Information Sciences 181 (2011) 4253–4272

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Mental map preserving graph drawing using simulated annealing[☆]

Chun-Cheng Lin[a], Yi-Yi Lee[b], Hsu-Chun Yen[b,*]

[a] Department of Industrial Engineering and Management, National Chiao Tung University, Hsinchu 300, Taiwan, ROC
[b] Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan, ROC

ARTICLE INFO

Article history:
Received 26 May 2009
Received in revised form 19 December 2010
Accepted 2 June 2011
Available online 6 June 2011

Keywords:
Mental map
Simulated annealing
Graph drawing

ABSTRACT

Visualizing graphs has been studied extensively in the community of graph drawing and information visualization over the years. In some applications, the user is required to interact with a graph by making slight changes to the underlying graph structure. To visualize graphs in such an interactive environment, it is desirable that the differences between the displays of the original and the modified graphs be kept minimal, allowing the user to comprehend the changes in the graph structure faster. As the *mental map* concept refers to the presentation of a person's mind while exploring visual information, the better the mental map is preserved, the easier the structure change of a graph is understood. It is somewhat surprising that preserving the user's *mental map* has largely been ignored in the graph drawing community in the past. We propose an effective mental-map-preserving graph drawing algorithm for straight-line drawings of general undirected graphs based on the simulated-annealing technique. Our experimental results and questionnaire analysis suggest this new approach to be promising.

1. Introduction

Graphs are widely recognized as a popular and useful model for capturing a wide variety of real-world concepts in application areas such as social networks, databases, discrete event systems, biological pathways, and many more [17]. Through a nice visual display of a graph, the user is likely to gain a better insight into the information conveyed by the graph. As it is virtually infeasible to manually draw graphs of high complexity in structure and size, many sophisticated automatic drawing methods for displaying complicated graphs have been developed over the years (see, e.g., [9,10,20]). The interested reader is referred to [1,21] for nice surveys on graph drawing.

In order to find nice drawings, graph drawing algorithms are often designed to meet certain *aesthetic criteria* (see, e.g., [31]), which specify the desired graphical structures and drawing properties. Such aesthetic criteria include minimizing the number of edge crossings or bends, minimizing the drawing area, maximizing the degree of symmetry, and many more. Unfortunately, the problem of simultaneously optimizing two or more criteria, in many cases, is NP-hard. Although the computational efficiency as well as the drawing quality (i.e., the extent to which the output drawings conform to the desired aesthetic criteria) remain the most popular measures for judging the effectiveness of a drawing algorithm, more and more attention has been given to the issue of preserving the *mental map* [30] in the framework of interactive graph drawing.

In some applications, the user can interact with a graph by applying operations such as adding/deleting one or more nodes/edges to the graph on a regular basis. To cope with this type of dynamically changing graphs, most automatic drawing strategies rely on using an existing static graph drawing algorithm to redraw each modified graph from scratch. Although

[☆] A preliminary version of this work was presented at Asia Pacific Symposium on Information Visualization 2006, February 1–3, 2006, Tokyo, Japan.
[*] Corresponding author.
E-mail address: yen@cc.ee.ntu.edu.tw (H.-C. Yen).

0020-0255/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2011.06.005

such strategies are convenient and simple, they suffer from the following drawback. Since the user is already familiar with the original drawing, applying an existing static graph drawing algorithm to the modified graph may render the output drawing ill-behaved as far as preserving the user's "mental map" is concerned. To see this, consider Fig. 1. Fig. 1(a) is the drawing of a graph produced by the well-known spring algorithm [12]. Suppose we add node 4 and connect it to the remaining nodes of the graph. Fig. 1(b) and (c) are two possible drawings of the modified graph produced by the spring algorithm under different settings of spring lengths. Clearly, Fig. 1(c) is a better one as far as preserving the mental map is concerned, since it inherits more information from the original drawing (i.e., Fig. 1(a)). Fig. 2 (originally given in [22]) shows a more dramatic example illustrating the importance of preserving the mental map. The initial graph (Fig. 2(a), excluding the thick curve) consists of a chain of quadrangles, which are displayed nicely by the "Circular with Radial" (CR for short) layout algorithm [36]. When we connect the two outermost nodes by a single edge which is marked by the thick curve in Fig. 2(a), all the quadrangles collapse into one in the layout redrawn by the CR algorithm (see Fig. 2(b)). Note that the basic idea of the CR algorithm is to place all the nodes on a circle and then perform a crossing minimization heuristic, which still results in at least $n/4$ crossings (where n is the number of nodes) in this example. Unfortunately, the new drawing suffers from a rather small angular resolution such that it is hard to recognize the original structure. An alternative layout is displayed in Fig. 2(c) which shows a complete ring of quadrangles as it is constructed from the initial drawing. The above two examples suggest that using a static drawing algorithm to draw a modified graph from scratch may result in something that looks completely different from the original, causing the user to have to spend a considerable amount of extra time to comprehend the new drawing.

Eades, Lai, Misue, and Sugiyama [13] were the first to propose the *mental map* concept of graph drawing, which can be thought of as the presentation of a person's mind while exploring visual information of a graph. What makes the preservation of the mental map important in graph drawing is that the user is able to quickly recognize the redrawn layout of the modified graph, based on his/her understanding of the original layout. Most of the existing algorithms for the problem of preserving the mental map focus on *preserving the orthogonal ordering* (see, e.g., [25]) or adding constraints to the input graph (see, e.g., [2,19]). For two nodes u and v in a graph, the preservation of the orthogonal ordering means the relative above/below and left/right positions of the nodes remain unchanged after the graph is redrawn. Preserving the user's mental map is sometimes referred to as a stability problem (see, e.g., [26]). Recently, Brandes and Pampel [3] have showed that it is NP-hard to preserve the orthogonal ordering even for the simplest graphs (paths). For specific drawing styles taking mental map preservation into account, the reader is referred to [22] for circular drawing and [16] for online dynamic graph drawing. Experimental studies concerning the effect of preserving the mental map can be found in [32,33,35]. The importance of preserving the mental map can also be found in applications including distributed mobile object environments [15], online learning environments [37], and more.

The basic requirement of a graph drawing algorithm is to produce an aesthetically pleasing layout in 2-D or 3-D, taking the abstract representation of a graph as the input. In Davidson and Harel's work [6], a nice-looking graph drawing refers to a drawing which: (1) distributes nodes evenly, (2) makes edge-lengths uniform, (3) minimizes edge-crossings, and (4) keeps nodes from coming too close to edges. To draw graphs nicely, a simulated annealing (SA) method was used in [6] with the energy cost function designed to respect the above requirements. The implementation of their work allows the user to adjust the relative weights of criteria in the energy cost function among various parameters, though some of these criteria may be in conflict with each other. The SA method offers a more flexible way of drawing graphs than many other algorithms. Our simulated annealing graph drawing method is basically based on the strategy reported in [6]. The interested reader is also referred to [7,8] for earlier work of applying the SA method to graph drawing.

As there are many intuitive factors that affect the mental map, Bridgeman and Tamassia [5] formally defined those intuitive factors for *orthogonal drawings* and presented some statistical results to evaluate their effects on human perception of similarity. They used many metrics to measure the similarity between different drawings, and then asked respondents to evaluate the best measures that reflect their perception of similarity among a set of orthogonal drawings. Finally, they used a statistical method to find the relationship between these measurements and users' feelings, and suggested the following for measuring the degree of preservation of the mental map: *ranking*, *orthogonal ordering*, *λ-matrix*, *unweighted nearest neighbor within*, *average distance*, *weighted nearest neighbor between*, *shape*, and *relative distance*. Previous studies ([32,33,35]), on the other hand, often apply student-based surveys to evaluating the performance of mental-map-preserving graph drawing algorithms.

In this paper, we propose a mental-map-preserving graph drawing algorithm for straight-line drawings of general undirected graphs, based upon the simulated annealing graph drawing approach of [6] with the energy cost function incorporating

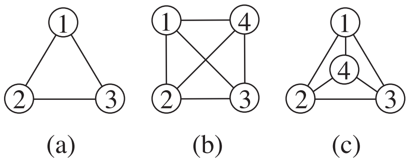


Fig. 1. An example of the mental map problem.

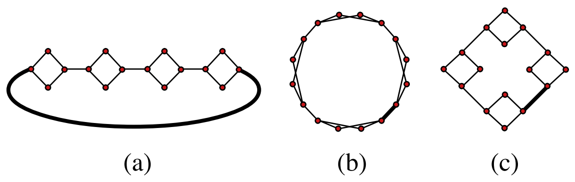


Fig. 2. An example of the mental map problem in circular drawing [22].

six criteria of [5] to reflect the user's mental map. As pointed out in [24], student-based surveys may not always be appropriate as far as evaluating the quality of a mental-map preserving graph drawing algorithm is concerned. Hence, in our study we consider not only experimental evaluation but also student-based questionnaire analysis, in order to better justify the claimed performance of our design. The experimental results show that to a certain extent, our approach is capable of preserving the drawing contour of a graph and the relative positions of nodes when applying a vertex/edge insertion/deletion operation to a graph. Similar to [6], our approach is very flexible in the sense that it allows the user to adjust the relative weights of criteria among various parameters. We also see from the experimental evaluation that there exists a trade-off between drawing graphs nicely and preserving the mental map. As for the questionnaire analysis, a high percentage of respondents choose the modified drawing produced by our approach as the best mental-map-preserving modified drawing in their minds.

The main contributions of this paper are summarized as follows:

1. Most of the previous work of preserving the mental map focused on how to preserve the orthogonal ordering (see, e.g., [25]) or add constraints to the input graph (see, e.g., [2,19]). To our knowledge, there have not been graph drawing algorithms aiming at preserving the mental map on straight-line drawings of general undirected graphs while using more sophisticated criteria simultaneously.
2. Our approach allows the user to adjust the relative weights of various criteria, making our drawing algorithm very flexible.
3. By taking samples among a group of students with basic knowledge of graphs and graph drawing, Bridgeman and Tamassia [5] suggested a number of good criteria suitable for capturing the notion of the mental map. Our graph drawing approach incorporates the criteria of [5] into the energy cost function of a simulated annealing graph drawing algorithm to preserve the drawing contour of a graph and the relative positions of nodes when modifying the graph.
4. Our approach obtains the location of each node in each iteration, resulting in a smooth process from the initial drawing to the final drawing–a desirable property in information visualization.

The rest of this paper is organized as follows. In Section 2, we introduce the simulated annealing graph drawing method, upon which our graph drawing algorithm is based. Section 3 involves the design of an energy cost function to preserve the mental map. Section 4 shows the experimental results of our approach. Section 5 gives a questionnaire analysis of our approach. A conclusion is given in Section 6.

2. Preliminaries

As mentioned in the previous section, the four requirements of a "*nice drawing*" include distributing nodes evenly, making edge-lengths uniform, minimizing edge-crossings, and keeping nodes from being too close to edges. In this section, we first introduce the general framework of simulated annealing, and then show how to draw graphs nicely using the simulated annealing graph drawing algorithm proposed by Davidson and Harel (DH for short) [6].

2.1. Introduction to simulated annealing

Simulated annealing (SA) [23] is a generalization of the Monte Carlo method for examining the equations of states and frozen states of *n*-body systems. SA is flexible and has successfully been applied to solving various combinational optimization problems, including solving the maximum clique problem [18], determining the thickness of a graph [28], etc. It allows "uphill" moves–moves that spoil, rather than improve, the temporary solution. This unique feature allows SA to escape from a local minimal solution, though it is not guaranteed that a global minimum can be reached eventually.

The basic steps of SA are given as follows. The first step of SA is to generate an initial configuration (either randomly or heuristically constructed) and initialize the so-called temperature parameter T. Then the following is repeated until the termination condition is satisfied. A new solution from the neighborhood of the current solution is selected randomly. The *energy cost* of the new solution is calculated over the objective function and T is decremented. Let ϕ and ϕ' denote the current and new energy costs, respectively. If $\phi' < \phi$, the new solution will be accepted. If $\phi' > \phi$, this solution will be accepted with a probability from a Boltzmann distribution function $e^{-(\phi'-\phi)/kT}$, where k is the Boltzmann constant. We assume $k = 1$. This kind of "uphill" moves will be fewer as SA proceeds, because T decreases after each iteration.

2.2. The graph drawing algorithm using simulated annealing

The input of the DH algorithm is a graph $G = (V,E)$, where V is the set of nodes and E is the set of edges. Each node in V has a unique index. The key design in SA is the *energy cost function*, on which the efficiency of the algorithm highly depends. A good energy cost function should reflect the properties of nice-looking drawings. The criteria used by the DH algorithm in the design of the energy cost function are given as follows, in which $\lambda_1,\ldots,\lambda_5$ are normalizing factors that define the relative importance of each criterion compared to the others.

1. *Node distribution*: $\forall(i,j) \in V \times V$, $a_{i,j} = \lambda_1/d_{i,j}^2$ is added so that nodes are spread evenly, where $d_{i,j}$ is the distance between nodes i and j.
2. *Borderlines*: $\forall i \in V$, $m_i = \lambda_2(1/r_i^2 + 1/l_i^2 + 1/t_i^2 + 1/b_i^2)$ is added so that nodes are prevented from being too close to the borderlines, where r_i, l_i, t_i, b_i stand for the distances between node i and the right, left, top and bottom sides of the drawing frame, respectively.
3. *Edge lengths*: The DH algorithm uses $c_k = \lambda_3 d_k^2$, where d_k is the length of edge k. Instead, our algorithm uses $\sigma = \lambda_3 s^2$, where s is the standard deviation of edge lengths, because the utilization of c_k tends to produce drawings with shorter edge lengths which may not be desirable in some cases of preserving the mental map.
4. *Edge crossings*: $\forall(k,l) \in E \times E$, we simply add the term $e_{k,l}$ which equals $\lambda_4 f_k f_l$ if edges k and l cross; $e_{k,l} = 0$ otherwise, where f_k (resp., f_l) is the weight attributed to edge k (resp., l), representing the importance of edge k (resp., l) not intersecting other edges. Note that $\lambda_4 = \lambda_5/g_{min}^2$ where g_{min} is the given minimum distance between nodes and edges.
5. *Node-edge distances*: For node i and edge k with distance $g_{i,k}$, the term $h_{i,k} = \lambda_5/g_{i,k}^2$ is added.

In light of the above, the energy cost ϕ of a drawing of graph $G = (V,E)$ is computed as follows:

$$\phi = \sum_{(i,j)\in V\times V, i \neq j} a_{i,j} + \sum_{i\in V} m_i + \sigma + \sum_{(k,l)\in E\times E, k \neq l, k\notin N(l)} e_{k,l} + \sum_{(i,k)\in V\times E, k\notin N(i)} h_{i,k},$$

where we slightly abuse the notation $N(\cdot)$ with $N(l)$ denoting the set of neighboring edges of edge l, and $N(i)$ representing the set of the edges incident to node i. In fact, only a node moves from the original configuration to the new configuration in each step of the algorithm. For notational convenience, the node in the original (resp., new) configuration is called p (resp., p'). If the calculation of a component takes p (resp., p') into account, it implies that the component belongs to the energy cost function of the original (resp., new) configuration. Therefore, the difference of the energy costs between the original and the new configurations can be computed more efficiently as follows: $\phi' - \phi = \sum_{(p,j)\in N(p)}(a_{p',j} - a_{p,j}) + (m_{p'} - m_p) + (\sigma' - \sigma) + (\sum_{(k,l)\in N(p')\times E, k \neq l} e_{k,l} - \sum_{(k,l)\in N(p)\times E, k \neq l} e_{k,l}) + \sum_{k\in E, k\notin N(p)}(h_{p',k} - h_{p,k})$, where σ' is the variance of the edge lengths in the new configuration.

3. Our mental-map-preserving graph drawing algorithm

The work of Bridgeman and Tamassia [5] can be regarded as the most complete work on mental map metrics. They presented the results of a user study based on several similarity measures between orthogonal drawings, which reflect the user's mental map according to the user's direct responses. As a result, our approach to the mental map preservation problem is to incorporate some of the best measures used in [5] for straight-line drawings into the DH algorithm. In what follows, we introduce the main components of our algorithm.

3.1. Configuration

Given a graph $G = (V,E)$, a *configuration* specifies the placement of nodes in V on a plane, assuming that edges are drawn as straight-line segments, and G is drawn within a given rectangular frame. If there is no *a priori* information about the nodes, we place the nodes randomly within the drawing frame. The *neighborhood* of a configuration C is a configuration that is obtained from C by changing the location of a certain node. In each iteration, we choose a node arbitrarily and then assign it a new random location within a circle of decreasing radius around its original location. The setting of a decreasing radius may accelerate the convergence of the algorithm [6].

3.2. The mental map cost function

In order to preserve the mental map, we define the *mental map cost* M between the original drawing D and the modified drawing D' to reflect the similarity of drawings based on certain "components" used in the experiments reported in [5]. Note that the value of each component in M is nonnegative, and hence $M = 0$ implies that the two drawings are viewed as the same drawings. A larger magnitude of M means a higher degree of difference between the two drawings.

Note that all the measures proposed in [5] except the "*Shape*" measure are defined in terms of *point sets*, which are derived from the edges and nodes of the graph instead of the edges and nodes themselves. Each point in the original drawing has a corresponding point in the modified drawing. Although there are a variety of ways to select the point sets as indicated in [5,27], in this paper we choose the point set including all the nodes of the graph, because there is at most a node moving in each trial of our SA algorithm–it suffices to compute only the values related to the moved node.

We adopt the notations originally used in [5] in our subsequent discussion. P and P' always refer to the point sets of drawings D and D', respectively. Since only one point moves in each trial of our SA algorithm, we let p denote the moved point, and $p' \in P'$ is the corresponding point of p (and vice versa). The number of nodes in V is n. Recall that $d_{p,q}$ denotes the Euclidean distance between points p and q.

In the following, we define components used in M, in which $\eta_1, \ldots, \eta_6$ are the normalizing factors that define the relative importance of a criterion compared to the others.

1. *Ranking*: Considering that the relative horizontal and vertical positions of a node should not vary too much, we add to M the component

$$rank(P,P') = \frac{\eta_1}{UB}\min\{|right(p) - right(p')| + |above(p) - above(p')|, UB\},$$

where $UB = 1.5(|P| - 1)$; $right(p)$ and $above(p)$ are the numbers of points to the right and above of p, respectively. Note that the upper bound UB is set to $1.5(|P| - 1)$ rather than $2(|P| - 1)$ because the probability under which the actual maximum value happens is very small.

2. *λ-matrix*: Two point sets P and P' have the same order type if, for every triple of points (p,q,r), they are oriented counterclockwise if and only if (p',q',r') are also oriented counterclockwise. The λ-matrix is designed to preserve an order type of points. Here we use a simplified version, which calculates the number of nodes to the left of two designated nodes. Let $\lambda(p,q)$ be the number of points in P to the left of the directed line from p to q. We add to M the term

$$lambda(P,P') = \frac{\eta_2}{UB}\sum_{q \in P, q \neq p} |\lambda(p,q) - \lambda(p',q)|,$$

where $UB = n\left\lfloor \frac{(n-1)^2}{2} \right\rfloor$.

3. *Unweighted nearest neighbor within*: This component captures the intuitive idea that a node's nearest neighbor should remain its closest neighbor after the graph is redrawn. Let $nn(p)$ be the nearest neighbor of p in P and $nn(p)'$ be the corresponding node in P' to $nn(p)$. The component is defined as:

$$nnw(P,P') = \frac{\eta_3}{UB}\sum_{p \in P}\min\{|nearer(p)|, 1\},$$

where $UB = |P|; nearer(p) = \{q | d_{p',q} < d_{p',nn(p)'}, q \in P, q \neq p, q \neq nn(p)\}$.

4. *Weighted nearest neighbor between*: This concept is based on the assumption that a node should not move too far from the original position, and hence its new position should be as close to its original position as possible. As the new drawing is presented, the user ought to be able to find a certain node which is recorded in his/her mental map. The component is defined as:

$$nnb(P,P') = \frac{\eta_4}{UB}\sum_{p \in P}|nearer(p)|,$$

where $UB = |P|(|P| - 1); nearer(p) = \{q | d_{p,q} < d_{p,p'}, q \in P, q \neq p\}$.

5. *Shape*: This component records the edge orientation of the graph. Note that the original measure in [5] was designed for orthogonal drawings. To cope with straight-line drawings, the measure is modified as follows:

$$shape = \frac{\eta_5}{UB}\sum_{e \in E} edits(e,e'),$$

where $UB = |E|$ and

$$edits(e,e') = \begin{cases} 1, & \text{if the included angle between } e \text{ and } e' \text{ is small,} \\ 0, & \text{otherwise,} \end{cases}$$

in which e' is the corresponding edge of e in D'. Note that whether the included angle between two edges is small is determined by users.

6. *Average relative distance*: The *average relative distance* is the average of the distances from p to the remaining nodes. Therefore, we add the term

$$rdist(P,P') = \frac{\eta_6}{n(n-1)}\sum_{p,q \in P}|d_{p,q} - d_{p',q}|.$$

Although eight best similarity measures for orthogonal drawings were suggested in [5], only six of them are included in our mental map cost function. The two measures designed for preserving the orthogonal ordering (called *Orthogonal Ordering* in [5]) of nodes and minimizing the movement distance of the selected node (called *Average Distance* in [5]) are excluded in our work for the reasons given below. First, if some new nodes and edges are arbitrarily added to a drawing causing new edge crossings, then those new edge crossings cannot be eliminated by an algorithm which preserves the orthogonal ordering of nodes. Also, minimizing the number of edge crossings is usually regarded as the most important aesthetic criterion in graph drawing [29]. Therefore, we exclude the measure of preserving the orthogonal ordering of nodes. As our experimental results indicate, a high degree of orthogonal order of nodes is still preserved without taking this measure into account. On the other hand, the measure of minimizing the movement distance of the selected node is excluded because it has been implied by the selection of the neighborhood of a configuration in the SA algorithm. In addition, it can also be viewed as being achieved in part by the *edge length* component in the energy cost function. Finally, the exclusion of the two measures results in a more efficient SA algorithm, as expected.

3.3. Our algorithm

Our algorithm, described in Algorithm 1, is almost the same as the DH algorithm except the if condition at step 2(b) is replaced by the conjunction of the original condition and $M' < \epsilon_M$, where M' is the value of the mental map function at the new drawing D' and ϵ_M is the maximal tolerance of the mental map cost that accepts the new drawing D'.

```
Algorithm 1 SA_MENTAL_MAP
1: denote the input drawing as $D$ and set an initial temperature $T_0$;
2: for $i$-th stage, denote the temperature as $T_i$, and repeat the following for $30|V|$ trials:
    (a) randomly select a point $p \in P$ (note that $P$ is the point set of $D$); select a new position $p'$ in a decreasing-radius
        circle around $p$; let $D'$ be a new drawing which is almost the same as $D$ except $p$ is replaced by $p'$;
    (b) let $\phi$ and $\phi'$ be the values of the energy cost function at $D$ and $D'$ respectively; let $M'$ be the value of the mental
        map function at $D'$; if ($\phi' - \phi < 0$ or $random < e^{-(\phi'-\phi)/T_i}$) and ($M' < \epsilon_M$), then set $D \leftarrow D'$;
3: $T_{i+1} = \gamma T_i$;
4: if the termination condition is satisfied, stop; otherwise go to step 2.
```

The energy cost and the mental map cost are considered independently in Algorithm 1, because they serve for different purposes. The former is used for drawing graph nicely, whereas the latter is used for preserving the mental map. Each component in the mental map cost can be normalized with the upper bound of value 1 except for the '*Average relative distance*' component (which is of small value while preserving the mental map), and hence, it is easy to control the relative weights of these components to preserve the mental map. However, the values of the components in the energy cost are so diverse that it is difficult to adjust their weights to generate a nice drawing when the algorithm converges. Consolidating the two costs with different scales into one does not help, since it becomes more difficult to adjust the weights to reach convergence, and more importantly, the goals associated with the two costs are likely to be in conflict with each other. As a result, the energy cost and the mental map cost are considered independently in Algorithm 1.

Also note in Algorithm 1 that the SA scheme is only applied to the energy cost, because it would be difficult to achieve convergence if applying to the two costs simultaneously. In addition, we believe that in most cases a user prefers a nice drawing with a reasonable preservation of the mental map than a bad drawing with a well-preserved mental map. Consequently we apply the SA scheme only to the energy cost.

The cooling schedule used in Algorithm 1 follows the basic geometric schedule described in [23].

1. *Initial temperature*: We set the initial temperature high enough in order for any configuration to work as the starting point of the algorithm. A lower initial temperature can be chosen in the beginning, if the initial configuration is known to bear some resemblance to the desired final result.
2. *Temperature reduction*: The *temperature reduction* stage determines when to change the temperature and how to change it. This stage affects the number of moves at each temperature and the range of each move. Typical applications set the number of trials at each temperature to be polynomial in the size of the input. In our setting, the number of moves at each temperature is set to $30|V|$, suggested by the DH algorithm. The cooling schedule rule is geometric. If T_i denotes the temperature at the i-th stage, then the temperature at the next stage is $T_{i+1} = \gamma T_i$, where γ is a real number between 0.6 and 0.95. Similar to the DH algorithm, we let $\gamma = 0.75$. The range of each move decreases when the temperature reduces. If the range of movement at the i-th stage is R, then the range of movement serving for the next temperature is γR, which accelerates the convergence of the algorithm.
3. *Termination condition*: This is to decide when to stop the algorithm. Commonly used termination conditions include: stopping the algorithm when the solution is not modified after several consecutive stages, or fixing the number of stages to be a constant. To carry out a fair comparison between our algorithm and the DH algorithm, the number of stages in our experiments is set to ten, which is identical to that used in [6]. Experimental results have shown that, in many cases, substantial changes only happen in the early stages; subsequent stages yield only marginal improvement.

Now we turn our attention to the time complexity of our algorithm. As our algorithm is built upon the DH algorithm, it suffices to consider the time complexity of the DH algorithm. As indicated in [6], the DH algorithm runs in $O(|V|^2|E|)$ time because the number of stages is a constant, the number of trials in each stage is $30|V|$, and every term in the energy cost function can be computed in $O(|V||E|)$ time. Also, it is obvious that every component of the mental map cost can be computed in time no more than $O(|V||E|)$. Hence, Algorithm 1 also runs in $O(|V|^2|E|)$ time.

4. Experimental results

This section shows four experimental results for evaluating the performance of our mental-map-preserving graph drawing algorithm. In each experiment, the initial drawing of the input graph is a random drawing, and then a nice drawing is produced by the DH algorithm. Subsequently, like the experiment designed in [5], the graph is modified by adding two nodes with two and four incident edges, respectively. The locations of the two nodes are decided randomly, and the other end points of the six edges are connected to arbitrary nodes in the original graph. Finally, we compare the drawings of the modified graph produced by our algorithm and the DH algorithm.

Our prototype system runs on an Intel Core 2 Duo CPU E8400 @ 3.00 GHz PC with 3.00 GB memory. The statistics on the running time of the DH algorithm and our algorithm with respect to four examples are given in Table 1. Our algorithm runs a bit slower than the DH algorithm since more criteria are taken into account. Like the conclusion given in [6] regarding the DH algorithm, our algorithm, from a practical viewpoint, is only suitable for drawing dynamic graphs of small-size.

In order to see how the running time of our algorithm increases when the size of the input graph grows, we consider several graphs taken from the well-known graph data set–*Rome Graphs*,[1] which are originated from practical applications ranging from data flow diagrams to Petri nets. The largest graph in the Rome Graph data set has 110 nodes. We conduct experiments on the first 100 graphs in the Rome Graph set, and record the running time of redrawing the modified graph of each of the 100 graphs. To this end, we add two nodes, one with two edges and the other with four edges, to the nice drawing of each graph. The average running time versus the number of nodes is plotted in Fig. 3. It turns out that it takes about 25 s to redraw a 100-node graph, which is somewhat unacceptable for applications that require real-time responses.

Consider the first simple experiment shown in Fig. 4. An initial random drawing is given in Fig. 4(a). After applying the DH algorithm under different settings of parameter λ_4, three possible nice-looking drawings are displayed in Fig. 4(b), in which (i)–(iii) illustrate how less penalties on edge crossings affect the outcomes of the algorithm. Now we arbitrarily add two new nodes 7 and 8 to each of the three drawings, where node 7 is connected to nodes 4 and 5, and node 8 is connected to nodes 1, 3, 4, and 5 (see Fig. 4(c)). The modified graphs are redrawn by our algorithm, as shown in Fig. 4(d). For the sake of comparison, consider Fig. 4(e) and (f). Fig. 4(e) shows the three drawings obtained by applying the DH algorithm to drawings in (i)–(iii) in Fig. 4(c) directly. On the other hand, Fig. 4(f) displays the outcomes of applying the DH algorithm to random drawings of graphs associated with (i)–(iii) in Fig. 4(c) under the same setting of parameters.

In view of Fig. 4(c) and (d), it is obvious that our algorithm is capable of generating nice drawings. It is also not hard to see from Fig. 4(b) and (d) that the contours of drawings and the relative positions of nodes are very similar in the two sets of drawings, suggesting that our algorithm preserves a high degree of the mental map. In contrast, the drawings in Fig. 4(e) and (f) produced by the DH algorithm share little similarity with those in Fig. 4(b).

If we remove the additional nodes 7 and 8 and their adjacent edges from the drawings in Fig. 4(c)–(f), it becomes clearer why the drawings in Fig. 4(d) preserve better mental maps. See Fig. 5. Since the mental map criteria and the aesthetic criteria are often in conflict with one another, there is a trade-off between preserving a high degree of the mental map and achieving some aesthetic criteria such as minimizing the number of edge crossings. Fortunately, our algorithm allows the user to adjust the weights between different criteria, in order to fit his/her drawing requirements. For example, node 5 in Fig. 5(b) (i) (which is produced by our algorithm) does not preserve its original relative position (i.e., not satisfiable as far as preserving the mental map is concerned). Consider Fig. 4(c)(i). To keep node 5 in the same relative position (by changing the settings of parameters), node 8 cannot move into the quadrangle bounded by nodes 1, 2, 3, and 6, which in turn, induces additional edge crossings, though node 8, in this case, keeps its original relative position intact.

In the second experiment, the initial graph Fig. 6(b) is a complete degree-four tree with 21 nodes and 20 edges. Taking the initial random drawing shown in Fig. 6(a) as the input, the DH algorithm outputs the drawing in Fig. 6(b). Fig. 6(c) shows the graph transformed by inserting nodes 22 and 23 (together with six edges) into the tree. Fig. 6(d) is the drawing produced by applying our algorithm to the drawing in Fig. 6(c). We can observe that there are no drastic differences between Fig. 6(b) and (d). Fig. 6(e) (resp., Fig. 6(f)) is the drawing produced by applying the DH algorithm to the drawing in Fig. 6(c) (resp., a random drawing of the modified graph). Although the drawing in Fig. 6(f) almost has no common points with the drawing in Fig. 6(b), there are only two edge crossings in the drawing in Fig. 6(f), which achieves the minimum number of edge crossings among all possible drawings of the modified graph.

Fig. 7(a)–(d) simply display the drawings after removing the additional nodes 22 and 23 from Fig. 6(c)–(f), respectively. By comparing Fig. 7(b) with Fig. 7(a), it can be observed that the drawing in Fig. 7(b) almost preserves the shape and structure of the drawing in Fig. 7(a) except that the orderings of node pairs (6, 7) and (15, 16) are different. As mentioned earlier, preserving the mental map and optimizing some aesthetic criteria may not be achieved simultaneously. The reason why the order-

[1] http://www.dia.uniroma3.it/ ~ gdt/.

Table 1
The statistics on the total running time of the DH and our algorithms.

Problem	$|V|$	$|E|$	DH	our
Fig. 4(c)	8	15	0.1453 s	0.1781 s
Fig. 6(c)	23	26	0.7206 s	0.9837 s
Fig. 8(c)	65	68	5.7500 s	8.5030 s
Fig. 10(c)	102	130	19.3353 s	28.1252 s

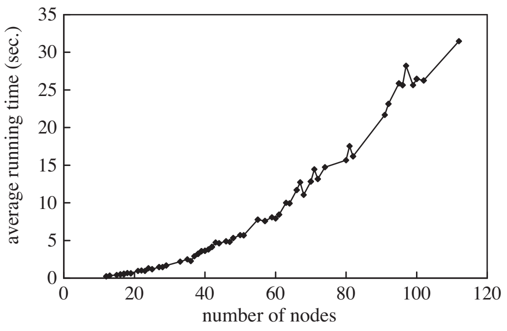


Fig. 3. Plot of average running time versus number of nodes.

ings of the two pairs of nodes are different is that the drawing in Fig. 7(b) corresponds to the drawing in Fig. 6(d) which induces fewer edge crossings than the case when the orderings are not modified. If we compare Fig. 7(c) with Fig. 7(a) assuming node 1 to be the root of the tree, the subtrees rooted at nodes 2, 3, and 4 look different although the counterclockwise ordering of the children of node 1 is preserved. By comparing Fig. 7(d) with Fig. 7(a), it is obvious that the mental map is not preserved.

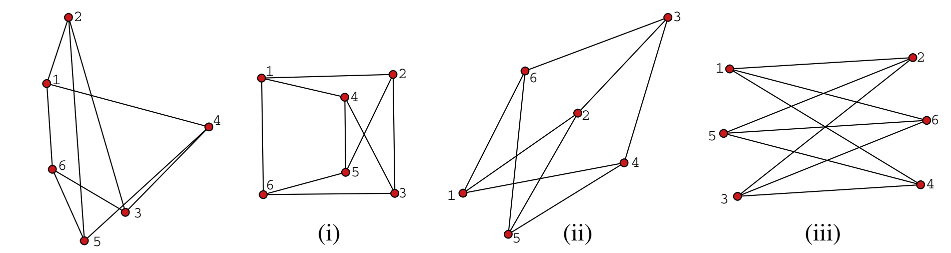
We also carry out an experiment on a complete binary tree with 63 nodes and 62 edges. Similar to the previous experiments, Fig. 8(a)–(f) show a random drawing of the initial graph, the nice drawing produced by the DH algorithm, the drawing of the modified graph by adding two nodes, the modified drawing produced by our algorithm, the modified drawing produced by the DH algorithm, and the drawing produced by applying the DH algorithm to a random drawing of the modified graph, respectively. Fig. 9(a)–(d) show the drawings after the two added nodes are removed from Fig. 8(c)–(f), respectively.

If we look at Fig. 8(b) and (d), there are only slight modifications between the two drawings, as only node pairs (7,15), (32,33), (36,37), (40,41), (58,59), and (60,61) are of different orderings. This is mainly caused by the fact that the measure of preserving the orthogonal ordering of nodes is excluded in our mental map cost function. If, on the other hand, such a measure is included, then additional edge crossings become inevitable, causing the output to be less attractive from the aesthetic viewpoint. In contrast, although the drawing in Fig. 8(e) has fewer edge crossings than the drawing produced by our algorithm, it differs a lot in comparison with the original drawing illustrated in Fig. 8(b), especially, in the left half of the drawing. Assuming node 1 to be the root of the tree, the subtrees rooted at nodes 8, 9, 14, 20, and 26 in the drawing of Fig. 8(a) are totally different from those in the drawing of Fig. 8(e). As for Fig. 8(f), the drawing has only one edge crossing, which achieves the minimum number of edge crossings among all the possible drawings of the modified graph. From Fig. 9, the advantage of our drawing algorithm becomes apparent when the primary concern is placed on the preservation of the mental map.

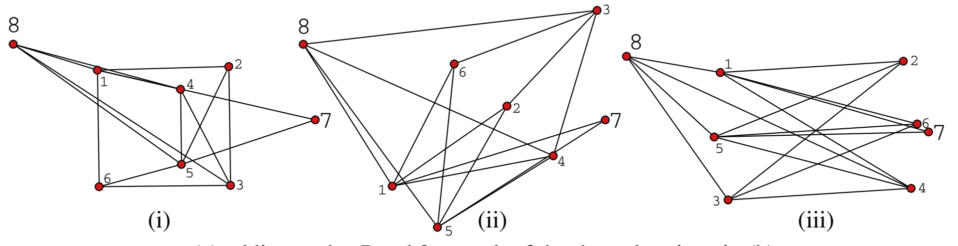
Finally, we present an experiment carried out on a graph with 100 nodes and 124 edges chosen from the Rome Graph data set. Similar to the previous experiments, Fig. 10(a)–(f) show a random drawing of the input graph, the nice drawing produced by the DH algorithm, the drawing of the modified graph by adding two nodes and six edges, the modified drawing produced by our algorithm, the modified drawing produced by the DH algorithm, and the drawing produced by applying the DH algorithm to a random drawing of the modified graph, respectively.

Although the drawings in Fig. 10 are more complicated, there is a common feature in (b)–(e) that most nodes are centralized in the bottom right part of the whole area. By comparing Fig. 10(b) with Fig. 10(d) and (e), it is clear that drawing (d) by our algorithm preserves the contour of the original drawing (b), but drawing (e) by the DH algorithm does not have this property.

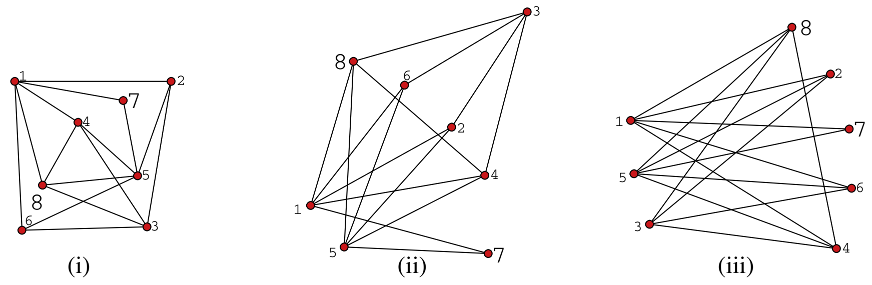
Figs. 11 and 12 show how our SA algorithm is capable of escaping from local minima. Fig. 11 gives the plot of the energy cost versus the number of trials in the process of applying the DH algorithm to Fig. 8(c) to yield Fig. 8(e). Recall from Algorithm 1 that the number of trials is equal to 30$|V|$ multiplied by the number of executed stages plus the number of trials executed at the current stage. Fig. 11 illustrates that some ‘uphill’ moves can successfully escape from local minima by using the DH algorithm. To see this, consider the energy costs in the window between the 16500-th trial and the 18000-th trial in



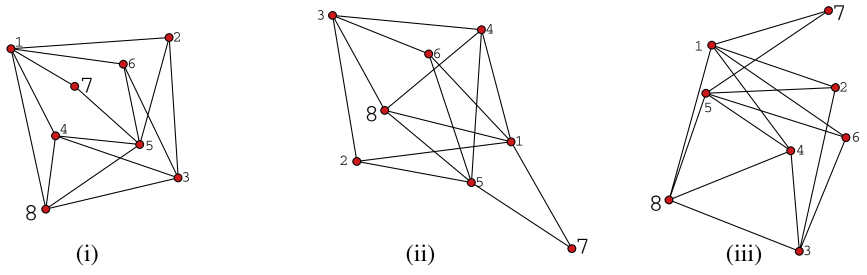
(a) the initial drawing. (b) applying the DH algorithm to (a) under different settings of parameters.



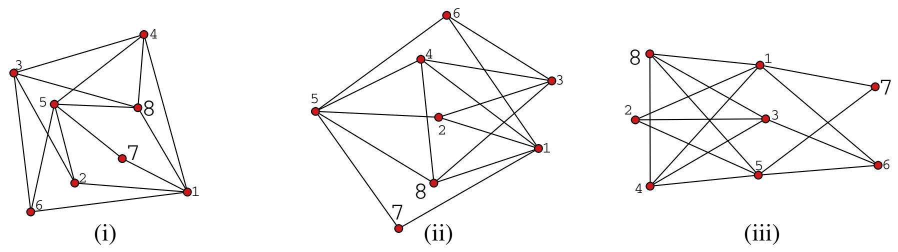
(c) adding nodes 7 and 8 to each of the three drawings in (b).



(d) applying our algorithm to each of the three drawings in (c).



(e) applying the DH algorithm to each of the three drawings in (c).



(f) applying the DH algorithm to a random drawing of the graph in (c).

Fig. 4. The experiment for an initial cubic graph with six nodes.

Fig. 11. The curve of the energy cost between the 16500-th trial and the 17000-th trial (‘plain’-moving) is almost flat. However, ‘downhill’-moving followed by ‘uphill’-moving near the 17000-th trial yields a better solution, which illustrates the advantage of the SA method.

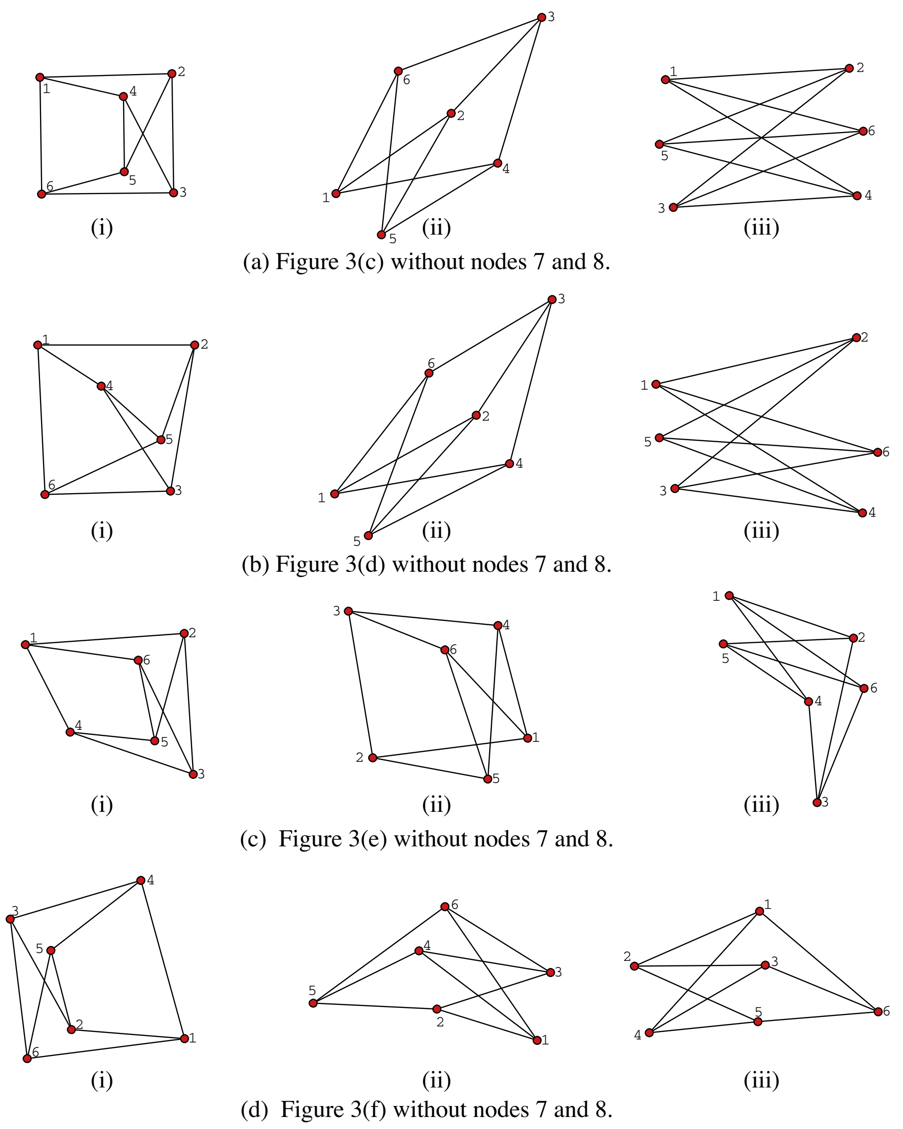
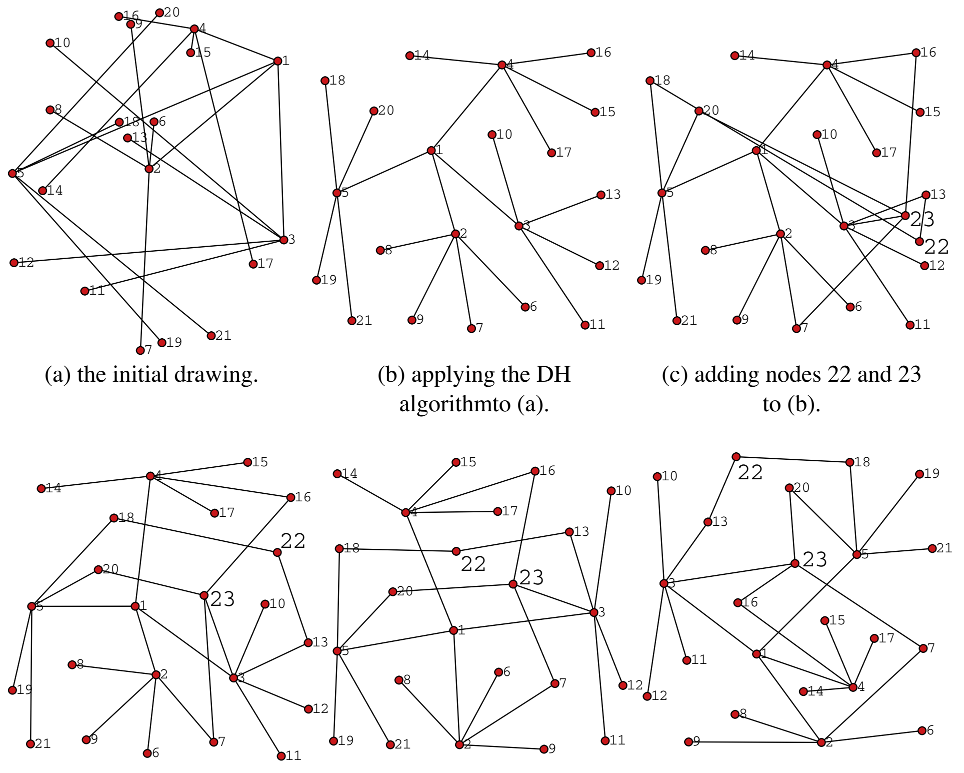

(i)
(ii)
(iii)
(a) Figure 3(c) without nodes 7 and 8.
(i)
(ii)
(iii)
(b) Figure 3(d) without nodes 7 and 8.
(i)
(ii)
(iii)
(c) Figure 3(e) without nodes 7 and 8.
(i)
(ii)
(iii)
(d) Figure 3(f) without nodes 7 and 8.

Fig. 5. Removing nodes 7 and 8 from Fig. 4 (c)(d) and (f).

Fig. 12 illustrates a "cost versus trial number" plot in the process of yielding Fig. 8(d) from Fig. 8(c) using our algorithm. Fig. 12(a) and (b) give the plots of the energy cost ϕ and the mental map cost M, respectively, against the number of trials. Fig. 12(c) and (d) give detailed views of the dotted-line boxes in (a) and (b), respectively. Note that the tolerance ϵ_M of M is set to 0.3 in this experiment, causing the configuration with the value of M larger than 0.3 to be rejected in Fig. 12(b) and (d), which in turn yields a considerable amount of 'plain-moving' in Fig. 12(a). From Fig. 12(a) and (c), we observe that our algorithm, an SA-based method, also has the ability to escape from local minima in many cases. It should be noted that the maximum energy cost in Fig. 11 (under the DH algorithm) is larger than that in Fig. 12(a) (under our algorithm) by approximately 4 billion. Such a disparity is due to the fact that the weight of the measure of penalizing the number of edge crossings (one of the most important measures in producing a nice drawing) is set to a huge value. In order to preserve the mental map, our algorithm produces the drawing in Fig. 8(d) with three more edge crossings than the drawing in Fig. 8(e), causing the extra 4 billion in the corresponding energy cost.

As our algorithm is based on the stochastic simulated annealing (SSA) strategy, it is not surprising that our algorithm is somewhat slow when applying to large graphs. A possible way to improve the performance of our algorithm is to consider using the *deterministic simulated annealing* (DSA) technique [11] instead, which runs faster in many real-world problems of large scale without sacrificing the accuracy of the solution. The basic idea of DSA is to use a hyperbolic tangent function to transform digital values of the energy cost in SSA (see Fig. 11) into analog values. Designing a DSA-based mental-map-pre-


(a) the initial drawing. (b) applying the DH algorithmto (a). (c) adding nodes 22 and 23 to (b).

(d) applying our method to (c). (e) applying the DH algorithm to (c). (f) applying the DH algorithm to a random drawing of the graph in (c).

Fig. 6. The experiment for an initial complete degree-four tree with 21 nodes.

serving graph drawing algorithm remains an interesting future research. In fact, consolidating the five factors $\lambda_1 - \lambda_5$ (which are of different scales) into a single hyperbolic function in the DSA framework is without doubt a challenge.

Note that a good measure of similarity between two graphs in terms of rotation, scaling and shift invariants is the *Procrustes statistic* [4], which is defined as follows:

$$R^2 = 1 - (tr(X^TYY^TX)^{1/2})^2/(tr(X^TX) \cdot tr(Y^TY))$$

where $X, Y \in \mathbb{R}^{n\times 2}$ denote the current and the next positions of nodes, respectively. If the statistic is minimized, it implies to optimally dilate, scale, rotate, and reflect X to fit Y [4]. It can be shown that $0 \leqslant R^2 \leqslant 1$ and if $R^2 = 0$, X and Y can be perfectly matched; if $R^2 = 1$, they cannot be matched by any $P \in \mathbb{R}^{2\times 2}$ at all.

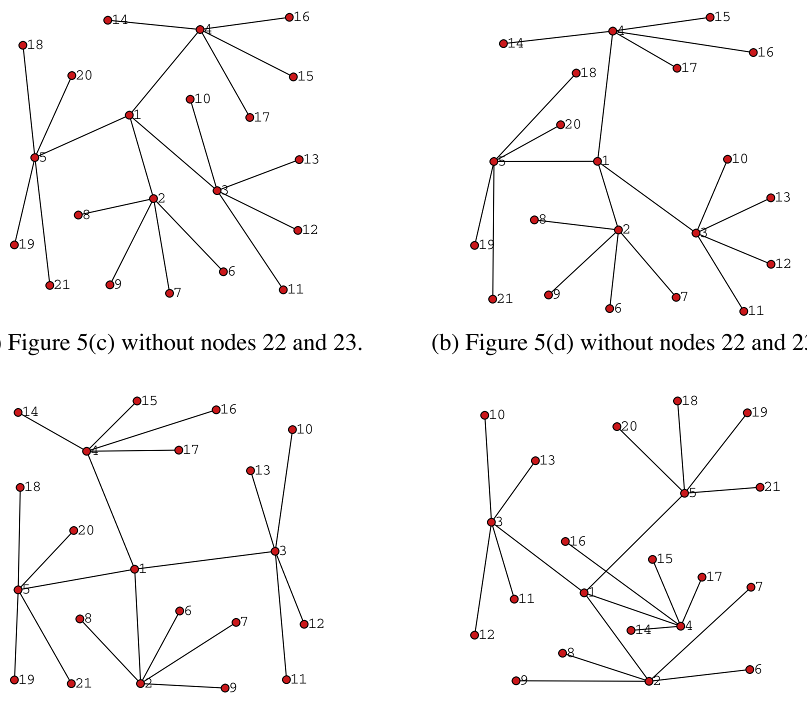
Although the Procrustes statistic cannot be used in our objective function, for comparison we measure the Procrustes statistics of the graphs used in the above experiments. More specifically, the Procrustes statistics between Fig. 4(c)(i) and (d)(i), between Fig. 6(c) and (d), between Fig. 8(c) and (d), and between Fig. 10(c) and (d) are 0.0866, 0.0692, 0.0288, and 0.0249, respectively. Since all the values are very small (less than 0.1), it is fair to draw a conclusion that the similarity by the Procrustes statistic is also preserved by our algorithm. As the output of our algorithm is produced after 10 stages, we plot the Procrustes statistics between the initial and final drawings of each stage in Fig. 13. As it turns out, all the Procrustes statistic values are small, suggesting that the similarity between each pair of adjacent stages is preserved.

5. Questionnaire analysis

In this section, we give a statistical analysis on two student-based surveys to evaluate the performance of our approach. Note that the experimental methodology used in the surveys is similar to those found in [5,32,33,35].

5.1. Experiment design

We develop an online system for our questionnaire survey. Our first survey includes ten questions. Each question consists of two phases:



(a) Figure 5(c) without nodes 22 and 23. (b) Figure 5(d) without nodes 22 and 23.

(c) Figure 5(e) without nodes 22 and 23. (d) Figure 5(f) without nodes 22 and 23.

Fig. 7. Drawings obtained from those in Fig. 6 by deleting nodes 22 and 23.

1. During the first phase, the system displays a nice drawing of the *original graph* (see Fig. 4(b)(i)) and a drawing of the *modified graph* (see Fig. 4(c)(i)) which is *slightly modified* from the original graph by arbitrarily adding or deleting at most three nodes and at most six edges. Also, a node is identified which appears at the same position in both the original and the modified graphs. In this phase, time is not recorded so the respondent can take whatever time he or she needs to understand the modification between the two graph drawings. The respondent is also allowed to take a break during this phase. The respondent clicks the 'ok' button to enter the second phase of answering a set of questions.
2. During the second phase, the system outputs eight drawings: the original drawing, the modified drawing, three *redrawn drawings*, as well as three *recovered drawings*. The three *redrawn drawings* are produced in the following ways: our algorithm taking the modified drawing as the input Fig. 4(d)(i), the DH algorithm taking the modified drawing as the input Fig. 4(e)(i), and the DH algorithm taking a random drawing of the modified graph as the input Fig. 4(f)(i). For convenience, the DH algorithm using the modified drawing (resp., a random drawing of the modified graph) as the input is denoted by DH_mdf (resp., DH_rand). Each redrawn drawing has a corresponding *recovered drawing*, in which the node or edge not in the original graph is deleted, and the edge not in the modified graph but in the original graph is added. For instance, Fig. 5(b)(i) is the recovered drawing of Fig. 4(d)(i). Note that the online system demonstrates the three redrawn drawings (and their corresponding recovered drawings) from left to right randomly, so the respondent does not know which redrawn drawing is produced by our algorithm.

Hypothesis 1. If the quality of the preservation of the mental map is measured in terms of the closeness of the drawing contour and the relative positions of nodes between drawings of the original and the modified graphs, our algorithm outperforms the DH algorithm.

Based on the above hypothesis, each respondent is asked to choose (click) the best mental-map-preserving redrawn drawing in his/her mind according to the following criteria:

- each redrawn drawing Fig. 14(c)–(e) is compared with the modified and the original drawings Fig. 14(a) and (b) to see if their contours look similar;
- the recovered drawing of each redrawn drawing Fig. 14(g)–(i) is compared with the original drawing Fig. 14(a) to see if the relative positions of nodes look similar (noticing that each node is labeled by a number, and the labeling is crucial).

Once the respondent makes the decision, the question is finished. Since time is recorded in this phase, respondents are asked to finish their questions as soon as possible.

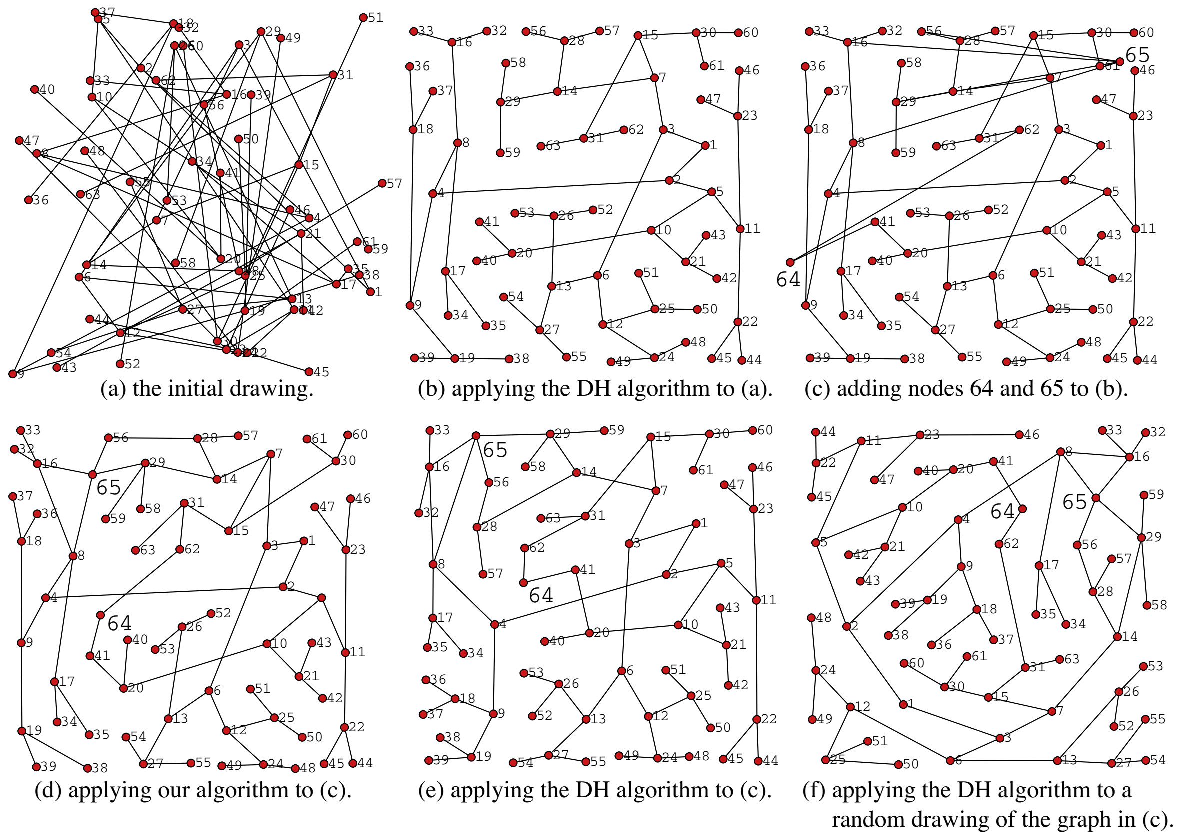

(a) the initial drawing. (b) applying the DH algorithm to (a). (c) adding nodes 64 and 65 to (b). (d) applying our algorithm to (c). (e) applying the DH algorithm to (c). (f) applying the DH algorithm to a random drawing of the graph in (c).

Fig. 8. The experiment for an initial complete binary tree with 63 nodes.

Aside from the ten questions mentioned above, the system contains three additional questions serving as a tutorial of our survey. We explain and operate the three tutorial questions in advance, and then ask the respondents to practice the three tutorial questions before answering the above ten questions.

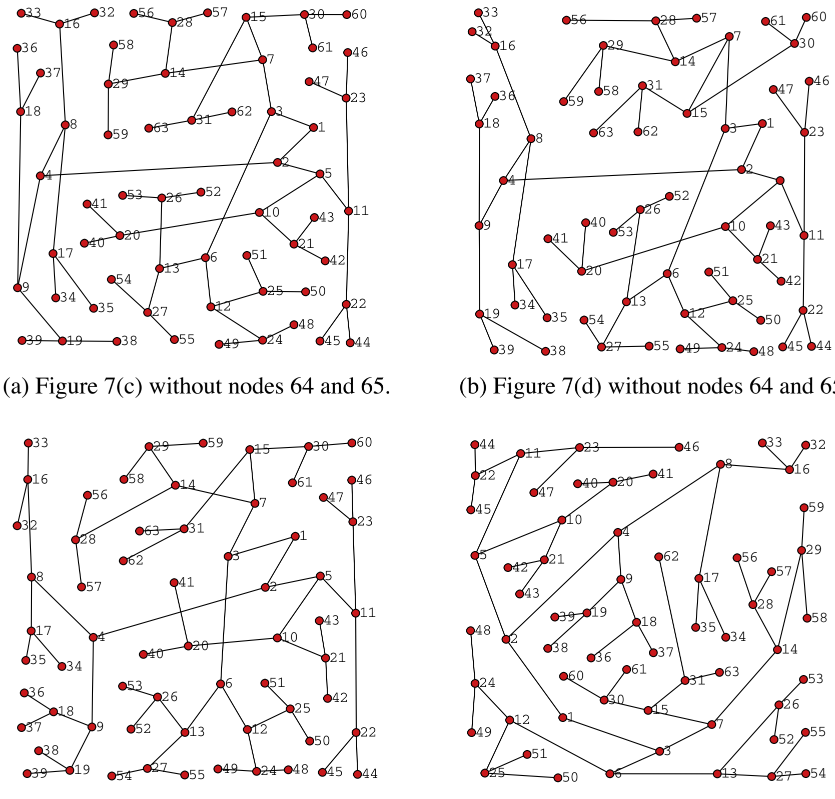
After finishing all the ten questions, respondents are asked to input their comments to this system. Note that this is not compulsive.

5.2. Source of questions

Before the survey began, five graduate students were recruited from the CS group in the EE Department of National Taiwan University in advance. We provided them with the five nice graph drawings shown in Figs. 4(b)(i)–(b)(iii), 6(b), and 8(b), and asked them to *slightly modify* those drawings, producing 25 slightly modified drawings in total. Note that not every modified drawing is suitable for the survey. For example, if the modified graph has too many symmetries, then the redrawn drawings produced by our and the DH algorithms would look very similar. Hence, 10 modified drawings from the 25 drawings were chosen appropriately as the questions in our questionnaire survey. They are denoted by Q1–Q10, in which Q1 and Q2, Q3 and Q4, Q5 and Q6, Q7 and Q8, Q9 and Q10 are modified from Figs. 4(b)(i)–(b)(iii), 6(b), and 8(b), respectively. In addition, three additional drawings (which are modified drawings of Figs. 4(b)(i), 6(b) and 8(b)) were chosen from the remaining 15 drawings to serve as the questions used in the tutorial of our survey.

5.3. Experimental process

We recruited 22 graduate students with basic knowledge of algorithms and data structures from either the CS group of the EE department or the CSIE Department in National Taiwan University. They were gathered in a computer classroom and each student was provided with a PC. To begin with, we explained to the students the purpose of our survey, as well as the concept of preserving the mental map in graph drawing. Then, we explained and operated the three tutorial questions on our online system, and then asked our students to practice those questions on the system. Once the tutorial session was completed, we asked those students to start the ten questions on the system formally. The entire procedure took about 50 min, including the tutorial session.



(a) Figure 7(c) without nodes 64 and 65.

(b) Figure 7(d) without nodes 64 and 65.

(c) Figure 7(e) without nodes 64 and 65.

(d) Figure 7(f) without nodes 64 and 65.

Fig. 9. Drawings obtained from those in Fig. 8 by deleting nodes 64 and 65.

5.4. Analysis

Since there are no outliers in the collected data, we do not exclude any data. The statistics of our questionnaire results are given in Table 2. Fig. 15 displays the stack charts, each of which illustrates the percentage of an approach over a given question. We observe from Table 2 that, on average, 82.27% of our respondents chose our redrawn drawing as the best mental-map-preserving redrawn drawing in their minds. The average time spent on a question is 29.89 s.

There are statistical differences in performance among the choices of the three approaches (percentage: $F = 156.88 > F(2,27) = 3.35$).[2] Post-hoc tests[3] between choice conditions result in the following conclusions:

- the percentage of choosing our redrawn drawing is greater than the percentage of choosing either DH_mdf or DH_rand redrawn drawing;
- it is not obvious which of the percentages of choosing DH_mdf and DH_rand redrawn drawings is larger.

From Table 2 and Fig. 15, it is clear that our approach does not behave well with respect to Q8. The reason behind this is that in Q8, the original graph is connected but the modified graph is disconnected. Our recovered drawing (in which some edges are added) looks unsatisfactory because it has the largest number of edge crossings among all recovered drawings, though our redrawn drawing performs well in terms of the preservation of the user's mental map. Thus, 36.36% of students chose the DH_mdf redrawn drawing, which looks nice although a bit weak in the preservation of the user's mental map. It is interesting to notice that although some students commented that they had a difficult time recognizing the user's mental map in Q9 and Q10, the percentages of choosing our approach for Q9 and Q10 remain high, according to our statistical analysis.

5.5. More complicated questions

In this subsection, we give a statistical analysis on another student-based survey, in which we ask respondents more complicated questions on three 100-node graphs taken from the Rome Graph data set. Unless stated otherwise, we continue

[2] Here we apply the standard two-tailed analysis, which is based on the critical values of the F distribution, under $\alpha = 0.05$.

[3] Like [33], the post hoc test used here is the Tukey test, under $\alpha = 0.05$.

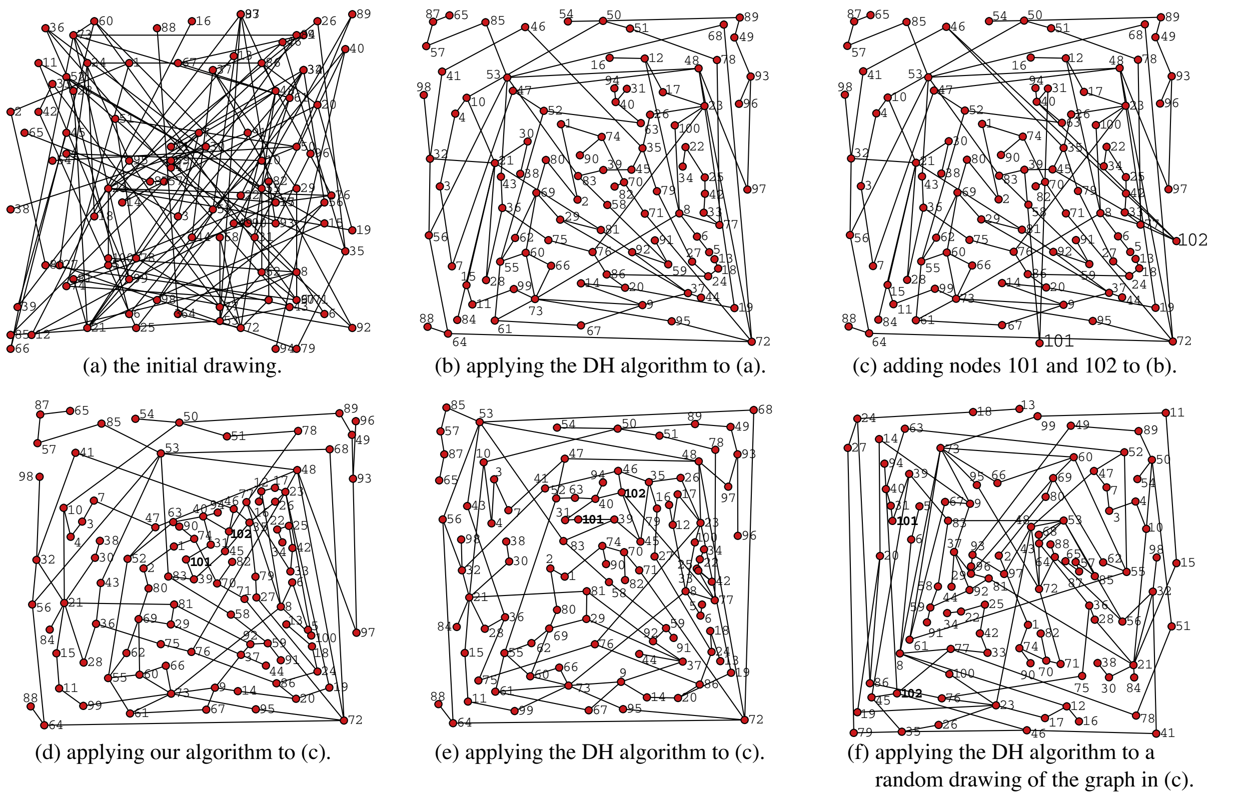

(a) the initial drawing.
(b) applying the DH algorithm to (a).
(c) adding nodes 101 and 102 to (b).
(d) applying our algorithm to (c).
(e) applying the DH algorithm to (c).
(f) applying the DH algorithm to a random drawing of the graph in (c).

Fig. 10. The experiment for an initial graph with 100 nodes.

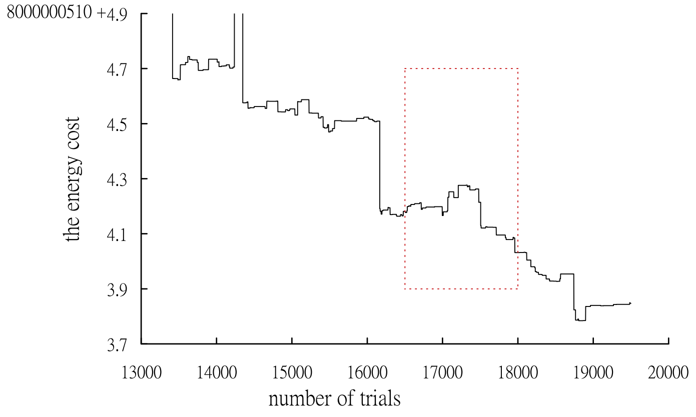


Fig. 11. The plot of the energy costs versus the number of trials by applying the DH algorithm to transforming Fig. 8(c) into (e).

applying the setting used in the previous subsections. In this survey, there were 14 respondents (graduate students), and similarly we gave the nice drawing produced by the DH algorithm, modified drawing, our drawing, DH_mdf drawing, and DH_rand drawing for each graph (drawings (b)–(f) in Fig. 10). For each of the three graphs extracted from the Rome Graph data set, we asked two more complicated questions based upon path tracing, i.e., there are six questions in total, where Q1' and Q2' denote the questions on the upper and lower drawing boundaries of the first graph, respectively; Q3' and Q4' (resp., Q5' and Q6') denote similar questions for the second (resp., third) graph. If the node labeled by *i* is denoted by v_i, we list the questions on the first graph (i.e., the graph in Fig. 10) as follows:

Q1': In Fig. 10, if we focus on the two paths $v_{65}v_{87}v_{57}v_{85}v_{53}$ and $v_{54}v_{50}v_{89}v_{49}v_{93}v_{97}v_{48}$ located near the top boundary of drawing (b), which of the three drawings (d)(e)(f) looks similar to (b)?

Q2': In Fig. 10, if we focus on the path $v_{88}v_{64}v_{72}$ located near the bottom boundary of drawing (b), which of the three drawings (d)(e)(f) looks similar to (b)?

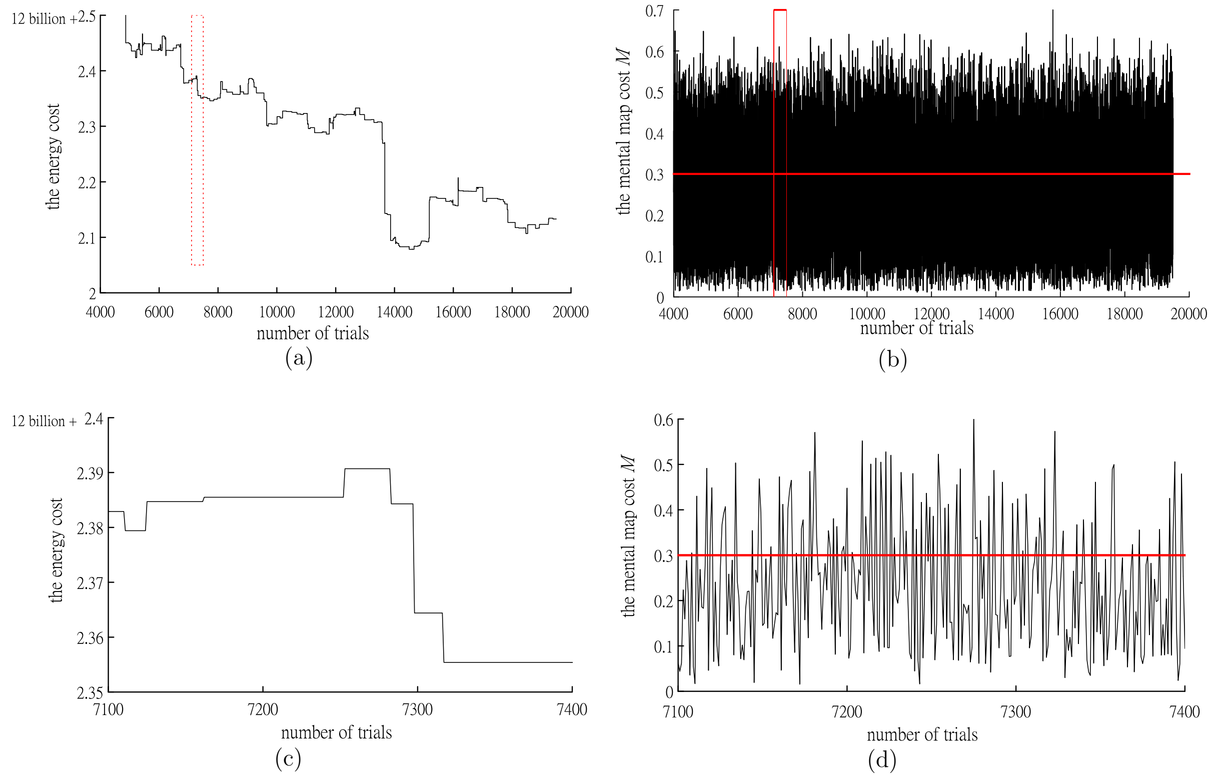


Fig. 12. The plots of the costs versus the number of trials by applying our algorithm to transforming Fig. 8(c) into Fig. 8(d).

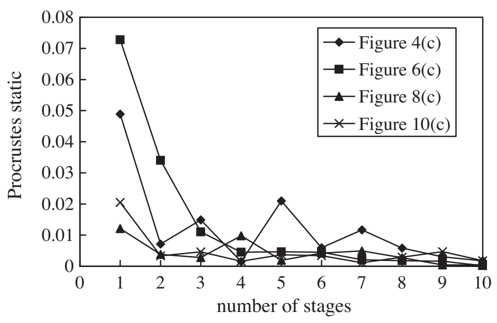


Fig. 13. Plot of Procrustes statistic versus number of stages.

Note that the above questions are designed according to Hypothesis 1.

The statistics of our questionnaire results are given in Table 3, and the respective stack charts are depicted in Fig. 16. From Table 3, on average, 86.90% of the respondents chose our redrawn drawing as the best mental-map-preserving redrawn drawing in their minds. The average time spent on a question is 17.24 s. We observe from Table 3 that when the percentage of choosing our drawing is lower (i.e., for Q1', Q2'), more time is spent because the DH_mdf drawings in those questions look also similar to the original drawing along those boundary paths (see Fig. 10(b) and (e)).

There are statistical differences in performance among the choices of the three approaches ($F = 68.18 > F(1,10) = 4.96$; $P = 0.000$). Post-hoc tests between choice conditions result in the following conclusions:

- the percentage of choosing our redrawn drawing is greater than the percentage of choosing either DH_mdf or DH_rand redrawn drawing;
- the percentage of choosing the DH_mdf redrawn drawing is greater than the percentage of choosing the DH_rand redrawn drawing.

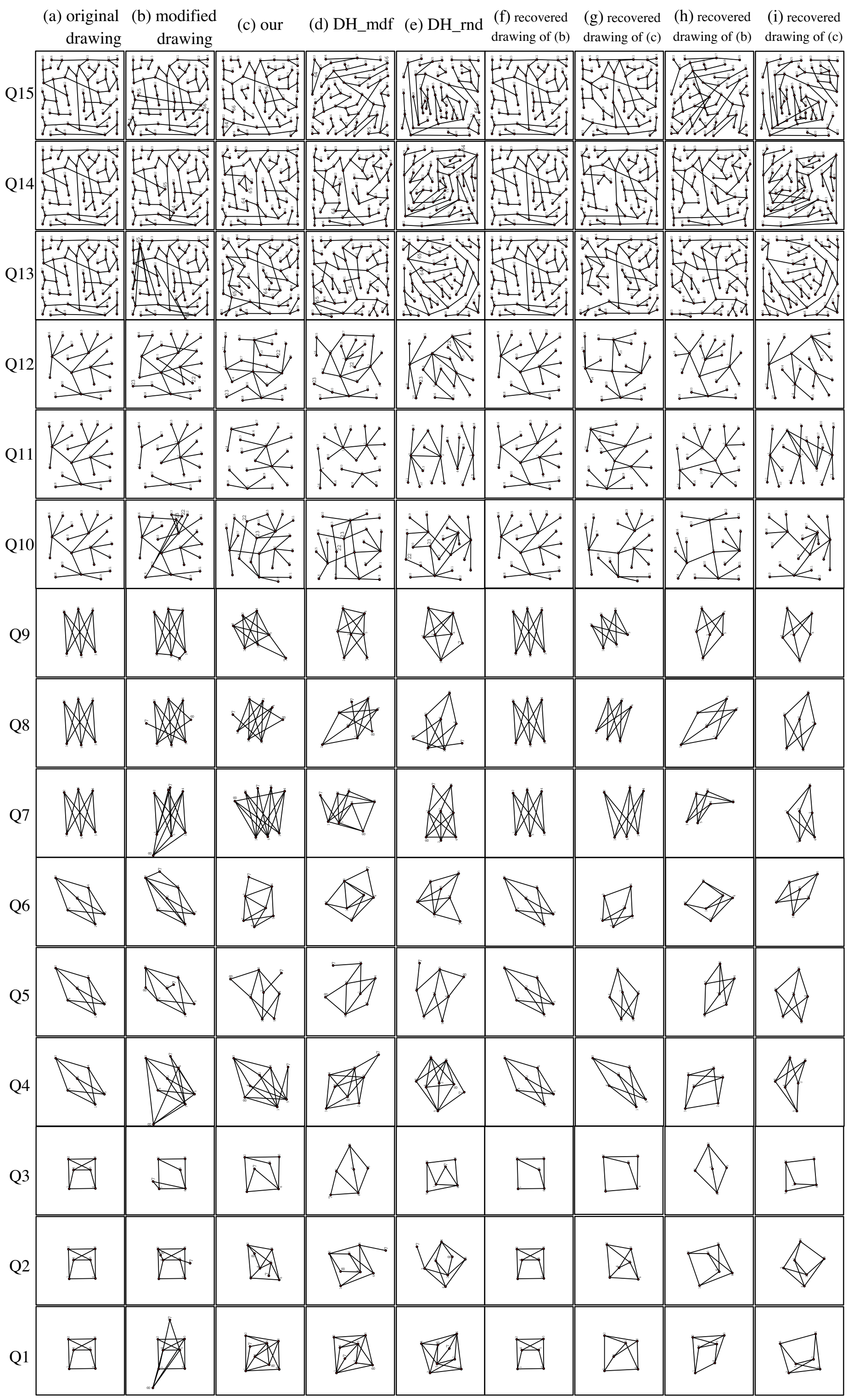

(a) original drawing
(b) modified drawing
(c) our
(d) DH_mdf
(e) DH_rnd
(f) recovered drawing of (b)
(g) recovered drawing of (c)
(h) recovered drawing of (b)
(i) recovered drawing of (c)
Q15
Q14
Q13
Q12
Q11
Q10
Q9
Q8
Q7
Q6
Q5
Q4
Q3
Q2
Q1

Fig. 14. The 15 questions used in our first student-based user study.

One may notice that in each question, no respondents chose the DH_rand drawings (see Table 3). This reflects that as a graph contains more nodes (like the 100-node graphs used in this experiment), respondents are more certain to reject the possibility of the similarity between an arbitrary drawing and the original drawing.

Table 2
Statistics of our first questionnaire results.

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	average
our	86.36%	77.27%	95.45%	72.73%	95.45%	90.91%	72.73%	54.55%	81.82%	95.45%	82.27%
DH_mdf	13.64%	18.18%	4.55%	4.55%	4.55%	9.09%	13.64%	36.36%	18.18%	0.00%	12.27%
DH_rand	0.00%	4.55%	0.00%	22.73%	0.00%	0.00%	13.64%	9.09%	0.00%	4.55%	5.45%
time (sec.)	34.02	28.13	23.96	28.20	14.16	29.30	44.11	39.68	35.45	21.86	29.89

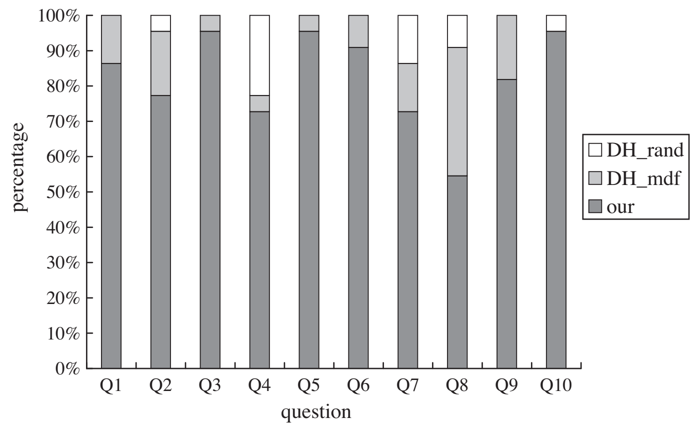


Fig. 15. The first student-based survey, in which the stack column chart illustrates the percentage of choosing each approach over each question.

Table 3
Statistics of our second questionnaire results.

	Q1'	Q2'	Q3'	Q4'	Q5'	Q6'	average
our	71.43%	64.29%	100.00%	100.00%	92.86%	92.86%	86.90%
DH_mdf	28.57%	35.71%	0.00%	0.00%	7.14%	7.14%	13.10%
DH_rand	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
time (sec.)	34.23	24.80	9.35	9.02	12.68	13.37	17.24

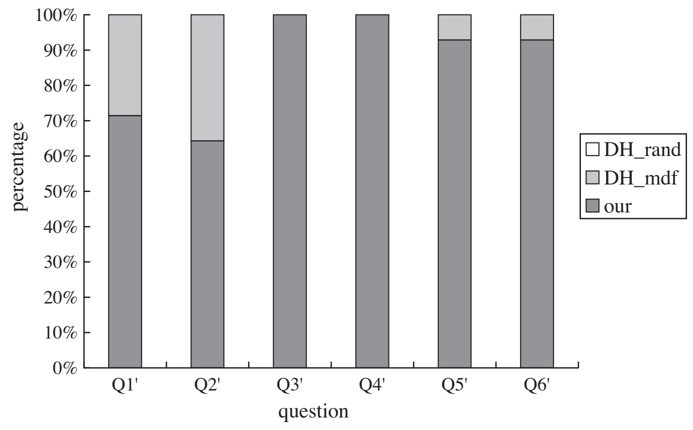


Fig. 16. The second student-based survey.

It is observed from Tables 2 and 3 that the average response time of the second survey (for complicated graphs) is shorter than that of the first survey by about 12 s, suggesting that tracing given boundary paths is in general helpful in recognizing the preservation of the user's mental map.

6. Conclusion and future work

In this paper, we have presented a simulated-annealing-based graph drawing algorithm, taking the mental map preservation into account. Our algorithm helps the user better preserve their mental maps in terms of preserving the drawing contours of graphs and the relative positions of nodes when a slight modification is made to the original graph. By using our algorithm, the user can reduce the time required to relearn the modified drawing. In addition, since the mental map preservation may be in conflict with the optimization of some aesthetic criteria, our implementation allows the user to adjust the weights of different criteria according to their requirements. Due to the limitations inherited from the simulated annealing scheme, the applicability of our algorithm is limited to graphs of small size. Hence, it would be of interest and importance to seek a more efficient mental-map preserving graph drawing algorithm capable of processing graphs of interest in the real world.

In addition to the factors considered in this paper, graph features such as color, node size, shape or line width may also play a key role in issues related to the preservation of the mental map. See [14] and [34] for the work of preserving the mental map from a psychological viewpoint. Therefore, a challenging line of future work is to take such additional features of graphs into account in the design of mental-map-preserving graph drawing algorithms. It is also of interest to incorporate the concept of mental map preservation into drawing dynamic graphs, which involves the evolution of graphs subject to a sequence of basic graph operations.

Acknowledgements

The authors thank the anonymous referees for comments that improved the content as well as the presentation of this paper.

References

[1] G.D. Battista, P. Eades, R. Tammassia, I.G. Tollis, Graph Drawing: Algorithms for the Visualization of Graphs, Prentice Hall, 1999.
[2] K.-F. Böhringer, F. Newbery Paulisch, Using constraints to achieve stability in automatic graph layout algorithms, in: Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, 1990, pp. 43–51.
[3] U. Brandes, B. Pampel, On the hardness of orthogonal-order preserving graph drawing, Proceedings of Graph Drawing 2008, Lecture Notes in Computer Science 5417 (2009) 266–277.
[4] U. Brandes, C. Pich, An experimental study on distance-based graph drawing, Proceedings of Graph Drawing 2008, Lecture Notes in Computer Science 5417 (2009) 218–229.
[5] S. Bridgeman, R. Tamassia, A user study in similarity measures for graph drawing, Journal of Graph Algorithms and Applications 6 (3) (2002) 225–254.
[6] R. Davidson, D. Harel, Drawing graphs nicely using simulated annealing, ACM Transactions on Graphics 15 (4) (1996) 301–331.
[7] C.F.X. de Mendonça, A Layout System for Information System Diagrams, Ph.D. thesis, University of Queensland, 1994.
[8] C.F.X. de Mendonça, T.A. Halpin, Automatic display of NIAM conceptual schemas diagram, Technical Report 209, Department of Computer Science, University of Queensland, 1991.
[9] U. Dogrusoz, E. Giral, A. Cetintas, A. Civril, E. Demir, A layout algorithm for undirected compound graphs, Information Sciences 179 (7) (2009) 980–994.
[10] U. Dogrusoz, K.G. Kakoulis, B. Madden, I.G. Tollis, On labeling in graph visualization, Information Sciences 177 (12) (2007) 2459–2472.
[11] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley-Interscience, 2001.
[12] P. Eades, A heuristic for graph drawing, Congressus Numerantium 42 (1984) 149–160.
[13] P. Eades, W. Lai, K. Misue, K. Sugiyama, Preserving the mental map of a diagram, Proceedings of Compugraphics 91 (1991) 24–33.
[14] S.L. Franconeri, A. Hollingworth, D.J. Simons, Do new objects capture attention?, Psychological Science 16 (2005) 275–281
[15] Y. Frishman, A. Tal, Movis: a system for visualizing distributed mobile object environments, Journal of Visual Languages and Computing 19 (3) (2008) 303–320.
[16] Y. Frishman, A. Tal, Online dynamic graph drawing, IEEE Transactions on Visualization and Computer Graphics 14 (4) (2008) 727–740.
[17] B. Genc, U. Dogrusoz, A layout algorithm for signaling pathways, Information Sciences 176 (2) (2006) 135–149.
[18] X. Geng, J. Xu, J. Xiao, L. Pan, A simple simulated annealing algorithm for the maximum clique problem, Information Sciences 177 (22) (2007) 5064–5071.
[19] W. He, K. Marriott, Constrained graph layout, Constraints 3 (4) (1998) 289–314.
[20] X. Huang, W. Lai, A. Sajeev, J. Gao, A new algorithm for removing node overlapping in graph visualization, Information Sciences 177 (14) (2007) 2821–2844.
[21] K. Kaufmann, D. Wagner (Eds.), Drawing graphs: methods and models, Lecture Notes in Computer Science, 2025, Springer, 2001.
[22] M. Kaufmann, R. Wiese, Maintaining the mental map for circular drawings, Proceedings of Graph Drawing 2002, Lecture Notes in Computer Science 2528 (2002) 12–22.
[23] S. Kirkpatrick, C.D. Gelatt Jr., M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.
[24] A.M. MacEachren, How Maps Work: Representation, Visualization, and Design, The Guilford Press, 1995.
[25] K. Misue, P. Eades, W. Lai, K. Sugiyama, Layout adjustment and the mental map, Journal of Visual Languages and Computing 6 (2) (1995) 183–210.
[26] F. Newbery Paulisch, W.F. Tichy, Edge: an extendible directed graph editor, Software – Practice and Experience 20 (S1) (1990) 63–88.
[27] S. North, Incremental layout in dynadag, Proceedings of Graph Drawing 1995, Lecture Notes in Computer Science 1027 (1996) 409–418.
[28] T. Poranen, A simulated annealing algorithm for determining the thickness of a graph, Information Sciences 172 (1-2) (2005) 155–172.
[29] H.C. Purchase, Which aesthetic has the greatest effect on human understanding?, Proceedings of Graph Drawing 1997, Lecture Notes in Computer Science 1353 (1997) 248–261
[30] H.C. Purchase, Performance of layout algorithms: comprehension, not computation, Journal of Visual Languages and Computing 9 (6) (1998) 647–657.
[31] H.C. Purchase, Metrics for graph drawing aesthetics, Journal of Visual Languages and Computing 13 (5) (2002) 501–516.
[32] H.C. Purchase, E. Hoggan, C. Görg, How important is the "mental map"? – an empirical investigation of a dynamical graph layout algorithm, Proceedings of Graph Drawing 2006, Lecture Notes in Computer Science 4372 (2007) 184–195.
[33] H.C. Purchase, A. Samra, Extremes are better: investigating mental map preservation in dynamic graphs, Proceedings of Diagrams 2008, Lecture Notes in Computer Science 5223 (2008) 60–73.
[34] R.A. Rensink, Change detection, Annual Review of Psychology 53 (2002) 245–277.

[35] P. Saffrey, H. Purchase, The ''mental map'' versus ''static aesthetic'' compromise in dynamic graphs: a user study, in: Proceedings of 9th Australasian User Interface Conference (AUIC 2008), vol. 76 of Conferences in Research and Practice in Information Technology, 2008, pp. 85–93.
[36] J. Six, I. Tollis, A framework for circular drawings of networks, Proceedings of Graph Drawing 1999, Lecture Notes in Computer Science 1731 (2000) 107–116.
[37] V. Tam, R. Mak, A. Kwan, Intelligent visualization techniques for reusable learning objects to facilitate an online learning environment, in: P. Tsang, R. Kwan, R. Fox (Eds.), Enhancing Learning through Technology, World Scientific, 2007, pp. 199–208.