EVALUATION AND ANALYSIS OF THE LAYOUT RESULTS

To quantitatively verify whether the proposed layout method can reduce visual clutter considerably, the layout results for the IEEE standard test system data were verified using the three-link-related quantitative indicators mentioned earlier. We quantitatively compared the system with the same number of nodes and links as the IEEE test system but for which the connections were set arbitrarily between the nodes. Because in the proposed algorithm, the nodes are arranged using a space-filling technique, indicators, such as the size of the space used in the layout in the evaluation, were not included. Furthermore, we used d3.js [3] library to verify the layout results visually and used graphology [18] to apply Louvain clustering and verify the quantitative indicators of the graph.

After clustering the nodes for the IEEE standard test system data, we analyzed the layout results for arranging the nodes in the cluster in ascending order from left to right for each row and for arranging the nodes in the cluster in snake-like order. Fig. 4 displays the result of applying the algorithm to the data with 1062 nodes and 1683 links for visualizing the data. Nine color-coded clusters were apparent because of performing clustering. Connections between clusters barely exist. The result of a smaller system is displayed in Fig. 5. The nodes are arranged in snake-like (SL) order within each cluster.

Next, we analyzed quantitative indicators. In addition to the original data, we created and compared two other data with the same node and link information as the IEEE standard test data. In one of the two data, the nodes were randomly mixed within each cluster (i.e., locally random). In the other data, the entire nodes were randomly mixed regardless of clusters (i.e., globally random). Here, the random data were generated repeatedly (10,000 times) to derive the average.

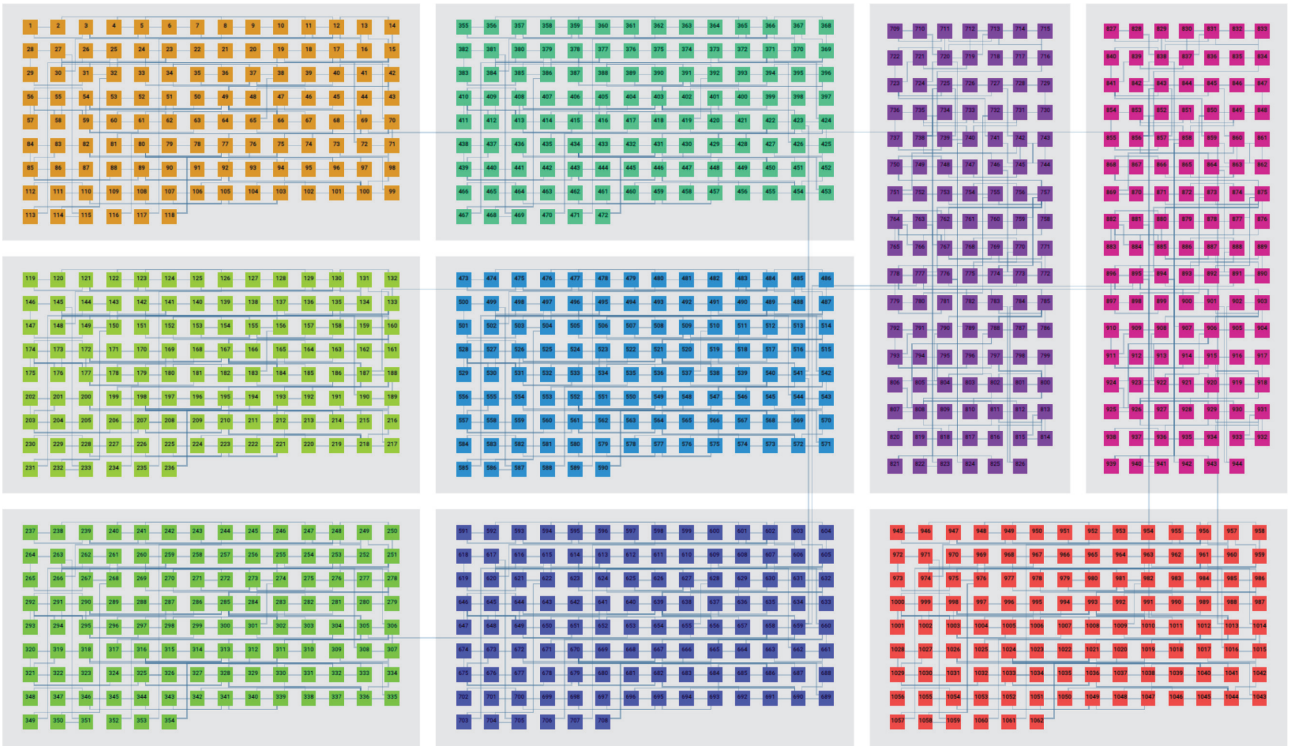

Fig. 4. Result of applying the algorithm to the IEEE test system consisting of 1062 nodes and 1683 links. Each node is represented by a square, and the color of the square is mapped according to the cluster number assigned as a result of Louvain algorithm.

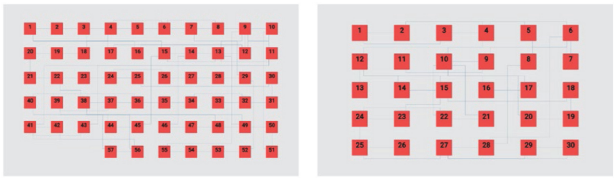

Fig. 5. Results of applying the algorithm to a system consisting of 57 nodes and 80 links (left) and a system consisting of 30 nodes and 41 links (right).

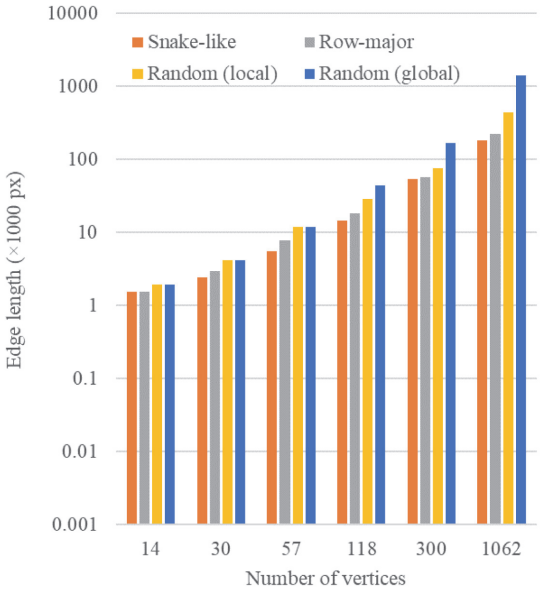

Fig. 6. Total sum of the link lengths when each system is visualized. As the number of nodes increases, the total sum of the link lengths increases exponentially. Snake-like (SL) ordering achieved the best performance, and random (global) achieved the worst performance.

The results based on the link length as the criteria (Fig. 6) revealed that the SL order produced the shortest total sum of the link lengths. In particular, SL exhibited superior performance as the number of nodes increased compared with listing the nodes sequentially in row major (RM). This result implicitly verified our earlier assumption in which nodes with adjacent IDs were assumed to be connected to each other. Next, the total sum of the link lengths increases exponentially as the number of nodes increases when the nodes are arranged randomly within the row cluster (RL) and when the entire nodes are randomly arranged, disregarding the clustering results (RG). As presented in Table 1, the system data with 14 to 57 nodes are grouped into a single cluster. Therefore, the difference between RL and RG is not considerable. However, for data with more than 118 nodes, not breaking the clustered groups yields superior indicator values than mixing the entire nodes.

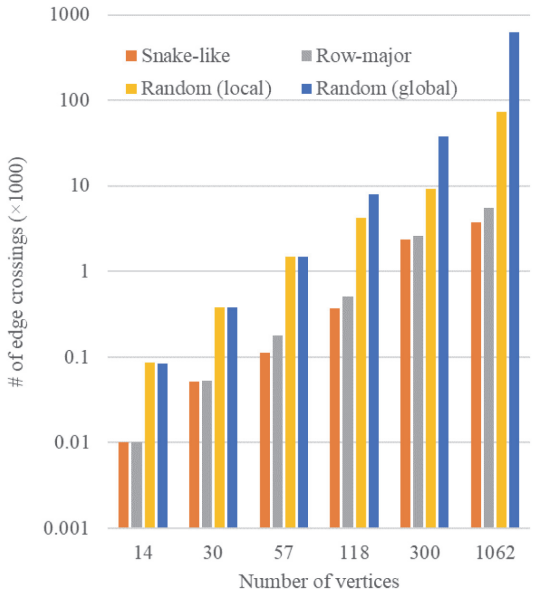

Fig. 7. Total number of link crossings when each system is visualized. SL and RM ordering did not differ for systems with a small number of nodes. However, with the increase in the number of nodes, the number of crossings increased exponentially in the following order: RM, random (local), and random (global).

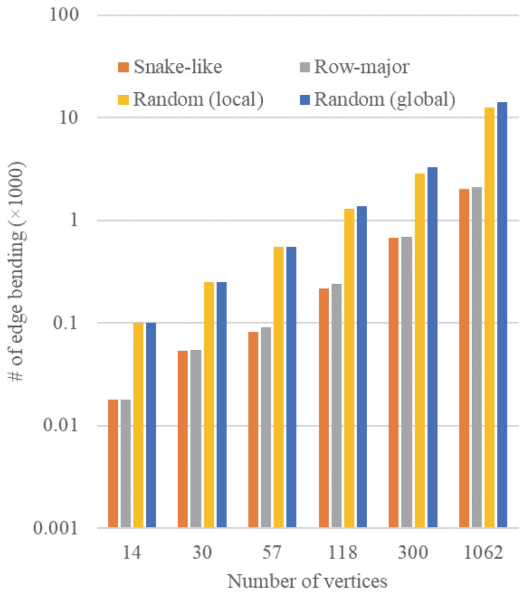

Fig. 8. Total number of link bending when each system is visualized. Unlike the two previous indicators, SL and RM ordering did not differ considerably. In random (local) and random (global), the indicator values are almost ten times larger than the indicator values of the other two cases.

Next, in link crossing (Fig. 7), the overall trend was similar to the aforementioned link length. In this case, the SL achieved the best performance, whereas the RG exhibited the worst performance. In particular, as the number of nodes increased, the ratio of links crossing each other increased considerably when the nodes were arranged randomly. SL and RM do not differ for data with a small number of nodes. However, in SL, the number of link crossings reduced as the number of nodes increased.

Finally, in link bending, the results exhibited a slightly different trend than the previous two indicators (Fig. 8). SL and RM exhibit similar values, and RL and RG exhibit similar values. This result could be attributed to the criteria for link bending. If two nodes are located in adjacent rows or columns, the number of link bending is maintained at two. If the two nodes are located such that they are at least two spaces apart both row- and column-wise, the number of link bending becomes three. Thus, for RM, the row of each node is maintained while only its column is changed, compared with the SL layout. Therefore, the indicator values between RM and SL do not differ markedly. By contrast, in RL and RG, many occurrences of nodes being arranged two spaces or more apart occur in both the horizontal and vertical directions as the nodes are mixed. Therefore, the number of link crossings increased considerably as many node pairs emerged require three-link bending.

VI. DISCUSSION

In the squarified treemap applied in this study, each cluster is arranged in a square shape, if possible. However, because higher priority was given to preserving the shape, the connection relationship between the clusters could not be considered. Therefore, link connections sometimes crossed over other clusters. Thus, cases in which the link lengths were long occurred, and many crossings with other links were apparent. For this case, the cluster layout results can be improved by deriving the connection relationship between clusters based on the clustering results and subsequently applying the adjacency-preserving spatial treemap [19] of Buchin et al. in which the adjacency relationships are preserved.

In arranging the links, the number of links passing through the same path were reduced as much as possible. Therefore, we arranged the link to pass through the middle point between the nodes if an adjacent row and column exist. If more than one row or column was present in the middle, we arranged the link to pass through the point at two-third of the width of the node apart from the node. Thus, we achieved the objective of the technique to some extent. This part can be improved if the gap between the nodes is divided and several links pass through the area, as in the research by Freivalds and Glagoļevs [20]. However, too many links passing through a certain area could increase visual complexity. Therefore, adjustment should be performed according to visual analysis the user plans to perform.

Because the arrangement of clusters through clustering and treemap and the node layout and link layout process within each cluster are performed sequentially, the time complexity of the process can be examined in stages. First, in the Louvain algorithm used in this study, various modified versions of this algorithm exist. A study by Blondel [21] introduced an algorithm that has a time complexity of O(m) (where *m* denotes the number of links). Next, as mentioned by Scheibel [22], the layout process performed using treemap is rarely discussed in the information visualization field. Therefore, this process was excluded from analysis. Next, the node layout process using the SL order proposed in this study can arrange the nodes sequentially when the number of rows and columns are derived based on the number of nodes in each cluster. This process can be performed in O(n) (where *n* denotes the number of nodes). Finally, the process of finding the path of the links can be performed in O(m) because this process only should be sequentially repeated as many times as the number of links. In the IEEE standard test system data with 1062 nodes and 1683 links, the layout was derived, and rendering was completed within 2 seconds in a computing environment equipped with AMD Ryzen 3800X CPU. For smaller data, the results could be obtained within 1 second.

VII. CONCLUSION AND FUTURE WORK

In this study, an algorithm was proposed to visualize the connection relationship of a power system for the visual analysis of the power system. Two aspects were considered in this process. Visual analysis tools were constructed in the form of multiple-coordinated views because of the characteristics of the multivariate analysis study. Therefore, the proposed algorithm was designed to represent the connection relationship of the system in a limited space. The orthogonal layout that power system researchers were familiar with was configured. In the future, real system data can be used instead of test system data for empirical studies. In this study, we assumed a high probability that nodes with adjacent unique IDs were connected. However, in future research, this issue will be resolved by adopting the matrix reordering method proposed by Behrisch et al. [17] or another reordering method that can obtain similar results. Moreover, the data used in this study are represented by a multi-graph in which multiple connections exist between a pair of nodes. However, duplicate connections were excluded, and one connection was represented in the visualization process. This part can be investigated to study the link layout algorithm based on the purpose of the task of the user. The algorithm could also be applied to real-world data in the follow-up study.

AKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2021R1C1C1009327).

REFERENCES

1. R. D. Christie. "Power Systems Test Case Archive." (accessed 14 September 2022, 2022).
2. "Power World Corporation Homepage." PowerWorld Corporation. (accessed 14 September 2022.
3. M. Bostock, V. Ogievetsky, and J. Heer, "D³ data-driven documents," *IEEE transactions on visualization and computer graphics,* vol. 17, no. 12, pp. 2301-2309, 2011.
4. M. Franz, C. T. Lopes, G. Huck, Y. Dong, O. Sumer, and G. D. Bader, "Cytoscape. js: a graph theory library for visualisation and analysis," *Bioinformatics,* vol. 32, no. 2, pp. 309-311, 2016.
5. Y. Wang, Z. Jin, Q. Wang, W. Cui, T. Ma, and H. Qu, "Deepdrawing: A deep learning approach to graph drawing," *IEEE transactions on visualization and computer graphics,* vol. 26, no. 1, pp. 676-686, 2019.
6. O.-H. Kwon and K.-L. Ma, "A deep generative model for graph layout," *IEEE Transactions on visualization and Computer Graphics,* vol. 26, no. 1, pp. 665-675, 2019.
7. B. Shneiderman, "Tree visualization with tree-maps: 2-d space-filling approach," *ACM Transactions on graphics (TOG),* vol. 11, no. 1, pp. 92-99, 1992.
8. M. Bruls, K. Huizing, and J. J. v. Wijk, "Squarified treemaps," in *Data visualization 2000*: Springer, 2000, pp. 33-42.
9. H.-Y. Wu, M. Nollenburg, and I. Viola, "Multi-level area balancing of clustered graphs," *IEEE Transactions on Visualization and Computer Graphics,* 2020.
10. S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow, "Hola: Human-like orthogonal network layout," *IEEE transactions on visualization and computer graphics,* vol. 22, no. 1, pp. 349-358, 2015.
11. H.-Y. Wu, M. Nöllenburg, F. L. Sousa, and I. Viola, "Metabopolis: scalable network layout for biological pathway diagrams in urban map style," *BMC bioinformatics,* vol. 20, no. 1, pp. 1-20, 2019.
12. U. Rüegg, S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow, "Stress-minimizing orthogonal layout of data flow diagrams with ports," in *International Symposium on Graph Drawing*, 2014: Springer, pp. 319-330.
13. H. Gibson, J. Faith, and P. Vickers, "A survey of two-dimensional graph layout techniques for information visualisation," *Information visualization,* vol. 12, no. 3-4, pp. 324-357, 2013.
14. S. Chakrabarti and E. Kyriakides, "PMU measurement uncertainty considerations in WLS state estimation," *IEEE Transactions on Power Systems,* vol. 24, no. 2, pp. 1062-1071, 2009.
15. W. Zheng, W. Wu, A. Gomez-Exposito, B. Zhang, and Y. Guo, "Distributed robust bilinear state estimation for power systems with nonlinear measurements," *IEEE Transactions*

on Power Systems, vol. 32, no. 1, pp. 499-509, 2016.
16. N. Dugué and A. Perez, "Directed Louvain: maximizing modularity in directed networks," Université d'Orléans, 2015.
17. M. Behrisch, B. Bach, N. Henry Riche, T. Schreck, and J. D. Fekete, "Matrix reordering methods for table and network visualization," in *Computer Graphics Forum*, 2016, vol. 35, no. 3: Wiley Online Library, pp. 693-716.
18. G. Plique. "Graphology, a robust and multipurpose Graph object for JavaScript." Zenodo. https://doi.org/10.5281/zenodo.5681257 (accessed September 12, 2022.
19. K. Buchin, D. Eppstein, M. Löffler, M. Nöllenburg, and R. Silveira, "Adjacency-preserving spatial treemaps," *Journal of Computational Geometry,* vol. 7, no. 1, pp. 100-122, 2016.
20. K. Freivalds and J. Glagoļevs, "Graph compact orthogonal layout algorithm," in *International Symposium on Combinatorial Optimization*, 2014: Springer, pp. 255-266.
21. V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment,* vol. 2008, no. 10, p. P10008, 2008.
22. W. Scheibel, D. Limberger, and J. Döllner, "Survey of treemap layout algorithms," in *Proceedings of the 13th international symposium on visual information communication and interaction*, 2020, pp. 1-9.

San Hong

San Hong is currently pursuing a bachelor's degree in Computer Science at Soongsil University, South Korea. His research interests include human-computer interaction, information visualization.

Sangjun Park

Sangjun Park is currently pursuing a bachelor's degree in Computer Science at Soongsil University, South Korea. His research interests include human-computer interaction, data visualization, android applications and UX optimization.

Chanil Kim

Chanil Kim is currently pursuing a bachelor's degree in Computer Science at Soongsil University, South Korea. His research interests include human-computer interaction and information visualization.

Hyunjoo Song

Hyunjoo Song is currently an Assistant Professor at the School of Computer Science and Engineering, Soongsil University, South Korea. His research interests include human-computer interaction, information visualization, visual analytics, eye tracking and health informatics.