AUTOMATIC METRO MAP DESIGN TECHNIQUES

Jonathan M. Stott, Peter Rodgers
Computing Laboratory, University of Kent, Canterbury, United Kingdom
jms30@kent.ac.uk; P.J.Rodgers@kent.ac.uk

ABSTRACT

We describe a computer-based system to automatically lay out metro maps using multicriteria optimization. The starting layout for the method is the geographic layout or a sketch of the map. The system attempts to improve this layout. It measures the map by calculating a number of criteria. These criteria are weighted and summed together: stations are moved if the sum of the weighted criteria is reduced. An iteration of the method consists of attempting to move each station in the map. There are several enhancements to the method: stations are placed on an grid, a clustering algorithm is used to find groups of stations that can be moved together, and labelling of stations is performed with a number of additional criteria. Example results using real-world metro maps look promising.

1. INTRODUCTION

Metro maps can be seen worldwide and have possibly become the "most memorised cartographic items in the world" [19]. Ever since construction of the first railways began in the early 19th century, there has been a need to map the networks. This is particularly the case with metro networks, where people needed to plan short journeys across a city or metropolitan area. Before long, the metro networks were rapidly expanding and the traditional topographic maps were quickly becoming cluttered and difficult to read. The London Underground diagram [23], designed by Harry Beck and first published in 1933 [11], marked a significant departure from the more traditional topographic maps. The diagram works by straightening meandering lines with line segments drawn either horizontally, vertically or diagonally at 45° and by using a non-linear scale so that the central area of the diagram is shown at a larger scale than the extremities. The effect was to produce a diagram that proved to be extremely clear and concise and has been embraced as an iconic image of London. Since the original diagram was published, the key design features have remained even though the London Underground has since expanded greatly.

It might appear at first glance that metro maps are simple to design. However, this is almost always not the case and it can require a cartographer with lots of skill and design knowledge to be able to produce effective maps. There are many aspects which contribute together to produce effective maps: the position of stations and the lines between stations, the width of lines; the symbols used to represent stations (circles or dots or ticks are most often used, but not exclusively so); the font and size of text used for labelling; the amount of metadata to include (such as roads, rivers or coastlines); and the size, shape and resolution (for computer displays) of the medium being used to display the maps. Even what might seem to be trivial changes – slightly increasing the size of labels, for example – can drastically alter the overall appearance of a map. The challenge faced by metro map designers is to balance these issues so that the map is as easy to use as possible by people travelling on the metro system.

Metro maps can be represented as either schematic diagrams or as graphs. Previous work has looked at drawing metro maps using a force-directed algorithm with a graph model or a generalization of schematic diagrams. Hong *et al.* [12] implemented five methods for drawing metro maps using modifications of spring-based algorithms for drawing graphs [4]. As each method was further refined to draw metro maps the results became more acceptable. However, their approach was severely limited by not taking the topology of the metro systems into account. They claimed that passengers on a metro train wouldn't be concerned as to whether they were travelling north or south – this seems somewhat unreasonable, especially as many metros use northbound or southbound trains. It would also suffer problems if metadata such as roads, rivers and coastlines were to be included on the finished map. A number of papers consider drawing diagrams using simplification or generalization of schematic diagrams [1][2][3][6][7][10]. While none of these specifically consider drawing metro maps, their approaches could possibly be modified for drawing metro maps. The idea of generalizing schematic diagrams is that the diagram is modified such that the essential information required for following a route is preserved (such as intersections and approximate direction). Distances are usually exaggerated to enlarge areas containing dense information (such as intersections in a city) and areas containing sparse information (such as a long stretch of highway) are reduced in scale.

As well as generalizing metro maps, it is important that stations are labelled with their names. Labelling point features is discussed in a survey by Christensen, Marks and Shieber [9]. Typically, labels are placed in one of a set of finite positions (the labelling space) around the point feature with some order of preference indicating which position is most preferable. There are a number of approaches that attempt to solve this, including using an exhaustive search of all the potential positions in the labelling space for each point feature, a greedy algorithm or a gradient descent algorithm such as simulated annealing. Most attempts at labelling maps or graphs work on static maps. However, this might not be ideal if the labels have an effect on how the map or graph is drawn. It might be possible to improve the labelling for a map or graph if the size and position of labels are taken into account while the diagram is being drawn.

The approach we have taken uses multicriteria optimisation and a hill climber [22]. We use a graph model to represent the metro maps with stations represented by nodes and connections between stations represented by edges. We have implemented a total of eight criteria – three for positioning labels and five for positioning nodes. Each criterion is weighted and nodes and labels are repositioned such that the total of the weighted criteria is always reduced. The process continues for a fixed number of iterations. We have also implemented a node clustering algorithm so that groups of nodes can be moved simultaneously where moving individual nodes would not work. We have implemented our method in a software tool so that we can experiment with using real-world examples.

While we have concentrated on drawing real-world metro maps, the same design techniques can be applied to diagrams showing other data. The metro map metaphor has been proposed for a varied group of applications such as presenting project plans, book outlines, academic course structures and web page layout [5][8][18][21]. Our method can be easily adapted to these areas.

The remainder of this paper is organised as follows: the Section 2 introduces our technique used for laying out metro maps; Section 3 shows examples of our method at work and provides a critical analysis as to their quality and effectiveness; Section 4 concludes and discusses potential further work.

2. METRO MAP LAYOUT

Graph Model

The model we use for the abstract representation of a metro map is a graph. In this case, a graph, G, is a set of nodes, V, with connections between pairs of nodes represented by a set of edges, E. For drawing metro maps, we use the nodes to represent stations on the network and edges to represent a single connection between two stations. In some cases, there may be several edges connecting two nodes. We also use the term 'line' to represent a subset of edges that form a particular line on the network (such as the Central or Northern Lines on the London Underground map). A graph model is used because of its programmatic flexibility which makes tasks such as finding neighboring nodes or incident edges relatively easy.

The graph is embedded on an integer square grid, as shown in Figure 1. This means that nodes can only be centred on grid intersections, but there is no requirement for edges to follow grid lines. The spacing between adjacent grid is denoted by g. The grid allows us to dramatically reduce the number of potential locations for nodes and also makes producing orthogonal maps easier as nodes are more likely to be in line with each other.
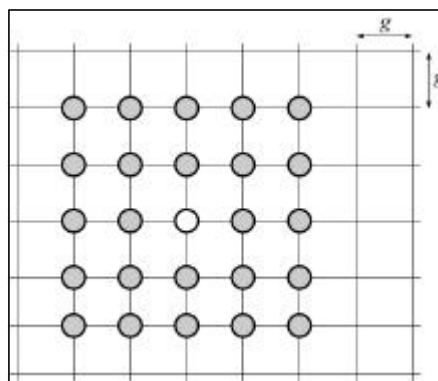

Figure 1. Grid used for embedding the metro map graph where g represents the grid spacing. The potential locations considered when moving the white node with a movement radius $r = 2$ are shown with grey nodes.

An important consideration which greatly affects the finished maps is that of the initial layout of the map. There are essentially two possible ways of generating an initial layout for the map. The first is to start with a random layout; the second is to start with the actual geographic layout of the map.

Starting with a random layout is going to make producing even a slightly geographically correct map very difficult. Nodes would have no concept of where they should be, so it would be impossible to say that a certain line should be oriented in a particular direction (north-to-south or east-to-west) or getting nodes which are close in reality to appear close on the final map. Even if nodes had some concept of their relative positions (say, one node should be north of another or that a group of nodes are close together), it would likely require a large number of iterations and/or massive fluctuation in the starting layout that it becomes very difficult to predict and reason about potential movements of nodes.

Starting with an actual geographic layout is much more likely to produce better results than a random layout. The entire method then becomes a process of iterative refinement – fewer and smaller node movements would theoretically be required to produce finished maps of acceptable quality than starting with a random layout. Finding the geographic locations of stations in order to position the nodes becomes a time-consuming process, but precise accuracy is not always required. A simple freehand sketch based on knowledge of the geography of the metro system might suffice, particularly for simple maps.

We start with a geographic layout (or a close alternative using a sketch of the map). However, we need to make sure that nodes are centred on grid intersections, so a way to snap the nodes to the grid is needed. Calculating the nearest grid intersection to a node is simple but care must be taken to ensure that more than one node does not share the same grid intersection. In the case of contention for a particular intersection, the node being snapped should be moved to the nearest grid intersection that is vacant. If the value of g is too large, particularly dense graphs (especially areas where the average length of an edge is less than $0.5g$) may make it difficult to find points close to the starting point for the node. In this case, the value of g should be reduced.

Multicriteria Optimisation

Our process is built around a multicriteria optimisation method that combines a number of criteria that are judged to affect the quality of the graph and some method to change the layout of the graph. We take a hill climbing approach, meaning that we are always making improvements to the graph with each change in the layout.

Multicriteria optimisation is an iterative process. At each iteration, an attempt is made to move each node, followed by each cluster, followed by each node label. The number of iterations can either be fixed beforehand or the process can be stopped if no further improvement can be made. We always use a fixed number of iterations, the number of which are indicated for the examples presented in Section 3. Care is made to ensure that the method is deterministic. This allows us to see how subtle changes in the criteria weightings affect the finished maps without randomness becoming a factor.

Criteria

We have a total of five criteria, c_1 to c_5, which we use in order to measure the quality of the generated maps. Each criterion measures some geometric property of the map, such as the number of edge crossings or the length of edges. The intention is that a high value for a particular criterion indicates potential improvement of that particular geometric property. Each criterion, c_i, is weighted with a weighting, w_i, such that their values are comparable when totalled (so that no one criterion has a particularly large influence over the others). The hill climber uses the total of the weighted criteria, t, in order to compare iterative improvements to the map when an attempt to move a node is made.

Edge Crossings. In the case of metro maps, edge crossings imply some kind of topographic feature such as two unconnected lines crossing. If an edge crossing is intentional, then a dummy node can be inserted at the crossing point and the map drawn with that node in place. The dummy node can then be removed after the map is drawn. However, unintentional edge crossings can affect the readability of the map [20]. The edge crossings criterion, c_1, sums the number of edges that cross.

Edge Length. A common feature of metro maps is that stations should be spaced evenly along lines and that the spacing should be should be consistent across the entire map. This is because the map is drawn to a varying scale such that the scale decreases towards the extremities of the map. We decided that the edge length criterion, c_2, should be based on the grid spacing g – the "ideal" spacing between two nodes should be exactly one grid spacing. Therefore, the criterion should penalize edges that are greater than g. The edge length criterion is calculated using (1).

$$c_2 = \sum_{e \in E} \left(\frac{|e|}{g} - 1 \right) \tag{1}$$

where $|e|$ is the length of edge e. We do not need to worry about edges where $|e| < g$ as the only way this could happen is if a node was not centred on a grid intersection which is disallowed.

The edge length criterion has one interesting side-effect concerning diagonal edges. In this case, the minimum length value of $|e|$ is $\sqrt{2}$. This has the effect of penalising diagonal edges slightly even when they are the shortest possible length. Therefore, this criterion marginally prefers vertical and horizontal edges over diagonal edges.

Angular Resolution. In some metro maps there are occasions where many lines pass through a single node so that the node has many incident edges. If the edges are drawn such that there is only a small angle between adjacent edges then it can be difficult to distinguish between them (particularly if the edges are similarly coloured). It therefore seems reasonable that there is some criterion that ensures that there is as large an angle as possible between adjacent edges incident to a node. The angular resolution criterion, c_3, calculated using (2), has this effect.

$$c_3 = \sum_{v \in V} \sum_{\{e_1, e_2\} \in E_v} \left| \frac{360°}{\deg(v)} - \boldsymbol{q}(e_1, e_2) \right| \tag{2}$$

The criterion takes each pair of edges (e_1, e_2) from the set of edges E_v incident to the node v and finds the angle $\boldsymbol{q}(e_1, e_2)$ between them. It then sees by how much $\boldsymbol{q}(e_1, e_2)$ differs from the ideal spacing of all the edges connected to v. The absolute value is used so that edges which are too close together get penalised as much as edges which are far apart; if $\boldsymbol{q}(e_1, e_2)$ is the same as the average angle, then the difference will be zero.

Line Straightness. One of the important features of metro maps is that lines appear to pass through nodes so that the entry edge is more-or-less directly opposite the exit edge. It is not desirable if the line appears to turn sharply (so that it makes a 90° or 135° turn). This is made all the more important is there are two (or more lines) passing through a node – if both of the entry edges are opposite each other and both of the lines make a 90° turn so that the exit edges are opposite, the readability of the map is degraded (especially if the colour of the lines are similar).

To counter this, we introduce the line straightness criterion, c_4, given by (3).

$$c_4 = \sum_{v \in V} \left(\sum_{\substack{e_1, e_2 \in E \text{ where } e_1 \text{ and } e_2 \\ \text{are the only two edges of} \\ \text{the same line incident to } v}} \boldsymbol{s}(e_1, e_2) \right) \tag{3}$$

where $\boldsymbol{s}(e_1, e_2)$ is the angle between adjacent edges e_1 and e_2. If e_1 and e_2 are parallel (i.e. they pass through v in a straight line), $\boldsymbol{s}(e_1, e_2)$ evaluates to 0°; if the two edges are at right angles, then $\boldsymbol{s}(e_1, e_2) = 90°$. The intuitive effect of this is to penalise turns in edges where $\boldsymbol{s}(e_1, e_2)$ is large more than turns where $\boldsymbol{s}(e_1, e_2)$ is small or zero.

Four-gonality. The purpose of this criterion is to ensure that edges are drawn at some multiple of 45°, either orthogonally (vertically or horizontally) or diagonally with respect to the grid. The four-gonality criterion, c_5, has the effect of penalising edges that are not some multiple of 45° and is calculated using (4).

$$c_5 = \sum_{\{u,v\} \in E} \left| \sin 4 \left(\tan^{-1} \frac{|y(u) - y(v)|}{|x(u) - x(v)|} \right) \right| \tag{4}$$

where (u, v) is an edge between nodes u and v, and $y(v)$ and $x(v)$ are the y-and x-coordinate of node v respectively.

Total Weighted Criteria. The total weighted criteria is given by (5).

$$t = w_1 c_1 + w_2 c_2 + w_3 c_3 + w_4 c_4 + w_5 c_5 \tag{5}$$

The values for w_i can be modified by the user depending on the characteristics of the particular metro map being drawn.

Node Movement

The way that nodes are moved greatly affects the outcome of the final drawing of the metro map. There are a number of points that need to be considered when selecting a position to move a node to: the total value of the weighted criteria, t; whether or not another node occupies that grid intersection; whether moving the node would occlude other nodes or edges; how far to move the node; whether the distance to move the node is reduced with each iteration (cooling); and whether the cyclic ordering of edges incident to a node would change.

Our approach is to specify a maximum radius within which a node can move. This is given as some multiple, r, of the grid spacing g. As the whole process effectively refines a sketch of the map or the geographic layout of the map, the value of r is usually fairly small (generally than 10). Larger values of r would allow movements that could alter the map so that it differs too greatly from reality. A larger value of r is chosen for maps with small values of g so that nodes can move greater distances if there are many grid intersections between connected nodes. In the case of a large map, such as the London Underground map, g has to be small to allow for enough grid intersections for nodes in the dense centre of the map, while the extremities are relatively sparse. An example of the potential movements for a node when $r = 2$ is shown in Figure 1.

A cooling process is used whereby r is reduced to 1 by the last iteration. We use a linear cooling schedule, but alternative schedules, such as a logarithmic or an irregular schedule, might also be applied. The purpose of cooling is to allow large movements to be made during early iterations, with later iterations moving nodes less and less to allow for finer refinement. We experimented with a number of different cooling schedules such as a logarithmic schedule or an irregular schedule. However, the difference between results using each different schedule was negligible. It therefore made sense to use the least computationally expensive linear cooling schedule.

When considering potential movements for a node, the initial value of t, t_0, is first calculated. Each grid intersection up to r intersections from the initial node location is tested by moving the node there and calculating t. A set of locations, T, is remembered for each potential movement where $t < t_0$. The node is moved to the location in T that has the smallest value of t. In the case of no potential movements being discovered, the node is returned to its starting location; if there are more than one location with the same smallest value of t, we select the first improved location that was found.

Care must be made to ensure that locations that might cause another node or edge to become occluded are not considered. Such occlusions might occur if a node is moved on top of another node or edge, or a move is made whereby an edge incident to the node being moved ends up occluding another node or edge. This effectively means that edge crossings cannot be introduced.

Also considered is the clockwise cyclic ordering of edges incident to a node. In some examples (particularly with smaller maps) it is possible that in moving either a single node or a cluster of nodes, the topology of the map is drastically altered. A small example of this would be a Y branch in a line; the arms of the branch could swap round. To stop this from happening, a rule is introduced so that the clockwise ordering of the set of edges incident to a node is always preserved. In the case of the Y branch, the arms can only swap round if the clockwise ordering of the edges incident to the pivot node (where the arms join) is changed.

Labelling

Labelling is an integral part of metro maps and as such it seems logical that it should form an integral part of our multicriteria optimisation method. To this end, a number of criteria are introduced for label placement. Figure 2 shows our chosen labelling space. Numbers show preferred label positions, with 1 being the most preferable position and 8 being the least preferable position.
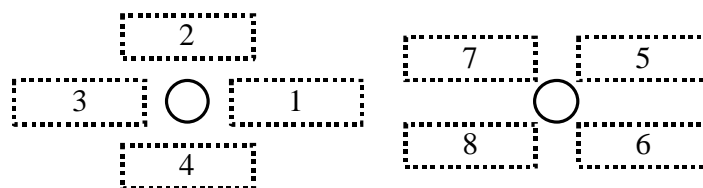

Figure 2. Labelling space for labelling nodes. Numbers represent the preferential order of label positions with 1 being the most preferable position and 8 being the least preferable position.

We have implemented labelling as an integral part of our multicriteria optimisation process. This means that we need both a strategy for moving labels and a way to determine which position is best. Labelling was performed once at each iteration, usually after the individual nodes were moved. We could have attempted to label the map during the node movement and considered the label positions for each potential node location, but this proved to be excessively slow. Potential label positions are tested in the same way that potential node locations were tested when moving nodes; the total weighted label criteria is calculated for each position and the one with the best improvement to the initial label position is chosen.

We implemented four labelling criteria. The first three criteria count the number of nodes, edges and other labels that intersect the label. When checking for an intersection, the label is considered as a rectangle fitted around the text of the label. The fourth criterion allows us to place a preference on the label positions by using the numbers given in Figure 2 as the value of the criterion. As with the node movement criteria, the labelling criteria are weighted with individual weightings.

Node Clustering

One problem that we discovered when testing the method was a particular kind of local minima where edges that were too long could not be reduced in length. Typically, a small group of nodes might be connected together towards the edge of the map, but they are connected to the rest of the map by one or more edges that are relatively long. These overly-long edges cannot be reduced in length by moving a single node. The intention with node clustering is to find these groups of nodes and to move them as a group. Exactly the same process is used as for moving a single node, except that the whole cluster is moved to each potential location and the criteria are measured. Again, if a better location for the cluster is discovered, the cluster is moved to the location with the best improvement to t.

To be able to move clusters of nodes, we first need to define what constitutes a cluster. A cluster is a subset of nodes which are connected by edges which are less than $2g$ apart (Figure 3). This results in clusters where every node in the cluster is at an adjacent intersection of the grid. Occasionally, a single node cluster may be formed: this is perfectly acceptable.
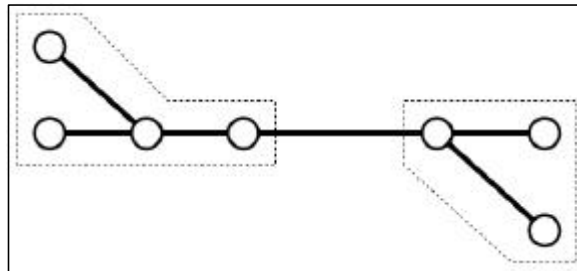

Figure 3. Example of node clustering - clusters are depicted by dotted lines.

3. EXAMPLE RESULTS

We implemented our method using a visual software tool written in the Java™ 5.0 programming language [13]. Using a visual tool allows us to change criteria weightings and experiment with the way they affect the resulting map. In the following examples, the initial layout is contrasted with the final layout as generated by our software tool. A critique of the performance of our method for each example is then given.

Bucharest Metro Map

The Bucharest metro system [16] features 44 stations and a route length of 93km. The system is split into four lines. The most notable feature of the map is the central loop with the other lines radiating from the loop. The map provides a good example of a small metro system. Figure 4 shows the initial layout of the map (which in this case is the geographical layout of the system) on the left and the final layout of the map after running our method on the right.

Figure 4. Bucharest metro map. Initial geographic layout (left) and the final layout after running our method (right).

The final map shows significant improvement over the initial layout. Every edge is now drawn using a multiple of 45° and lines have been drawn much straighter. This is particularly evident with the black line (vertical, through the center of the map) which is particularly strong. The central loop also has a distinct shape making it quite pronounced.

Figure 5. Example of problematic labeling in the Bucharest metro map.

Nodes have generally been labelled quite successfully. However there is one place in particular where labelling could be improved. The yellow line (horizontal, to the bottom-left of the central loop) is an example of where the labels have become ambiguous and also where labels have clashed (shown in Figure 5). The problem here was caused by the label for the Armata Poporului station being particularly long – there was no possible position for it which avoids clashing with one of the other labels or an edge without first moving the label for either the Gorjului or Politehnica stations. The current position was chosen as the best compromise. This problem could possibly have been solved by allowing labels to span more than one line. It might also have been possible to counter this problem by allowing labels and nodes to move at the same time, rather than as two separate steps. Doing it this way would possibly have forced the nodes to be spaced further apart thereby allowing more room for the labels. A third way of solving this problem could be to use labels at different orientations other than horizontal. Using diagonal labels (on the same angle as diagonal edges) would allow horizontal lines to be labelled such that there can be a shorter space between nodes.

Washington Metro Map

The Washington Metro map [24] is more complex than the Bucharest map but is still only a moderately sized map. The map features 84 stations and a total route length of 164km split into five lines. Figure 6 shows the initial layout of the map (which in this case is a sketch of the layout of the system) on the left and the final layout of the map after running our method on the right.

It is clear that the final map shows substantial improvement over the initial map. As with the Bucharest map, all the edges are now drawn orthogonally or diagonally and lines have been straightened out somewhat. There also appears to be more space in the centre of the map where it is most complex, allowing greater clarity in this area. However, there are several features of the map which make it particularly challenging to draw.
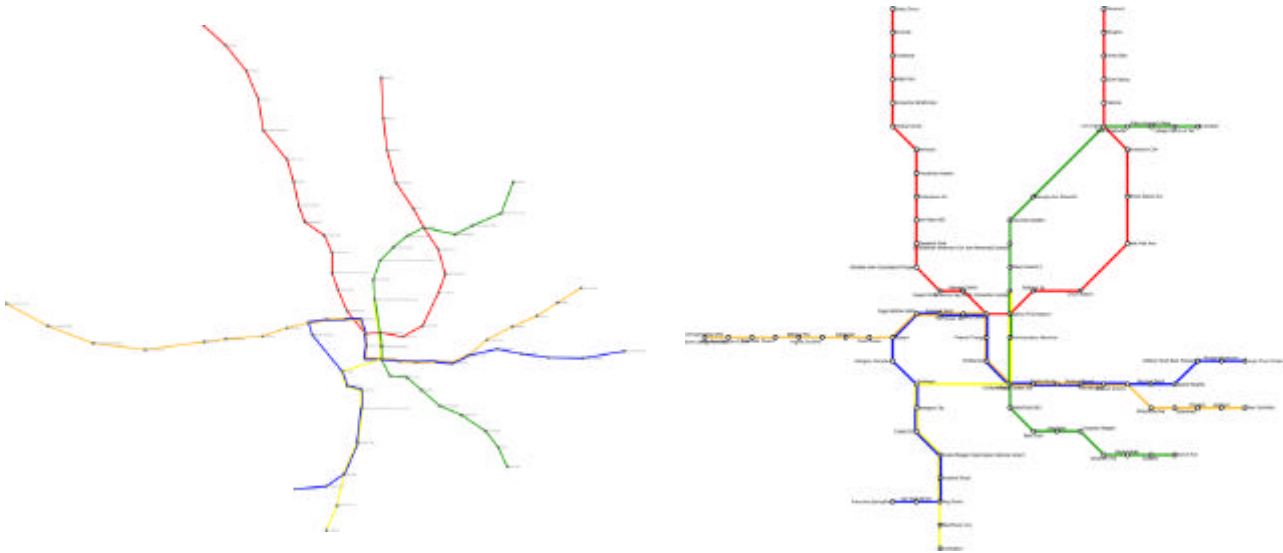
Figure 6. Washington Metro map. Initial geographic layout (left) and the final layout after running our method (right).

Figure 7. An example of a local minimum in the Washington Metro map. Neither the Capitol Heights or the Addison Road stations can move without first moving the labels.

Figure 8. An example of an ambiguously positioned label. The label for Deanwood might apply to either of the two stations.

Most of the problems arise from the presence of local minima. In a number of places, nodes are unable to move because an occlusion with a label will happen. Figure 7 shows such a local minimum. In this case, the map could be improved by moving the Addison Road node down to be in line with the Benning Road and Capitol Heights nodes. However, in doing this, the Addison Road label will occlude two nodes and two edges and the Capitol Heights label will occlude two edges and one node. This problem could possibly be solved if label movements are considered in conjunction with node movements, or even by allowing some node movements that do not take label occlusions into account (the labels would be repositioned in better positions later).

Another feature of the Washington Metro map is the large number of long station names. This makes the labels particularly lengthy. As we do not currently have a way to split long labels over more than one line, this can cause problems with the label occluding other labels, nodes and edges, especially on horizontal lines but also occasionally on vertical lines. Another problem with labels is that of ambiguously positioned labels. These are labels which have a length roughly equal to g and have been positioned in such a way as to fill the space between two nearby nodes. This is illustrated by the Deanwood node in Figure 8 – at first glance, the label could apply to either the left-hand or the right-hand node and it is only the presence of the label for the right-hand node that disambiguates the labelling. A potential solution to this would be to add a new labelling criterion that causes labels to be positioned as far as possible from other neighbouring nodes or as close as possible to the node that the label applies to.

Atlanta MARTA Rail Map

The Metropolitan Atlanta Rapid Transit Authority (MARTA) rail map [17] is an example of a small map with only 38 stations and 2 lines (the North-South and East-West lines) with a total route length of 74km. Figure 9 shows the final layout of the map after running our method.

Figure 9. Metropolitan Atlanta Rapid Transit Authority (MARTA) rail map after running our method.

Although the map is apparently straightforward, our method fails to find an acceptable layout. All of the lines have unnecessary bends, particularly the North-South Line between the Civic Center and Dunwoody stations. In the case of the right-angle bend at Civic Center, this was caused by a local minima whereby the labels for neighbouring stations obstructing the upwards movement of North Avenue. This could possibly be solved if labels could be repositioned while nodes are being moved. Also, many of the labels are positioned ambiguously. This is especially apparent on the East-West Line between the Five Points and East Lake stations and where two labels occlude each other to the left of the Five Points station. It is worth noting that changing the weightings of each of the criteria will produce different finished maps. It is likely that there are combinations of weightings that will produce better maps than the one shown in Figure 9, but finding the optimal criteria is a time-consuming process.

4. CONCLUSION

In this paper we have shown a method for automatically laying out metro maps using a generalized, schematic map. We have demonstrated our method on a number of real-world examples and discussed the strengths and weaknesses of the method. Our future work will look at refining the current technique, and developing new techniques that take a new perspective on metro map layout.

There are new criteria that might be introduced to improve the layout of the map. Symmetry is a notoriously hard feature to measure in graphs, but in some simple cases it could be measured effectively, for example where lines branch. More criteria might be designed for specific types of map. For instance, where geographic features such as rivers, road or coastlines appear, movement of nodes relative to the feature might be penalised. Also, in specific cases it might be desirable to ensure metro lines are in a certain orientation, or that two lines run in parallel.

Labelling is a major issue in producing metro maps, and several improvements to our current labelling strategy might be implemented. Firstly, all of the potential label positions might be tested on each node movement, rather than after the

node has been moved. Secondly, the orientation of the labels might be changed. Often in metro maps, labels have a diagonal orientation. This means that it is not simply a case of testing new orientations for labels, because the orientation of labels on a line segment is usually consistent. The orientation of a group of labels would then have moved at once. Criteria might also be introduced to ensure that labels along a line appear on the same side of the line rather than alternating between either side.

Our further work goes beyond refining our current technique. Our current method has an intrinsic problem in the time taken to generate a layout and we shall be looking at alternative layout methods that may improve the performance of the method. Finding optimal criteria weightings can also take a significant amount of time. In particular, orthogonal graph layout algorithms are typically quick to execute and orthogonal graph drawing has a close relationship to the desired layout of metro maps. It may be possible to integrate some of the design features of metro maps into orthogonal layout algorithms without significantly affecting performance.

REFERENCES

[1]   M. Agrawala, C. Stolte. Rendering Effective Route Maps: Improving Usability Through Generalization. *Proceedings of SIGGRAPH 2001*. pp. 241-249. 2001.

[2]   S. Avelar, M. Müller. Generating Topologically Correct Schematic Maps. In *Proceedings 9th International Symposium on Spatial Data Handling*. pp. 4a.28-4a.35. 2000.

[3]   T. Barkowsky, L. J. Latecki, K. Richter. Schematizing Maps: Simplification of Geographic Shapes by Discrete Curve Elimination. *Spatial Cognition II*, LNCS 1849. pp. 41-53. 2001.

[4]   G. di Battista, P. Eades, R. Tamassia, I. G. Tollis. *Graph Drawing: Algorithms for the Visualisation of Graphs*. Prentice Hall. 1999.

[5]   R. Burkhard, M. Meier. Tube Map: Evaluation of a Visual Metaphor for Interfunctional Communication of Complex Projects. *Proceedings of I-KNOW 2004*. pp. 449-456. 2004.

[6]   S. Cabello, M. de Berg, S. van Dijk, M. van Kreveld, T. Strijk. Schematization of Road Networks. *Proceedings of the 8th Annual ACM Symposium on Computational Geometry*. pp. 33-39. 2001.

[7]   S. Cabello, M. van Kreveld. Schematic networks: an algorithm and its implementation. In D. E. Richardson and P. van Oosterom, eds., *Advances in Spatial Data Handling*. pp. 475-486. Springer. 2002.

[8]   H. Casakin, T. Barkowsky, A. Klippel, C. Freska. Schematic Maps as Wayfinding Aids. *Spatial Cognition II*. LNCS 1849. pp. 54-71. 2001.

[9]   J. Christensen, J. Marks, S. Shieber. An Empirical Study of Algorithms for Point-Feature Label Placement. *ACM Transactions of Graphics*. Vol. 14, No. 3, pp. 203-232. 1995.

[10]   D. Elroi. Designing a network line-map schematization software enhancement package. In *Proceedings 8th Annual ESRI User Conference*. 1988.

[11]   K. Garland. *Mr. Beck's Underground Map*. Capital Transport Publishing. England. 1994.

[12]   S. H. Hong, D. Merrick, H. A. D. do Nascimento. The metro map layout problem, in N. Churcher and C. Churcher, eds., *Information Visualisation 2004*, Vol. 35 of Conferences in Research and Practice in Information Technology, ACS, pp. 91-100. 2004.

[13]   Java™. Sun Microsystems. http://java.sun.com/. Accessed on 29 March 2005.

[14]   K. G. Kakoulis, I. G. Tollis. A Unified Approach to Labelling Graphical Features. *14th Annual ACM Symposium in Computation Geometry*. June 1998.

[15]   M. van Kreveld, T. Stijk, A. Wolff. Point Labelling with Sliding Labels. *Proceedings of the 14th Annual ACM Symposium of Computational Geometry*. pp. 337-346. 1998.

[16]   Metrorex (Bucharest Metro) Map. http://www.metrorex.ro/. Accessed on 15 April 2005.

[17]   Metropolitan Atlanta Rapid Transit Authority (MARTA) network map. Available at http://www.itsmarta.com/getthere/schedules/index-rail.htm. Accessed on 25 April 2005.

[18]   K. V. Nesbitt. Getting to More Abstract Places with the Metro Map Metaphor. *Proceedings of the 8th International Conference on Information Visualisation (IV04)*. IEEE, July 2004.

[19]   M. Ovenden, *Metro Maps of the World*, Capital Transport Publishing, England, 2003.

[20]   H. C. Purchase, R. F. Cohen, M. Jones. Validating Graph Drawing Aesthetics. *Proceedings of the Symposium on Graph Drawing*. Springer LNCS 1027. pp. 435-446. 1995.

[21]   E. S. Sandvad, K. Grønbæk, L. Sloth, J. L. and Knudsen. A Metro Map Metaphor for Guided Tours on the Web: the Webvise Guided Tour System. *Proceedings of the 10th International World Wide Web Conference*, Hong Kong, May 1-5, 2001. ACM: New York, 2001. pp. 326-333.

[22]   J. M. Stott, P. Rodgers. Metro Map Layout Using Multicriteria Optimization. *Proceedings of the 8th International Conference on Information Visualisation (IV04)*. IEEE, July 2004.

[23]   Transport for London website. London Underground map. http://www.tfl.gov.uk/tube/maps/. Accessed on 22 April 2005.

[24]   Washington Metro Map. http://www.wmata.com/metrorail/systemmap.cfm. Accessed on 5 March 2004.