Optimization of Hierarchical Graph Layout with a Genetic Algorithm and Sprawl/Clutter Metrics

Ayana Murakami
Ochanomizu University
Tokyo, Japan
murakami.ayana@is.ocha.ac.jp

Takayuki Itoh
Ochanomizu University
Tokyo, Japan
itot@is.ocha.ac.jp

***Abstract*—Graph layouts are useful for visualizing relationships between data entities. Nodes represent each entity, and edges represent the relationships between the nodes. Hierarchical graph visualization is known as an efficient method to represent the overview of large-scale graphs, where some clusters are generated depending on node properties and are drawn as a set of multiple nodes (meta-nodes). However, it is often difficult to determine the layouts of nodes and edges of large-scale graphs, including hierarchical graphs. Although many graph layout methods have been proposed so far, the quality of the layouts by many of such methods including force-directed layout methods strongly depends on the initial positions of the layouts. Therefore, it is often a difficult task to obtain desired layouts.**

This paper presents a layout optimization method for hierarchical graphs using a genetic algorithm. Specific metrics for hierarchical graph layouts are used to evaluate the generated layout. This method makes users easy to select a favorite layout from several layouts which are optimized based on the evaluation results. First, hierarchical graph layouts are generated by applying an existing method multiple times. Then, they are optimized using a genetic algorithm. Users can select their favorite layouts from several optimized layouts. Consequently, this method does not require adjusting initial positions of the layouts to obtain good layouts. The paper also introduces examples using human relationship graph datasets, including a co-authorship dataset.

***Index Terms*—graph layout, hierarchical graph, optimization**

I. Introduction

Graphs represent data elements as nodes and their relationships as edges. In recent years, graph visualization methods have been used in a variety of fields. The readability of a graph depends greatly on the layout of its nodes. In particular, hierarchical graphs consisting of multiple sets of nodes (meta-nodes) are known as an effective way to visualize the overview of large graphs. However, generating a highly readable layout is extremely difficult due to the complexity of connection relations and the need to consider meta-information.

Many methods have been studied for generating graph layouts, including methods that apply force-directed models and dimensionality reduction techniques. Here, each of these methods has its own strong characteristics, and it is not always possible to achieve a flexible layout. In contrast, graph layout methods based on deep learning have been actively discussed in recent years. As an example, graph layout generation methods that treat graphs as sequences and use LSTM (Long Short Term Memory) and RNN (Recurrent Neural Network) as learning models have been presented. However, these studies have yet to generate layouts for hierarchical graphs, and taking much time to train large-scale graphs remains a significant challenge.

Graph layout evaluation is another issue that has been discussed for a long time, and many studies have been published on both numerical and aesthetic evaluation. In particular, numerical evaluation metrics that focus on graph structure based on graph theory, rather than graph visualization results, have been discussed a lot. However, most of them are for non-hierarchical graphs. There are very few studies on evaluation methods specialized for visualization results of hierarchical graphs. In contrast, Liu et al. [6] proposed Sprawlter, a new numerical evaluation criterion specialized for hierarchical graphs. Remark that they used Sprawlter metric only for evaluating the layout results of hierarchical graphs, not for optimizing layouts.

In this paper, we propose a method of hierarchical graph layout optimization using mathematical optimization techniques to solve the above problems. This method first applies existing hierarchical graph layout methods to the given dataset to generate several different layouts, and then executes the optimization process with these layouts as initial layouts. The Sprawl and Clutter metrics are applied as objective functions of the optimization. Since there are two objective functions, the problem is treated as a multi-objective optimization problem. We adopt a genetic algorithm (GA) to optimize the layout. The advantages of GA are that they can search a wide solution space and are unlikely to fall into local solutions.

In this paper, we apply GA to obtain a set of optimized hierarchical graph layouts, and then visualize the evaluation results of each layout in a scatterplot. This scatterplot not only produces a layout with a moderately high evaluation, but also allows the user to select a favorite layout depending on the user's visualization purpose and preferences.

II. Related Work

A. Graph Layout

Graph layout is actively discussed as a technique for visualization that represents the relationship of nodes by their edges. In particular, a graph in which multiple nodes with tight connections or similar attributes form a cluster is called a hierarchical graph. There have been many hierarchical graph layout methods. As an example, the method presented by Itoh et al. [5] performs clustering according to the attributes of each node so that nodes belonging to the same cluster are placed close to each other. Fig. 1 shows an example of a graph layout by this method, where nodes belonging to the same meta-node are drawn in the same color.

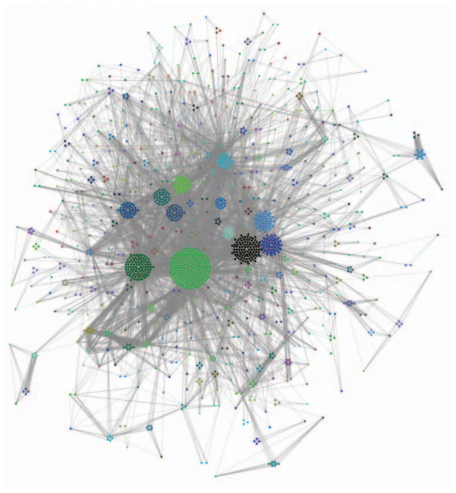


Fig. 1: An exapmle of graph layout drawn by Koala algorithm [5].

B. Evaluation of Graph Layout

Evaluation of graph layout is an important issue and has been actively discussed in recent years. There have been two main perspectives of graph layout evaluation metrics: first, metrics based on the connectivity of the graph, and second, metrics related to the aesthetics of the layout. Purchase [7] has selected five items related to edge crossing and graph layout symmetry as examples of metrics that can lead to improved readability. After comparative experiments, they concluded that the number of intersecting edges (Crosses) had the greatest impact on aesthetics among the five items.

Here, different visualization objectives require different indicators to evaluate the graph layout. For example, Ware et al. [10] analyzed the important indicators for the task of finding the shortest path in a graph and concluded that indicators of graph continuity had the greatest impact on the task. From the above, multiple types of metrics should be simultaneously considered for graph layout. Moreover, the important metrics also differ depending on the target of visualization.

These metrics are basically for non-hierarchical graphs; in other words, there are few metrics for evaluating graph layouts specific to hierarchical graphs. To address this issue, Liu et al. [6] proposed Sprawlter, a numerical evaluation metric for layout appearance, which consists of two metrics: Sprawl, which evaluates space waste, and Clutter, which evaluates cluttering among nodes and edges.

Conventional graph layout methods require adjustments of multiple parameters to achieve better layouts. There are multiple evaluation functions; in other words, there is no absolute single important indicator. In addition, there is often a trade-off relationship between such evaluation indices. In addition, there is no single best layout for a graph; in other words, there are many "good" layouts of a certain level among countless graph layouts created by adjusting parameters. This graph layout problem is one of the multi-objective optimization problems in a multimodal solution space [9]. There have been various methods for solving multi-objective optimization problems, and each method has both advantages and challenges. In this paper, we adopt mathematical optimization methods.

C. Optimization method and Graph layout

It is very important to select the appropriate algorithm according to the feature of the optimization problems. The feature of graph layout problems is that they have a lot of local-optimal solutions in a high-dimensional solution space.

Many algorithms for multi-objective optimization problems have been studied, and there are many papers comparing these algorithms. There are two major solutions: One of the solutions is to transform multiple objective functions into a single objective function by weighting them, and then finding their optimal solution. The other is to search for a set of solutions that optimize each of the multiple objective functions. According to Gunantara et al. [3], the latter method is more efficient in a high-dimensional solution space.

In multi-objective optimization problems, each objective function is often in a trade-off relationship, and therefore, there are multiple local-optimal solutions usually. Pareto solutions, a set of solutions that outperforms other solutions, are often searched for as candidate solutions of the multi-objective optimization problem [4]. As an example, Fig. 2 illustrates the search for a nondominated solution to the problem of minimizing two objective functions f_1 and f_2. The set of superior solutions indicated by the red dots is called the Pareto solution. The curve drawn by the Pareto solutions is called the Pareto Front.

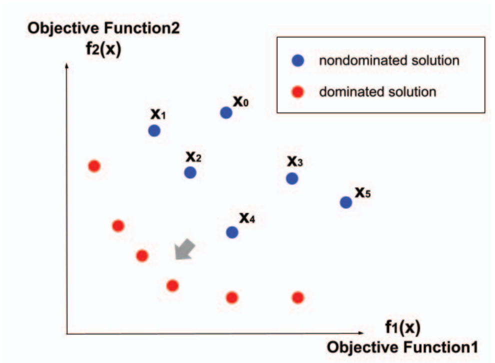


Fig. 2: Pareto Front in 2-dimentional search space. We suppose that the smaller values of both axises are, the better the solution is.

In this paper, we adopt GA [2] as a method for obtaining Pareto solutions. GA simultaneously evaluates and selects an arbitrary number of candidate solutions. Therefore, it can efficiently search for solutions in a wide and high-dimensional solution space.

The Nondominated Sorting Genetic Algorithm (NSGA) proposed by Srinivas et al. [8] is one of GA and has been applied to many optimization problems. Deb et al. [1] proposed NSGA-II, which improves the computational complexity of NSGA. It introduces elite selection as a method for selecting the next generation of genes, so we can search for diverse solutions faster than NSGA.

III. Optimization of Graph Layout

This section presents our method for the optimization of the layout of hierarchical graphs. In our proposed method, first of all, multiple layouts of a hierarchical graph are generated by iterating a layout algorithm [5]. The algorithm is called Koala, in the following description, Koala means the algorithm to draw a hierarchical graph layout. Graph Layouts generated by force-directed graph drawing model, such as Koala, is influenced by initial position of nodes. The better initial position is, the better graph layout is generated. Therefore, in this paper, we optimize graph layouts by optimizing initial position of meta-nodes. We adopt NSGA-II in order to optimize them. In an optimizing process, Sprawl and Clutter metrics evaluate the hierarchical graph layouts respectively. Finally, an arbitrary layout of the hierarchical graph layout is selected from the Pareto solutions.

Fig. 3 shows the processing steps of this method. Each process is described in detail in the following sections.

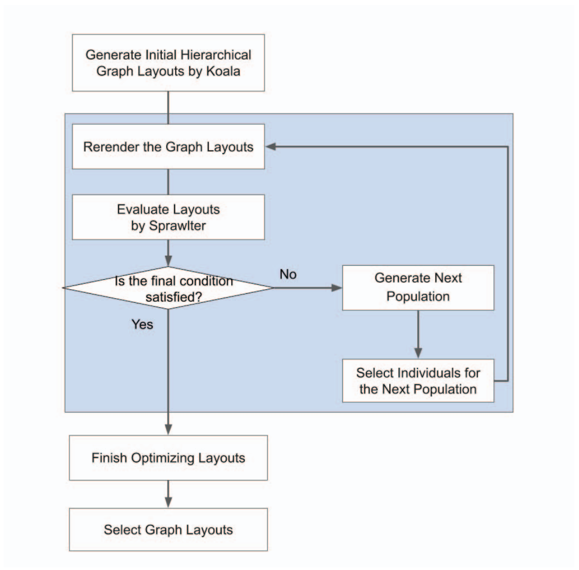

Generate Initial Hierarchical Graph Layouts by Koala
Rerender the Graph Layouts
Evaluate Layouts by Sprawlter
Is the final condition satisfied?
No
Yes
Generate Next Population
Select Individuals for the Next Population
Finish Optimizing Layouts
Select Graph Layouts

Fig. 3: Processing procedure of GA.

A. Hierarchical Graph Layout Drawn by Koala

In our implementation, hierarchical graph layouts are generated with Koala algorithm [5]. In the algorithm, important nodes which have connections to multiple large clusters are separated, and clusters are placed based on the distance. This improves the visibility of connections of such important nodes. This clusters of nodes is called mata-nodes. In this process, initial position of each nodes are determined romdomly. Therefore, a layout generated by Koala is differenct each time. By executing Koala algorithm several times, several hierarchical graph layouts are generated.

The cohesiveness of a meta-node is determined by a parameter. Fig. 4 shows examples of hierarchical graph layouts drawn with different degrees of aggregation. Considering the visibility of the meta-nodes, we adopt the cohesiveness shown in Fig. 4b.

B. Optimization by a Genetic Algorithm

We can get a variety of layouts by adjusting the initial positions of meta-nodes. A set of graph layouts is generated by iteratively executing the method. The set of layouts is optimized by NSGA-II. We call this set of layouts a generation. In optimizing process, each layouts in a generation is evaluated with Clutter and Sprawl metrics respectively. These metrics of Sprawlter [6] are used as the objective functions of the optimization process.

GA works with a population of individuals. In this case, an individual is represented by a graph layout, and its chromosome is represented by x-y coordinates of meta-nodes consisting of the graph layout.

IV. Experiment

A. Dataset

We visualize a paper co-authorship dataset where the papers are published at the NERC Biomolecular Analysis Facility (NBAF) from 1998 to 2013. Here, authors correspond to nodes while co-authorships are represented as edges. The dataset consists of 1821 authors and 564 papers, each of the papers has a 12-dimensional feature vector.

The final size of the dataset used in this experiment after the preprocessing that eliminates isolated nodes is shown in Table I.

TABLE I: Results of data preprocessing.

	before preprocessing	after preprocessing
Nodes	1821	1538
Edges	9692	8040

The parameters of GA used in this experiment are shown in Table II. As mentioned above, since a chromosome has the x-coordinate and y-coordinate of all meta-nodes of the graph layouts, its length is equal to twice the number of meta-nodes. In this experiment, our implementation generated 460 clusters (meta-nodes), so the length of a gene in this experiment is $460 \times 2 = 920$.

B. Result

Fig. 5 shows a comparison of the evaluation result of the initial generation and each generation of the optimization process. The scatterplot represents comparisons between the

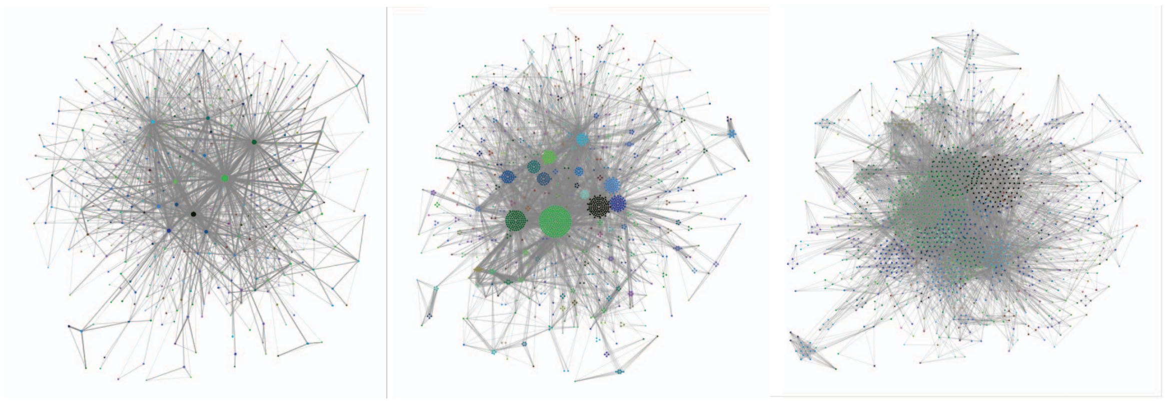


Fig. 4: Examples of hierarchical graph layouts generated by Koala. Depends on the cohesive level parameter, aggregation degree of nodes in a meta-node is different. The more highly cohesive nodes are, the smaller a meta-node become.

TABLE II: Parameters in this experiments.

Parameter	Value
Length of Gene	920 ($= 460 \times 2$)
Population in a generation	20
Crossover Probability	0.90
Mutation Probability	0.10

initial and the first, tenth, and final generations. The horizontal axis represents the Sprawl value while the vertical axis represents the Clutter value. Each point represents a single individual, a single hierarchical graph layout. In the scatterplot, the cross dots represent the values of individuals in the initial generation, and the circle dots represent the values of individuals in the each generation. Note that the number of the initial generation does not correspond to the counterpart of other generations, because GA generates different individuals from the initial generation through genetic manipulation.

In Fig. 5, the color of circle dots changes from red, green, to blue with increasing generation. The evaluation results for the whole generation, represented by both Clutter and Sprawl values, show improvement. A comparison between the initial population (blue cross dot) and the last population (blue circle dot) reveals that the overall evaluation value of the solution set improves with each successive generation.

Next, we display the evaluation score of each graph layouts, which is corresponding to several points plotted on the scatterplot shown in Fig. 5. Fig. 6 shows an example of the hierarchical graph layout before the application of GA (= blue cross dot in Fig. 5). Fig. 7 shows an example of the layout after the application of GA (= blue circle dot in Fig. 5). Different meta-nodes are assigned different colors. The caption number for each figure corresponds to the number assigned to the point in the scatterplot shown in Fig. 5.

C. Evaluation of Optimized Graph Layouts

We compare the graph layouts shown in Section IV-B before and after the optimization, and evaluate the effectiveness of the

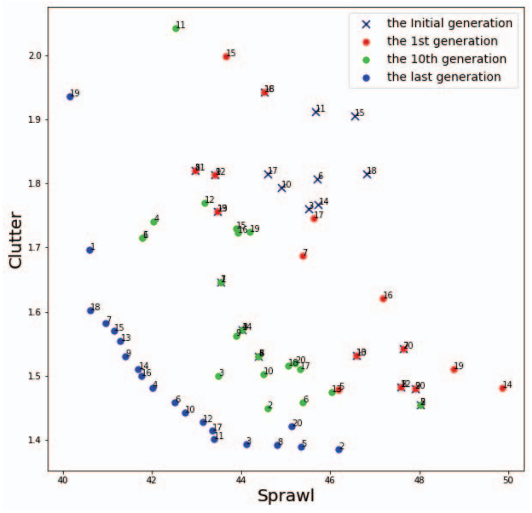


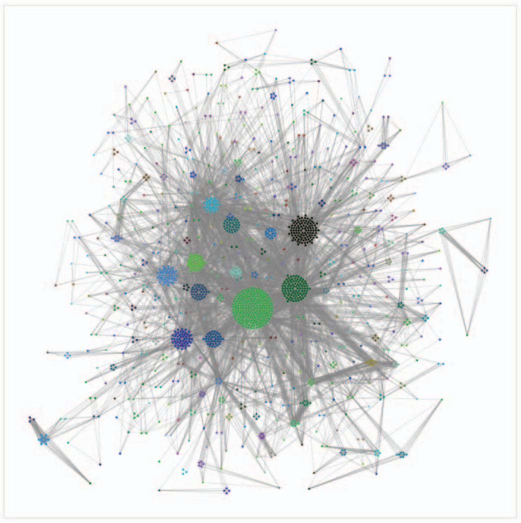
Fig. 5: Processing procedure of GA.

optimization.

1) Appearance Evaluation: In the hierarchical graph layout before optimization, there are some overlappings of nodes and edges. Overlaps of elements make worse Clutter score. For example, in Fig. 5, focusing on the blue dots of the first generation, graph layout no. 4 is highly evaluated among them. However, we can see some overlapping of nodes and edges in the graph layout shown in Fig. 6a.

Fig.8 is an enlarged view of the center part of Fig. 6a. In the area circled by the pink ellipse, we can see that two meta-nodes are interfering with each other. Next, in the area circled by the red oval, there are many small meta-nodes consisting of a small number of nodes. Nodes of different colors belong to different meta-nodes. The close proximity of nodes of different colors means that the distance between meta-nodes is close.

On the other hand, in the hierarchical graph layout after


(a) Layout no.4.

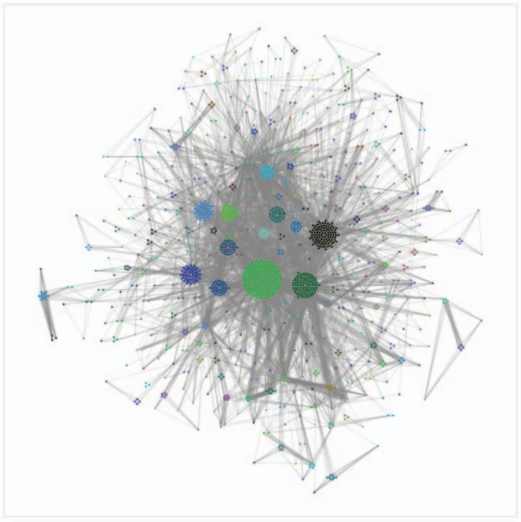
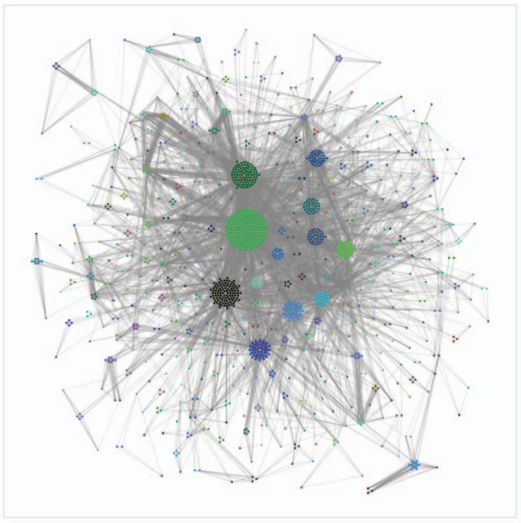

(b) Layout no. 14.

Fig. 6: Examples of layouts before optimization.


(a) Layout no.9.

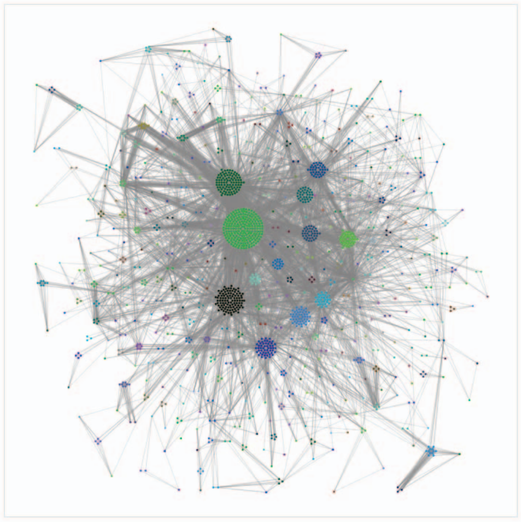

(b) Layout no. 18.

Fig. 7: Examples of layouts after optimization.

optimization shown in Fig. 7, such overlaps of meta-nodes are hardly observed. This means that the improvement in the evaluation of Clutter due to optimization was visually confirmed in the graph layout as well.

Next, among the graph layouts shown in Fig. 7, Fig. 7b has a high Sprawl evaluation, while Fig. 7a is a graph layout with low Sprawl evaluation. From a space-wasting perspective, comparing these two layouts, little difference is visually recognizable. This means that even though there is numerical improvement in the evaluation of Sprawl, the visual improvement is small and difficult to recognize.

2) Numeric Evaluation: We calculated the Sprawl and Clutter values of individuals of the initial and final generations, and compared them between them. The comparative results of the average of each generation is shown in Table III. The evaluation distribution of each layouts in a generation is shown in Fig. 9.

TABLE III: Comparison of Sprawl and Clutter averages between first and last generations.

	Sprawl	Clutter
First Generation	45.48	1.717
Last Generation	42.62	1.504

According to Fig. 9, the maximum value of the results is not improved by before and after optimization. In other words, there are several undesirable graph layouts in every generation. On the other hand, the values of Table III and the third quartiles of Fig. 9 have decreased significantly. This indicates

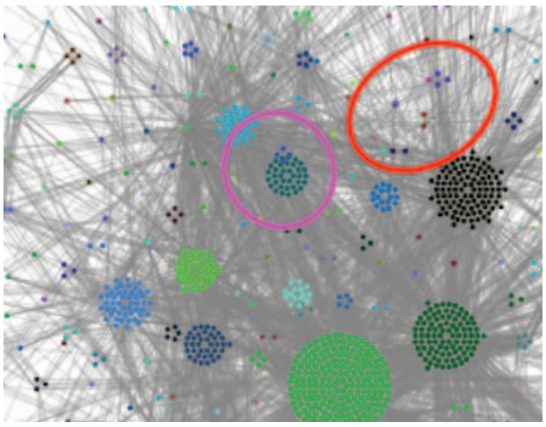


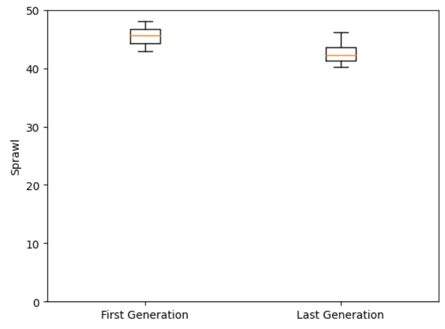
Fig. 8: An enlarged view of the center of the optimized layout shown in Fig.6a.

that the number of highly evaluated layouts in one generation has increased overall. In summary, the application of GA enable us to increase a number of highly evaluated hierarchical graph layouts. However, some layouts in one generation got low evaluation even after optimization. Therefore, it is very important to select an appropriate layout from the optimized hierarchical graph layouts at last.

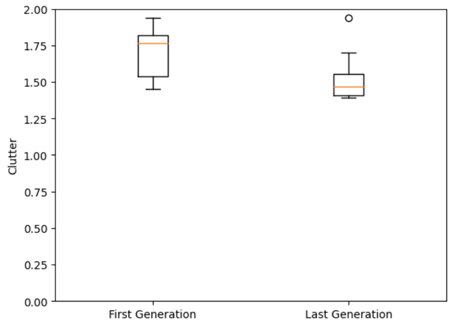
V. Conclusion

In this paper, we addressed the layout optimization of hierarchical graphs as a multi-objective optimization problem and applied an optimization method for hierarchical graph layout using GA. For the objective function, we applied Sprawlter, a numerical evaluation formula specialized for hierarchical graph layouts. The obtained hierarchical graph layouts were compared before and after optimization, and we found that the method is effective both numerically and visually.

For future work, we would like to apply more objective functions for GA that focuses not only on Sprawl and Clutter, but also on the metrics related to the connectivity of the graph. In particular, the hierarchical graph layout method employed in this paper [5], ensures a certain degree of space utility by internal parameters, so all layouts have high evaluation values for Sprawl, and numerical differences of Sprawl among layouts are small. In addition, it is difficult to recognize the superiority among layouts generated by the proposed method while looking at them. Therefore, it is necessary to apply different objective functions in order to achieve more drastic improvements.



(a) Comparison of the Sprawl metric values. The Sprawl metric evaluates space waste. The larger the Sprawl metric values are, the larger space on canvas is wasted.



(b) Comparison of the Clutter metric values. The Clutter metric captures cluttering among nodes and edges. The larger the Clutter metric values are, the more overlaps of nodes and the more cossings of edges a loyout has.

Fig. 9: Comparison of evaluation values of before and after the optimization. Left box represents distribution of evaluation values of layouts before optimization, Right box represents after optimization. For each metrics, larger number are worse.

References

[1] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II.”, IEEE transactions on evolutionary computation, vol. 6, no. 2, pp. 182-197, 2002.
[2] C. M. Fonseca, P. J. Fleming, “Genetic Algorithms for Multiobjective Optimization: Formulation”, Discussion and Generalization., Proceedings of the 5th international coference on genetic algorithms, vol. 93, pp. 416-423, 1993.
[3] N. Gunantara, “A Review of Multi-Objective Optimization: Methods and its Applications.”, Cogent Engineering, vol. 5, no. 1, 2018.
[4] T. Hiroyasu, M. Miki, S. Watanabe, T. Sakoda, J. Kamiura. “Evaluation of Genetic Algorithm for Objective Computation Methods.”, The Science and Engineering Review - Doshisha University, vol. 43, no. 1, pp. 41-52, 2002.
[5] T. Itoh, K. Klein, “Key-node-Separated Graph Clustering and Layouts for Human Relationship Graph Visualization.”, IEEE Computer Graphics and Applications, vol. 35, no. 6, pp. 30-40, 2015.
[6] Z. Liu, T.Itoh, J. Q. Dawson, T. Munzner. “The Sprawlter Graph Readability Metric: Combining Sprawl and Area-Aware Clutter”, IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 6, pp. 2180-2191, 2020.
[7] H. Purchase, “Which Aesthetic Has the Greatest Effect on Human Understanding?.”, In International Symposium on Graph Drawing, pp. 248-261, 1997.
[8] N. Srinivas, K. Deb, “Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms.”, Evolutionary computation, vol. 2, no. 3, pp. 221-248, 1994.
[9] Y. Tian, L. Si, X. Zhang, R. Cheng, C. He, K. C. Tan, Y. Jin, “Evolutionary Large-Scale Multi-Objective Optimization: A Survey.”, ACM Computing Surveys (CSUR), vol. 54.8, pp. 1-34, 2021.
[10] C. Ware, H. Purchase, L. Colpoys, M. McGill, “Cognitive Measurements of Graph Aesthetics.”, Information visualization, vol. 1, no. 2, pp. 103-110, 2002.