Layered Drawings of Directed Graphs in Three Dimensions

Seok-Hee Hong[1,2] and Nikola S. Nikolov[1,3]
[1] **IMAGEN Program, National ICT Australia Ltd.**
[2] **School of IT, University of Sydney, NSW, Australia**
[3] **Department of CSIS, University of Limerick, Limerick, Ireland**

{seokhee.hong,nikola.nikolov}@nicta.com.au

Abstract

We introduce a new graph drawing convention for 3D layered drawings of directed graphs. The vertex set is partitioned into layers with all edges pointing in the same direction. The layers occupy parallel planes and vertices in each layer occupy two parallel lines. Thus, the traditional 2D layered drawing of a directed graph is split into two vertical walls each containing a 2D layered drawing. We outline a technique for 3D layered digraph drawing which is an extension to the well-known Sugiyama method for 2D layered digraph drawing, and we also present some experimental results.

1 Introduction

The visual representation of hierarchically organized data has received great deal of interest by the Graph Drawing community in recent years. The interest is driven by the increasing need of easy to comprehend drawings of large and complex networks and various diagrams used in the field of Software Engineering, such as class hierarchies, data-flow diagrams, etc. Hierarchies are commonly modeled by directed graphs (digraphs). Most of the research effort in digraph visualization has been related to improvements of various aspect of the Sugiyama method, which has been the most popular method for creating 2D drawings of digraphs (Eades & Sugiyama 1990, Sugiyama, Tagawa & Toda 1981).

The increasing availability of powerful graphic displays opens opportunities for developing new methods for 3D graph drawing which may possibly outperform the 2D drawing methods in terms of reducing the visual complexity and thus making the visual representation of large and complex graphs easier to comprehend (Ware & Franck 1996). There has been very little research on extending the Sugiyama method in 3D. To the best of our knowledge, the graph drawing literature describes two attempts for 3D visualization of digraphs. The first one is the technique introduced by Ostry who suggests wrapping a 2D layered drawing around a cone or a cylinder (Ostry 1996). The second attempt is the method used in the graph drawing system GIOTTO3D which is conceptually different from the Sugiyama method (Garg & Tamassia 1997). GIOTTO3D employs a simple 3-phase algorithm for drawing hierarchies in 3D. In the first phase a planarization method is used to draw the graph in 2D; in the second phase vertices and edges are assigned z-coordinate so that all edges point into the same vertical direction and the total edge span is minimized; and the third phase is to fix the shape of vertices and edges.

This paper presents a new attempt to extend the Sugiyama method for computing 3D drawings of directed graphs. We propose an extra step between the second and the third step of the 2D Sugiyama method and we also introduce a specific 3D version of the third step. Our method can be applied to any directed graphs, such as class hierarchies that originate from Software Engineering applications, hierarchical relationships in a social network, etc. In particular, we have experimented with some of the graphs in the Rome data set (Di Battista, Garg, Liotta, Tamassia, Tassinari & Vargiu 1997).

The paper is organized as follows. In the next section we introduce some definitions and an outline of the Sugiyama method. Then in Section 3 we describe our 3D version of the Sugiyama method. In Section 4 we show some drawings produced by our method and we compare them to the corresponding 2D drawings. Finally, in Section 5 we draw some conclusions from this work.

2 Preliminaries

Let $G = (V, E)$ be a digraph without directed cycles. $N^-(v) = \{u : (u, v) \in E\}$ and $N^+(v) = \{u : (v, u) \in E\}$. A *layering* of G is defined as an ordered partition $L = \{L_1, L_2, ..., L_h\}$ of its vertex set into h subsets, called *layers*, such that $(u, v) \in E$ with $u \in L_i$ and $v \in L_j$ implies $j < i$. A digraph with a layering is a *layered digraph*. A layering is *proper* if all edges are between vertices in adjacent layers. If this is not the case then after the second step of the Sugiyama method dummy vertices which subdivide long edges, i.e. edges which connect vertices in non-adjacent layers, are introduced. Formally, for each edge $e = (u, v)$ with $u \in L_i$, $v \in L_j$, and $j < i - 1$, we introduce $i - j - 1$ dummy vertices $d^e_{j+1}, d^e_{j+2}, \ldots, d^e_{i-1}$ into layers $L_{j+1}, L_{j+2}, \ldots, L_{i-1}$, respectively. We also replace edge e by edges $(u, d^e_{i-1}), (d^e_{i-1}, d^e_{i-2}), \ldots, (d^e_{j+2}, d^e_{j+1}), (d^e_{j+1}, v)$.

The Sugiyama method for layered digraph drawing consists of four steps as shown in Algorithm 1.

The first step is to remove all directed cycles from the graph by inverting the direction of some edges. At the second step the vertices of the digraph are partitioned into layers. At the next step a linear order is established for the vertices in each layer. And at the last step x- and y-coordinates of all vertices are decided as well as the shape of the edges. Various algorithms, which emphasize on different properties of the drawing, have been suggested for each step of the Sugiyama method (Di Battista, Eades, Tamassia & Tollis 1999).

Algorithm 1 Sugiyama Method for Layered Digraph Drawing

Step 1 (Cycle Removal): Remove all directed cycles by inverting the direction of some edges.

Step 2 (Layer Assignment): Partition the vertex set into layers.

Step 3 (Vertex Ordering): Set a linear order of the vertices within each layer.

Step 4 (Coordinate Assignment): Assign x- and y-coordinates to each vertex. Determine the shape of each edge.

3 3D Layered Drawing of Directed Graphs

In this section we propose a 3D extension to the Sugiyama method. Our method is outlines in Algorithm 2.

Algorithm 2 3D Layered Digraph Drawing

Step 1 (Cycle Removal): Remove all directed cycles by inverting the direction of some edges.

Step 2 (Layer Assignment): Partition the vertex set into layers.

Step 3 (Vertex Splitting): Split the vertices in each layer into two groups.

Step 4 (Vertex Ordering): Set a linear order of the vertices within each group in each layer.

Step 5 (Coordinate Assignment): Assign x-, y-, and z-coordinates to each vertex. Determine the shape of each edge.

We assume no difference in the first step, the cycle removal. Directed cycles can be removed by inverting the directions of some edges (Eades, Lin & Smyth 1993). We assume no difference in the second step, the layer assignment, either. In our experiments we use the longest-path layering algorithm followed by the vertex-promotion heuristic proposed by Nikolov and Tarassov for reducing the number of dummy vertices (Nikolov & Tarassov to appear). Large number of dummy vertices means unwanted long edges and edge bends which make the drawing difficult to comprehend. Furthermore, the more dummy vertices the slower the next steps of the Sugiyama method.

Our vision is that layers occupy parallel planes. Within each plane we place each vertex on one of two parallel lines. This is done at the third step of the proposed method by a vertex-split algorithm described in Section 3.1. That is, we propose splitting each layer L_i into two lines, A_i and B_i. We also propose placing all A_i lines ($i = 1..h$) in the same plane which we call *wall A*. Similarly, all B_i lines lie in the same plane, which we call *wall B*. The final 3D drawing consists of two 2D layered digraphs drawn in the two parallel walls with eventual edges between the two walls.

By dividing the hierarchy into two walls we hope that the number of edge crossings in the two walls will be smaller than in a 2D drawing when the same algorithms for layer assignment and crossing minimization are applied. A 3D drawing can also be combined with appropriate navigation techniques which can significantly decrease the visual complexity. For example, each wall can be viewed separately, the camera may move along the edges between the walls, etc.

We have chosen to use no more than two walls and minimize the number of edges between them in order to avoid occlusion problems. This way we can always rotate the 3D drawing with two parallel walls in order to see each wall in detail.

Our version of the vertex-ordering step is presented in Section 3.2. For the fifth step we propose assigning x- and y-coordinates and edge shapes within each of the two walls by some of the algorithms used in the 2D Sugiyama method.

In the pictures shown in Section 4 we assign x-coordinates to the vertices by a variation of the technique proposed by Gansner et al. (Gansner, Koutsofios, North & Vo 1993). It consists of computing a layering of an auxiliary graph by the network simplex layering algorithm. We used a different layering technique instead, the one described above in the second step. The distance between the layers is constant.

3.1 Algorithm for Vertex Split

Since we perform the vertex split step after the introduction of dummy vertices we assume that $G = (V, E)$ is a properly layered digraph with a layering $L = \{L_1, L_2, ..., L_h\}$, i.e. each edge connects vertices in adjacent layers. By splitting the vertices between two walls we partition the edge set of a digraph into two subsets: *wall edges* and *ladder edges*. Wall edges are edges with both endpoints placed in the same wall, and ladder edges are edges with one endpoint in wall A and the other in wall B.

The vertex split should be balanced, i.e. the number of vertices in wall A should be close to the number of vertices in wall B. In order to avoid occlusion it is also preferable to have as few edges as possible between the two walls. Based on these two criteria we define the following minimization problem.

Balanced 2-layer Partitioning Problem
Consider two adjacent layers L_{i-1} and L_i. Let L_{i-1} be partitioned into subsets A_{i-1} and B_{i-1}. Find a partition of L_i into subsets A_i and B_i such that the number of edges between A_{i-1} and B_i plus the number of edges between A_i and B_{i-1} is the minimum.

We propose a greedy algorithm, Algorithm 3, for vertex split of layer L_i which solves the balanced 2-layer partitioning problem optimally.

Algorithm 3 GREEDY_SPLIT(i)

```
A_i ← φ
B_i ← φ
for all v ∈ L_i do
  if |N⁺(v) ∩ A_{i-1}| > |N⁺(v) ∩ B_{i-1}| then
    A_i ← A_i ∪ {v}
  else
    B_i ← B_i ∪ {v}
  end if
end for
if |A_i| > |B_i| then
  X is a synonym of A and x is a synonym of B
else
  X is a synonym of B and x is a synonym of A
end if
while (|L_i| is even and |X_i| > |x_i|) or (|L_i| is odd
and |X_i| > |x_i| + 1) do
  move vertex v ∈ X_i with the minimum |N⁺(v) ∩
  X_{i-1}| − |N⁺(v) ∩ x_{i-1}| to x_i
end while
```

Theorem 3.1 *For given i, A_{i-1}, and B_{i-1}, Algorithm 3 splits layer L_i into A_i and B_i so that:*

Property 1: $|A_i| = |B_i|$ *if* $|L_i|$ *is even, and* $||A_i| - |B_i|| = 1$ *if* $|L_i|$ *is odd.*

Property 2: The number of ladder edges between L_{i-1} *and* L_i *is the minimum.*

Proof. Property 1 is trivially implied by the **while** loop. Assume there exists another split of layer L_i into sets A'_i and B'_i that satisfies Property 1 and results in the minimum number of ladder edges between L_{i-1} and L_i. That is, assume the split defined by A_i and B_i either is not unique or does not satisfy Property 2. We will show that this assumption leads to contradiction which implies that the split defined by Algorithm 1 is the only one that satisfy both Property 1 and Property 2.

Without loss of generality assume that set A_i was larger than set B_i after the **for** loop in Algorithm 3 and let C_i be the set of vertices moved from A_i to B_i in the **while** loop. Let also A_i^{before} and A_i^{after} denote A_i before and after the **while** loop respectively. Similarly, let B_i^{before} and B_i^{after} denote B_i before and after the **while** loop respectively.

Case 1. There is a vertex $v \in A'_i \cap B_i^{before}$. By moving v to B'_i we will reduce the number of ladder edges by $|N^+(v) \cap B_{i-1}| - |N^+(v) \cap A_{i-1}| > 0$. Let $A''_i = A'_i - v$ and $B''_i = B'_i + v$. If the split defined by A''_i and B''_i does not satisfy Property 1 because B''_i is too large then clearly $|B''_i| > |B_i^{before}|$ and there is a vertex $u \in B''_i \cap A_i^{before}$ which can be moved to A''_i to satisfy Property 1 and further reduce the number of ladder edges. This is a contradiction with the assumption that A'_i and B'_i define a split with the minimum number of ladder edges, which is different from the split defined by A_i and B_i.

Case 2. $A'_i \subset A_i^{before}$. If A'_i is different from A_i^{after} then $B'_i \cap A_i^{after}$ is not empty. Let $v \in B'_i \cap A_i^{after}$. By moving v to A'_i we will reduce the number of ladder edges by $\alpha = |N^+(v) \cap A_{i-1}| - |N^+(v) \cap B_{i-1}| > 0$. Let $A''_i = A'_i + v$ and $B''_i = B'_i - v$. If the split defined by A''_i and B''_i does not satisfy Property 1 because A''_i is too large then clearly $|A''_i| > |A_i^{after}|$ and there is a vertex $u \in C_i \cup A''_i$ with $\beta = |N^+(u) \cap A_{i-1}| - |N^+(u) \cap B_{i-1}| < \alpha$. By moving u to B''_i we will achieve a split which results in $\alpha - \beta > 0$ less ladder edges than the spilt defined by A'_i and B'_i. This is a contradiction with the assumption that there is another split which satisfies Properties 1 and 2. □

For the whole vertex split step we propose choosing a random split of the first layer L_1 which has more than two vertices and then applying Algorithm 3 for all layers from L_2 to L_h. Clearly, this can be done in $O(|V|)$ time.

3.2 Vertex Ordering

For the next step of the Sugiyama method we propose an extension to the traditional layer-by-layer sweep technique. The order of the vertices in L_1 is fixed randomly and then the order of the vertices in each layer L_i for $i = 2..h$ is computed on the basis of the already fixed order of vertices in layer L_{i-1}. Once the vertex order of all layers is computed the same can be repeated in the opposite direction, from L_h to L_1, and so on until a terminating condition is met.

Various algorithms have been proposed for computing the order of vertices in layer L_i based on the fixed vertex order in L_{i-1}. They are aimed at either minimizing the number of edge crossings between the two layers (two-layer crossing minimization approach) or drawing as large as possible subset of edges between the two layers without crossings (maximum planar subgraph approach) (Di Battista et al. 1999, Mutzel 2001).

We propose a version of the well-known barycenter heuristic for two-layer crossing minimization. Assume $p : V \rightarrow \mathbb{N}$ is a function that gives the order of vertices in both $A_{i-1} = \{v_1, v_2, \ldots, v_k\}$ and $B_{i-1} = \{u_1, u_2, \ldots, u_l\}$. Let $p(v_i) = i$ and $p(u_i) = i$. Then we consider the following two options:

- *Barycenter Option 1 (BO1)*: $p(w) = \frac{\sum_{v \in N^+(w)} p(v)}{|N^+(w)|}$ for each $w \in L_{i-1}$
- *Barycenter Option 2 (BO2)*: $p(w) = \frac{\sum_{v \in N^+(w) \cap A_{i-1}} p(v)}{|N^+(w) \cap A_{i-1}|}$ for each $w \in A_{i-1}$ and $p(w) = \frac{\sum_{v \in N^+(w) \cap B_{i-1}} p(v)}{|N^+(w) \cap B_{i-1}|}$ for each $w \in B_{i-1}$.

With BO1 we assign a position to vertex w based on the positions of all its neighbours in layer L_i, while with BO2 the position of w depends only on its neighbours in its wall. In both cases we resolve conflicts only within A_i and B_i, i.e. if $p(u) = p(v)$ for some $u \in A_i$ and some $v \in B_i$ we do not consider it a conflict.

We employ two termination conditions for the layer-by-layer sweep. After going through all the layers in one direction we compute the number of edge crossings and if there has not been improvement compared to the previous layer-by-layer sweep in the opposite direction we stop. We also set up an upper bound on the number of sweeps which if reached we terminate the layer-by-layer sweep anyway. We chose to perform no more than 24 sweeps in both directions which is the upper bound used in the **dot** graph drawing system (Gansner et al. 1993).

4 Experimental Work

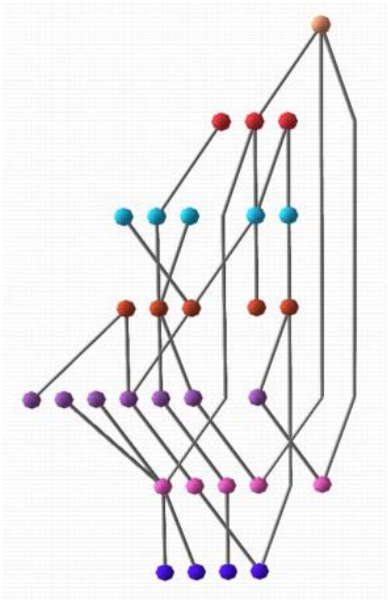
We implemented the described above 3D version of the Sugiyama method and compared the two options for vertex ordering to each other as well as to the corresponding 2D drawings of the same graphs. The 2D drawings are also made by our system assuming wall B is empty, i.e. all vertices were placed in wall A at the vertex-split step.

We used three graphs, grafo115 with 30 vertices, grafo11093 with 100 vertices, and grafo11138 also with 100 vertices, from the Rome graph set (Di Battista et al. 1997). The results are presented in Figures 1-3. We implemented our 3D drawing technique as part of GEOMI [1] and rendered the 3D drawings with the help of WilmaScope[2] (Dwyer & Eckersley 2004).

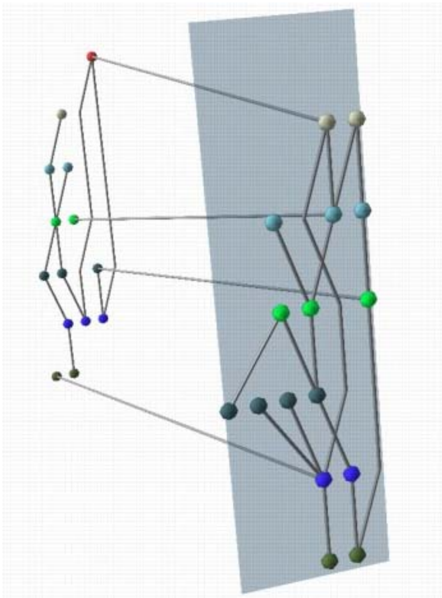
Each of Figures 1-3 shows a 2D drawing and two 3D drawings of the same digraph. The 2D drawing of the graph in Figure 1 has 12 edge crossings. As shown in Figures 1(b) and 1(c), there are 2 and 3 edge crossings between wall edges in the 3D drawings of grafo115 with BO1 or BO2, respectively. Similarly, we were able to reduce significantly the number of edge crossings between wall edges in the 3D drawings of grafo11093 and grafo11138 shown in Figure 2 and Figure 3 respectively. Our initial results do not show a clear winner between the two vertex ordering options BO1 and BO2. Further computational experiments are necessary to determine which of the two options is preferable. Our initial observation is that both of them significantly reduce the number of edge crossings between edges in the same wall.

[1] GEOMI (Geometry for Maximum Insight) is a visual analytic tool being developed by NICTA for the visualisation and analysis of large and complex networks such as social networks and biological networks.

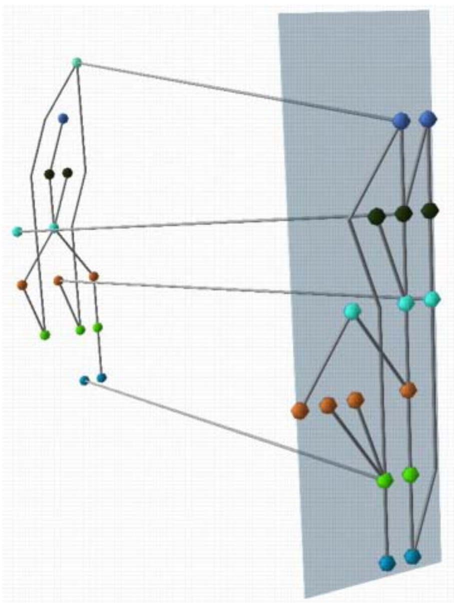
[2] http://www.wilmascope.org



(a) 2D drawing with 12 edge crossings

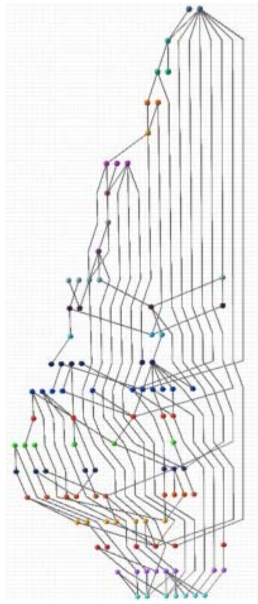


(b) 3D drawing with BO1, 2 crossings between wall edges

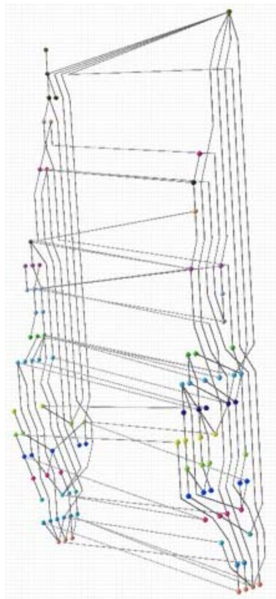


(c) 3D drawing with BO2, 3 crossings between wall edges

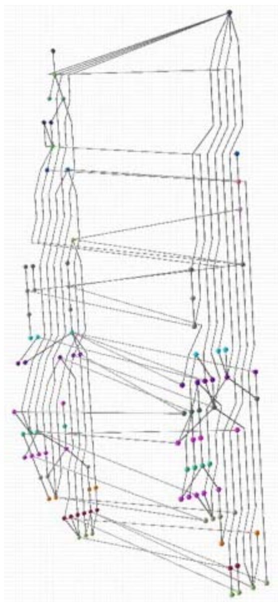
Figure 1: Layered drawings of grafo115



(a) 2D drawing with 195 edge crossings

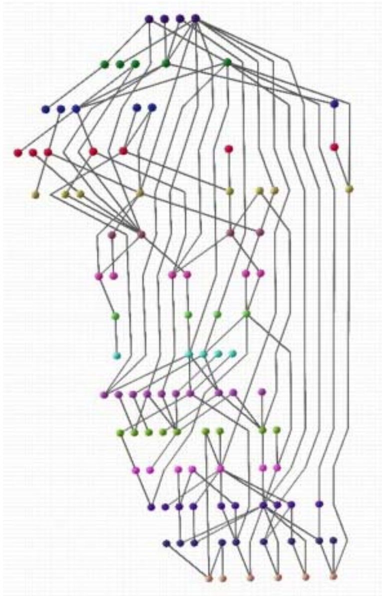


(b) 3D drawing with BO1, 56 crossings between wall edges

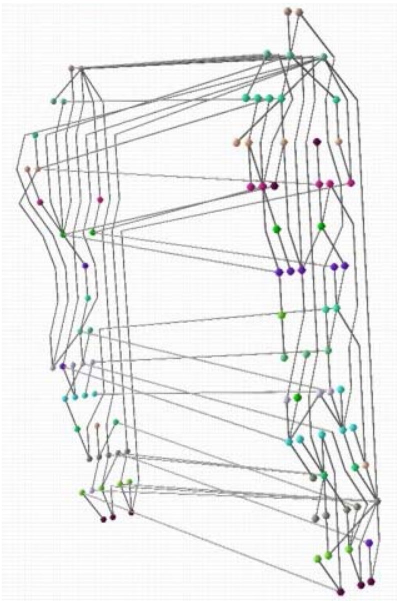


(c) 3D drawing with BO2, 48 crossings between wall edges

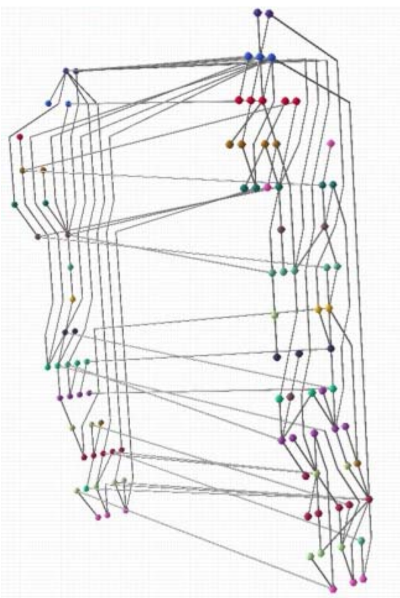
Figure 2: Layered drawings of grafo11093



(a) 2D drawing with 180 edge crossings



(b) 3D drawing with BO1, 31 crossings between wall edges



(c) 3D drawing with BO2, 24 crossings between wall edges

Figure 3: Layered drawings of grafo11138

5 Conclusions and Current Work

We introduce a 3D extension to the Sugiyama method. We propose splitting the drawing into two walls; each wall contains a 2D drawing of a layered digraph; the number of edges with endpoints in different walls is minimized subject to the random split of vertices in the first layer. This is done by introducing an additional vertex-split step into the Sugiyama method after the layer assignment step. We have also investigated two versions of the layer-by-layer sweep for ordering the vertices within each layer and wall which has shown similar results. The first computational results suggest that the proposed 3D graph drawing convention results in reducing the visual complexity of layered graph drawings significantly. We have experimented only with the barycenter heuristic for vertex ordering. Further experiments with other vertex ordering techniques may result in better drawings. It is also possible to develop new vertex ordering heuristics specific for the 3D layered graph drawings with two walls. We are also looking at the possibility of defining new optimization problems arising from 3D drawing aesthetic criteria.

We are planning extensive computational tests and user studies with real-world data sets from various areas, such as Software Engineering, Social Network Analysis, etc. In particular, our intention is to tune our method for 3D layered drawings of large and complex digraphs. A further step will be to design and implement appropriate navigation techniques for 3D layered digraph layouts.

References

Di Battista, G., Eades, P., Tamassia, R. & Tollis, I. G. (1999), *Graph Drawing*, Prentice Hall.

Di Battista, G., Garg, A., Liotta, G., Tamassia, R., Tassinari, E. & Vargiu, F. (1997), 'An experimental comparison of four graph drawing algorithms', *Computational Geometry: Theory and Applications* **7**, 303–316.

Dwyer, T. & Eckersley, P. (2004), Wilmascope a 3d graph visualisation system, *in* M. Junger & P. Mutzel, eds, 'Graph Drawing Software', Mathematics and Visualization, Springer-Verlag, pp. 55–76.

Eades, P., Lin, X. & Smyth, W. F. (1993), 'A fast and effective heuristic for the feedback arc set problem', *Information Processing Letters* **47**, 319–323.

Eades, P. & Sugiyama, K. (1990), 'How to draw a directed graph', *Journal of Information Processing* **13**(4), 424–437.

Gansner, E. R., Koutsofios, E., North, S. C. & Vo, K.-P. (1993), 'A technique for drawing directed graphs', *IEEE Transactions on Software Engineering* **19**(3), 214–230.

Garg, A. & Tamassia, R. (1997), GIOTTO: A system for visualizing hierarchical structures in 3d, *in* S. North, ed., 'Graph Drawing: Symposium on Graph Drawing, GD '96', Vol. 1190 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 193–200.

Mutzel, P. (2001), 'An alternative method to crossing minimization on hierarchical graphs', *SIAM Journal on Optimization* **11**(4), 1065–1080.

Nikolov, N. & Tarassov, A. (to appear), 'Graph layering by promotion of nodes', *Special issue of Discrete Applied Mathematics associated with the IV ALIO/EURO Workshop on Applied Combinatorial Optimization* .

Ostry, D. (1996), Some three-dimensional graph drawing algorithms, Master's thesis, University of Newcastle.

Sugiyama, K., Tagawa, S. & Toda, M. (1981), 'Methods for visual understanding of hierarchical system structures', *IEEE Transaction on Systems, Man, and Cybernetics* **11**(2), 109–125.

Ware, C. & Franck, G. (1996), 'Evaluating stereo and motion cues for visualizing information nets in three dimensions', *ACM Transactions on Graphics* **15**(2), 121–140. a reference for advantages of 3D visualisation of graphs.