HOLA: Human-like Orthogonal Network Layout

Steve Kieffer, Tim Dwyer, Kim Marriott, and Michael Wybrow

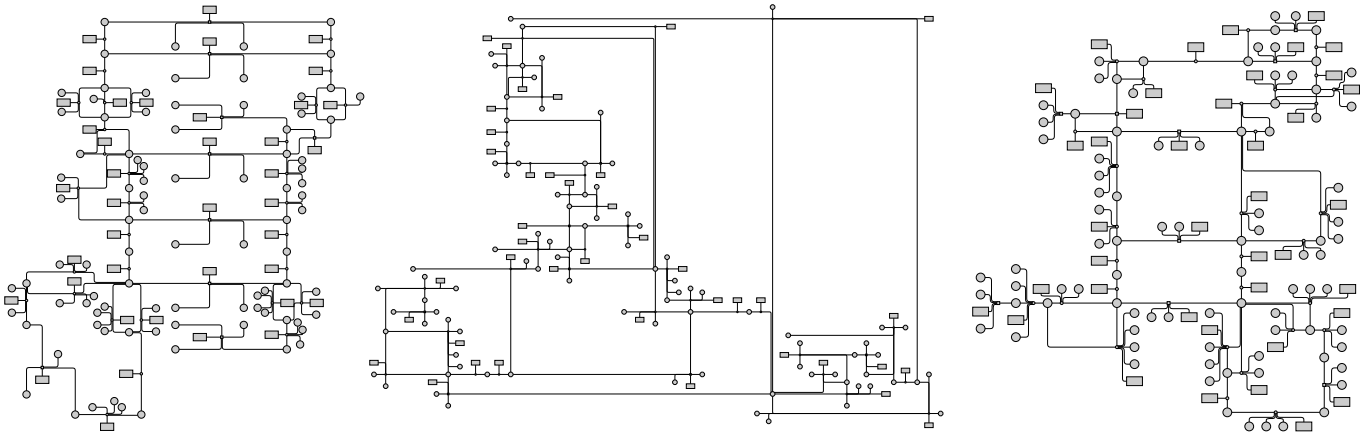


Fig. 1: Human, yFiles, and HOLA layouts of SBGN Glycolysis-Glygoneogensis pathway. It is clear that the human and the state-of-the-art layout algorithm from yFiles produce structurally quite different layouts. In this paper we explore the reasons why humans arrange such orthogonal network diagrams differently to standard algorithms and use this to inform the design of a new algorithm, HOLA (output shown right-most), that aims to produce more "human-like" layout.

Abstract— Over the last 50 years a wide variety of automatic network layout algorithms have been developed. Some are fast heuristic techniques suitable for networks with hundreds of thousands of nodes while others are multi-stage frameworks for higher-quality layout of smaller networks. However, despite decades of research currently no algorithm produces layout of comparable quality to that of a human. We give a new "human-centred" methodology for automatic network layout algorithm design that is intended to overcome this deficiency. User studies are first used to identify the aesthetic criteria algorithms should encode, then an algorithm is developed that is informed by these criteria and finally, a follow-up study evaluates the algorithm output. We have used this new methodology to develop an automatic orthogonal network layout method, HOLA, that achieves measurably better (by user study) layout than the best available orthogonal layout algorithm and which produces layouts of comparable quality to those produced by hand.

Index Terms—Graph layout, orthogonal layout, automatic layout algorithms, user-generated layout, graph-drawing aesthetics

1 Introduction

Visualisation of connectivity is essential to understanding relationships, dependencies and failures in complex systems such as those occurring in power-grids, software, biological pathways, brain-connections or financial markets. While direct visualisation of adjacency matrices (and exotic variations such as [9]) offer a non-diagrammatic way to represent network structures, the most widely-used and intuitive way to depict connectivity is through node-link diagrams.

Automatic layout of node-link diagrams has been studied since the 1960s. Force-directed approaches date from 1965 [14], layered-layout for directed graphs from 1981 [33] and orthogonal layout from 1986 [7] in which connectors consist of horizontal and vertical segments. The designers of these early algorithms relied on intuition and well-known gestalt principles to infer aesthetic and readability criteria. These early algorithm designs were also influenced by the availability of efficient heuristics for well-understood problems in graph theory (such as identification of a planar subgraph). For example, the first step in Topology-Shape-Metrics (TSM) approach from 1986 by Batini *et al.* [7] is to attempt to embed a planar subgraph without crossings. TSM remains the basis for most orthogonal layout methods in use today.

Despite these decades of research into network layout it is fair to say that currently no automatic layout algorithm produces layout of comparable quality to that which a human can produce with careful manual adjustment. This is why manual layout continues to be used for curation of metabolic pathways in biological repositories such as KEGG. As an example Figure 1 compares layouts produced using the state-of-the-art orthogonal layout algorithms in the yFiles graph layout library [4] with those produced manually. It is clear that the layouts are very different, and, as our studies described here show, humans overwhelmingly prefer orthogonal layouts produced by hand to those produced using traditional orthogonal layout algorithms. Here we address the problem of how we can develop automatic network layout algorithms that produce layouts that truly are of comparable quality to those produced manually.

We believe a necessary requirement for building such layout algorithms is an understanding of what humans value and like in network layout. One possible reason for the current discrepancy between manual and automatic layout is that formal user studies of requirements for good layout only occurred *after* the initial development of automatic layout methods. The earliest user studies that we know of were not until the mid 1990's. These used carefully designed diagrams to investigate whether the aesthetic criteria intuitively identified by the early algorithm designers did in fact assist with comprehension, e.g.

• *Steve Kieffer, Tim Dwyer, Kim Marriott, and Michael Wybrow are with Monash University. E-mail: {steve.kieffer, tim.dwyer, kim.marriott, michael.wybrow}@monash.edu.*

Manuscript received 31 Mar. 2015; accepted 1 Aug. 2015; date of publication xx Aug. 2015; date of current version 25 Oct. 2015. For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

[29, 23]. At the time this was taken very much as an affirmation of the existing algorithms since these studies largely confirmed that the aesthetic criteria identified by the algorithm designers, such as reducing crossings and bend points, did affect readability.

It is not until quite recently that studies such as [34, 12, 30] returned to first-principles in understanding the requirements for network layout by investigating the kind of node-link diagrams that humans construct by hand. Such "human authored layout" studies have identified aesthetic factors not considered by early algorithm designers or in the previous usability experiments. Unfortunately, this work is rather late to the party and—until now—has not influenced the design of automatic layout algorithms. This is not really surprising, as creating new algorithms or re-engineering existing algorithms to capture new aesthetics is difficult and time consuming. Furthermore, algorithms research is typically the domain of researchers who have little interest in human studies.

Here we bridge the gap between user studies and algorithm design and explore how such user studies can be used as "formative" input into the design of network layout algorithms. Our contributions are three-fold:

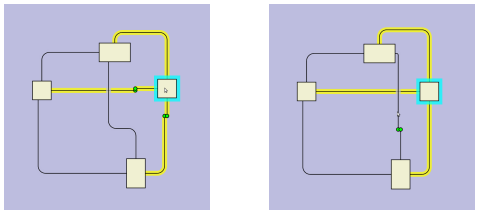
1) *A new "human-centred" methodology for automatic network layout algorithm design.* This methodology utilises user studies for both formative and summative feedback. It has three steps: 1) conduct user studies to determine the aesthetic criteria that humans value and the tradeoffs between them; 2) develop an algorithm that encodes these aesthetics; and 3) evaluate the algorithm in user studies by comparing it to both manual layout and the best existing automatic layout algorithms. Ours is the first work we are aware of that "closes-the-loop" of designing a network layout algorithm based on human preferences in this way.

2) *A measurably better automatic orthogonal network layout algorithm, HOLA (Human-like Orthogonal Layout Algorithm).* We have used the human-centred methodology to develop an orthogonal layout algorithm that is more effective in terms of both readability and aesthetics than the best available orthogonal layout algorithm. Our new approach is made possible by the flexibility of the *constraint optimisation* techniques described in [11] and the optimisation approach to edge-routing as described in [36]. Our user study also shows that HOLA produces layouts of comparable quality to those produced by hand: this is the first automatic layout algorithm that we know of that can claim this level of quality.

3) *The first formative user study designed to explore the aesthetic criteria valued by humans in orthogonal edge routing*. This is the first "human-authored layout" study in which the participants were not only able to move nodes but were also required to manipulate orthogonal edge routes. This permitted the interesting discovery that in manual layout people often create "aesthetic bend points" deliberately, and this became a special feature of our layout algorithm.

While an algorithm designed by our methodology can be expected to produce layouts that users *want*, it is fair to ask whether these are also the layouts that users *need*, i.e. which best facilitate their tasks. Our summative study has revealed however that users not only preferred HOLA layout aesthetically but also performed better on HOLA layout with tasks like finding a shortest path between two nodes. This raises the question whether the manual layouts that HOLA is designed to mimic are consciously made "usable" by their human designers, in addition to aesthetically pleasing, but this question is beyond the scope of this paper.

The reason that we have focused on orthogonal layout is its practical importance. Orthogonal diagramming conventions are widely used and sometimes even mandated by standards for electrical engineering, software engineering, process flow diagrams and other software engineering notations. Standards such as SBGN and KGML in the life sciences also use orthogonal conventions. Because of the poor performance of current layout algorithms, manual layout is widely used for the layout of diagrams of up to a few hundred nodes. These diagrams also typically have low edge density: cloning and other techniques are used to reduce the number of edges so as to improve readability. Thus, there is a clear need and motivation for better techniques for these kinds of diagrams that, in our opinion, has been largely overlooked in recent years by information visualisation researchers in the rush to draw larger and larger networks.



(a) Bend point "snap-to" when dragging a node

(b) Dragging an edge

Fig. 2: The Orthowontist online editor. We considered existing editors yEd and MS Visio, but found the controls for editing orthogonal connectors to be overly complex, so devised the simple interface employed in Orthowontist.

2 Human Studies Considering Network Layout

The first network layout user studies investigated whether the intuitive aesthetic criteria used to motivate the design of graph drawing algorithms did in fact affect human understanding of graphs. Early studies by Purchase and her colleagues [28, 24, 29, 27] investigated how reducing edge crossings, reducing bends, showing subgraph symmetry, increasing angle of incidence of edges entering/leaving a node, and orthogonality (which was taken to mean node placement on a grid) affected user performance on tasks like finding the shortest path between two nodes in abstract graphs. Another study by Purchase *et al.* [26] investigated the effect of these aesthetics on user preference in UML diagrams while Huang *et al.* [19] investigated their effect on both preference and performance for social networks. These studies reported strong negative effects on task performance and user preference for edge crossings and to a lesser extent edge bends and found that symmetry and orthogonality were preferred. Huang *et al.* found a preference for important nodes to be placed at the top of the drawing. Subsequent studies by Ware *et al.* [35] and an eye-tracking study by Huang [18] have suggested that the negative effect of edge crossings is reduced if the edges cross at large angles and that continuity of edges through nodes is important in path following. A recent study by Marriott *et al.* [22] examined the effect of these aesthetics on recall. Starting with Himsolt [16] some user studies have used preferences and task performance to compare different automatic layout algorithms and layout styles [25, 27, 19]. These findings have been inconclusive.

We believe that a new kind of user study pioneered by van Ham and Rogowitz [34] can provide particularly important formative input for network layout algorithm design. In such studies participants are asked to manually draw or edit graphs. Their drawings may reveal the existence of aesthetic criteria not previously considered and also provide insight into how people trade off the different competing criteria.

These studies of human-composed network layout have reported a number of findings about human preferences for layout that have not yet been incorporated into algorithms. In particular, van Ham and Rogowitz found that people like to arrange "clusters" in graphs such that the edges form a convex boundary. Dwyer *et al.* [12] found that layouts with low stress were strongly preferred over layouts with large variance in edge lengths. The same study also found that users strongly preferred force-directed or user-generated layouts that resemble force-directed layout over an orthogonal diagram produced by a standard algorithm (yFiles).

Most recently Purchase *et al.* [30] asked participants to draw a graph specified by an adjacency matrix using a graph drawing sketch tool. They found that edge crossings were avoided and that grid-like layouts were preferred. They also found that clusters were emphasised and that edge lengths were relatively uniform.

3 Manual Layout of Orthogonal Networks

One of the main contributions of this paper is to explore how user studies can inform the development of better network layout algorithms. Our human-centred design methodology is based upon using user studies as formative input into the design of the layout algorithm from the very start. We believe that studies in which users are free to layout the network in any way they like are particularly useful in being free of any bias imposed by an existing automatic technique. However, while the few previous such studies [34, 12, 30] are certainly relevant to understanding what humans like in orthogonal network layout, none considered orthogonal edges. We therefore conducted a human-composed network layout study specifically designed to identify the factors that humans regard as important for good orthogonal layout.

Like [12, 30] our study had two stages. In the first stage participants were asked to manually layout some small graphs. In the second stage (different) participants were asked to rank the manual layouts. This allowed us to identify the manual layouts that really were regarded as being of high quality. We also included the original layout and an automatically generated layout in this ranking.

3.1 Stage A

Apparatus & Materials: In the first stage participants were asked to manually improve the layout of 8 small graphs. The initial graphs are shown in the leftmost column of Fig. 3 while the other columns show some of the manual improvements. The graph stimuli had between 4 and 13 nodes and the initial layout was quite messy with many bends and crossings. Small graphs were used because it was unrealistic to require participants to spend the large amount of time needed to manually layout graphs with more than this number of nodes.

To edit the layout participants used an easy-to-use web-based interactive layout tool explicitly designed for manually editing orthogonal graph layouts. The tool, Orthowontist, was programmed in HTML5/JavaScript. It allowed the user to move nodes, to add, delete or move bend points in the orthogonal connector routes, and to change the position in which a connector enters the node. The tool logged editing actions and their time as well as recording the final layout, Fig. 2.

Participants: The study was advertised on *Monash Memo*, a university-wide bulletin. Seventeen participants undertook the study and all completed it. Three $50 gift cards were offered as incentive, and participants were instructed that they would win one of these if their layouts were ranked in the top three in Stage B of the study.

Procedure: Participants completed the study online, with the graph editing tasks taking an average 15 min 6 sec in all. This stage of the study had four components:

1) After reading an explanatory text and signing the consent form participants completed a short questionnaire to ascertain their prior experience with node-link diagrams. Their identities are protected, but they were also asked if they wished to leave contact details in case they won one of the gift cards.

2) The participants completed training in the use of Orthowontist.

3) In the main part of the study the participants were presented in turn with the 8 graphs arranged randomly. Participants were asked to edit the diagram until they felt that it "looked good" and clearly conveyed the connections between the nodes. They were asked to imagine that the diagram would be used to convey information and should be clear and readable. Participants were instructed that the experiment was not timed, and they could take as long as they liked. Once they were satisfied with the layout they moved to the next graph. The order of the graphs was randomised.

4) Finally the participants were asked to write down what their goals were when improving the layout of the networks.

3.2 Stage B

Apparatus & Materials: In the second stage participants were asked to choose the best layout obtained from Stage A for each of the 8 sample graphs. In addition to the seventeen human-made layouts of each graph, we also included the original messy layout, as well as the layout computed for that graph by the *classic orthogonal layout* command in the yEd diagram editing software (version 3.9.2) [4] with default settings (yFiles layout)[1]. There were thus eight graphs $g_1, g_2, \ldots, g_8$ with nineteen layouts each. We denote by $L_j^{(i)}$ layout j of graph g_i, where $i \in \{1,2,\ldots,8\}$, and $j \in \{0,1,\ldots,18\}$, with $j=0$ meaning the original messy layout, $j=18$ meaning the yFiles layout, and $j \in \{1,2,\ldots,17\}$ meaning the layouts created by the seventeen human respondents of the first stage.

Because of the large number of graphs to compare we used a tournament structure to identify the best layout. This meant that participants were only required to vote for the best layout out of the three presented to them in each match of the tournament. We wrote a web-based tournament tool so that the study could be conducted online.

Participants: The study was advertised on *Monash Memo*. It was completed by 66 participants. One participant was excluded because they consistently chose the third layout. A $50 gift card was offered as incentive, and it was explained that the winner would be the participant whose choices were the closest to the aggregate choices, thus that in order to win your best strategy was to choose the layouts which you thought other participants would also choose.

Procedure: The survey was conducted online. Upon loading, the tournament software organised the human-made and yFiles layouts for each graph g_i into a tournament structure with random seeding; i.e. the eighteen layouts of positive index $L_1^{(i)}, L_2^{(i)}, \ldots, L_{18}^{(i)}$ were shuffled into a random order to make the seeding of the tournament. For each *match* of the tournament the participant was presented with three layouts and asked to choose the best one. The tournament thus fell into three *rounds*, with six matches in the first, and two matches in the second, at which point the two best layouts among the eighteen of positive index had been chosen. For the final round these two layouts were pitted against the original messy layout $L_0^{(i)}$. Participants spent an average of 6.57s per choice (discounting outliers of one minute or longer), and the entire survey took an average of 7 min 53 sec to complete.

3.3 Results & Discussion

In response to the familiarity questionnaire of Stage A, 100% of participants said they were familiar with node-link diagrams, 94% said they had used a node-link diagram created by someone else, and 59% said they had created a node-link diagram in order to convey information to others.

Excluding outlying inter-drag pauses of a minute or longer, the average time spent editing each graph ranged from 1 min 4 sec (Graph 1) to 3 min 22 sec (Graph 5), with an overall average of 1 min 53 sec per graph.

Based on the tournament results in Stage B we computed the *mean rank* of each of the 19 layouts for each of the 8 stimuli/input graphs. Equal ranks were averaged so that the winner of a tournament received rank 1, layouts that lost in the final round rank 2.5 each, layouts losing in the second round rank 6.5 each, and layouts that lost in the first round received rank 14.5. These ranks were averaged over all participants to compute the mean rank μ for each layout of each graph. This was then adjusted to a *normalised inverted mean rank* $\bar{\mu} = 1-(\mu-1)/13.5$ ranging from 0 to 1, with 1 being the best possible score, and 0 the worst. Fig. 3 shows the original layout, the two best and two worst manual layouts for each of the input graphs, as well as the layout produced with yFiles.

Our analysis identified nine significant results (**R1-9**)[2]. The first two findings are novel:

[1] The yFiles orthogonal layout algorithm has many options. After extensive experimentation we came to the conclusion that the developers have already tuned the default settings to give best all-around results. For this reason, and also for reproducability, we determined to use default settings across all graphs used in our studies.

[2] Full results available online [2]

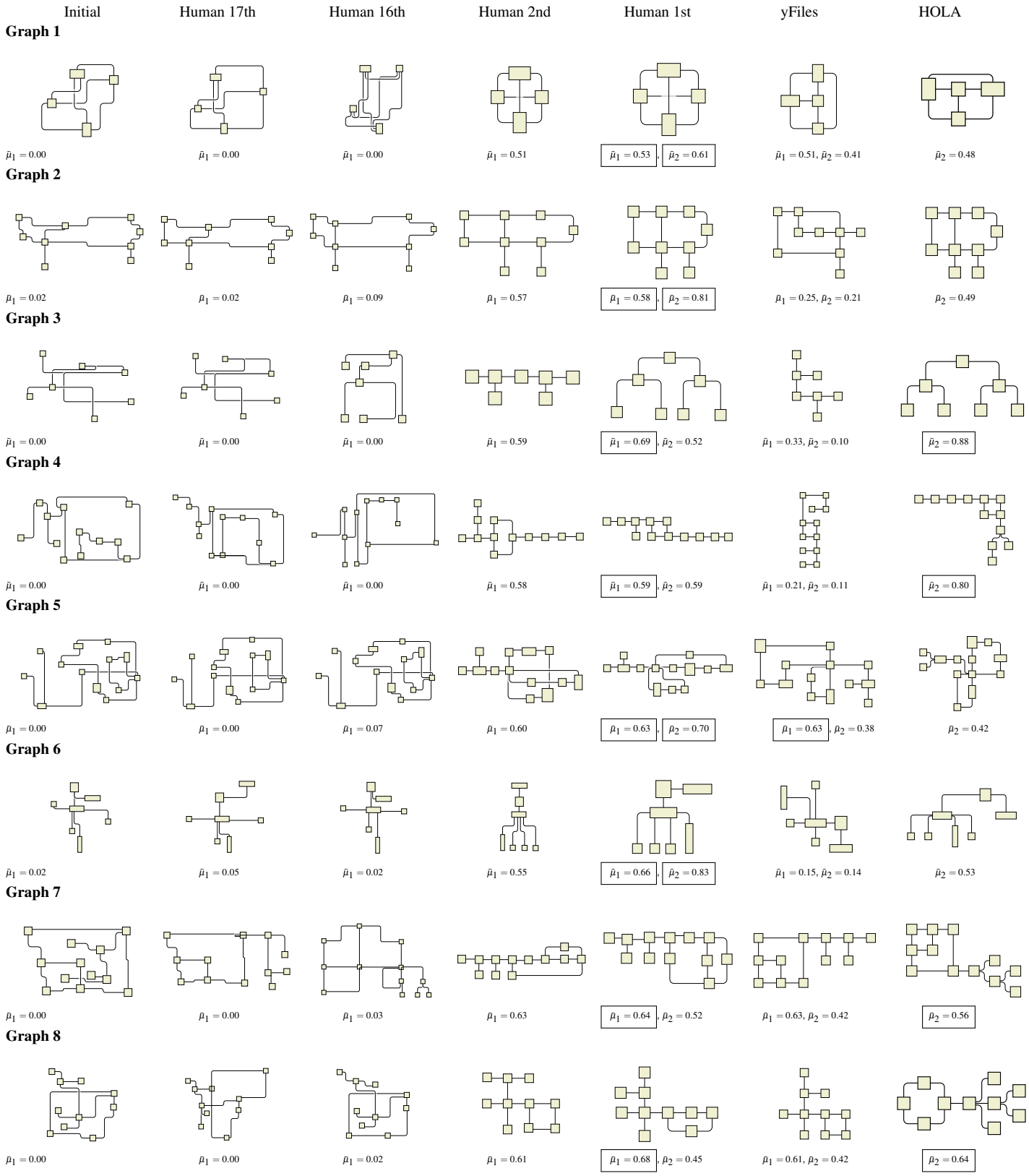
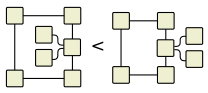


Fig. 3: The 8 graphs and some of their layouts from the manual layout of orthogonal graphs study. At left is the initial layout. The next 4 columns show the two worst and the two best manual layout. The two final columns show automatic layout from yFiles and our proposed algorithm HOLA. $\bar{\mu}_1$ = normalised inverted mean rank for first study, $\bar{\mu}_2$ = that for second study. Best possible value is 1, worst possible 0. Means in boxes indicate best actual mean rank for that contest.

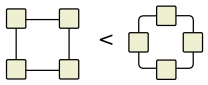
Table 1: Pearson's correlation coefficient between normalised inverted mean rank $\bar{\mu}$ and various indicators of the quality of a layout. Positive correlations indicate a feature which makes a better layout, negative correlations a worse layout. Single star * means significance at $p = 0.05$ level. Double star ** means significance at $p = 0.01$ level. A '–' indicates a feature not applicable to that graph.

	# Crossings	# Bend points	Seg Length Std Dev	Stress	Compactness	Gridiness	Symmetry	# Outer trees
Graph 1	**-0.427***	**-0.705****	-0.340	-0.059	**0.734****	**0.517***	**0.644****	–
Graph 2	-0.253	**-0.630****	**-0.411***	**-0.515***	**0.725****	**0.658****	**0.724****	0.148
Graph 3	**-0.436***	**-0.508***	**-0.798****	**-0.689****	**0.687****	**0.623****	**0.767****	–
Graph 4	–	**-0.666****	**-0.813****	**-0.859****	**0.695****	**0.883****	**0.752****	**0.740****
Graph 5	**-0.543****	**-0.569****	-0.306	-0.219	**0.870****	**0.805****	**0.454***	-0.055
Graph 6	-0.308	-0.048	**-0.599****	0.138	**0.698****	**0.698****	0.301	–
Graph 7	**-0.586****	**-0.626****	**-0.693****	-0.337	**0.703****	**0.712****	**0.600****	**0.491***
Graph 8	**-0.660****	**-0.733****	**-0.863****	**-0.662****	**0.759****	**0.754****	**0.743****	**0.749****

R1 users like trees placed outside i.e. the maximal acyclic components are placed outside the graph and not in inner faces. This is in line with users wishing to separate clusters [34, 30] and is supported by the positive correlations seen in the "# Outer trees" column in Table 1;



R2 users create "aesthetic bend points" we observed, in contrast to previous research, while bends overall were correlated with a poor ranking, nevertheless, certain bend points serving an obvious (to us) aesthetic purpose were present in most of the top-ranked human layouts. This is true for *all* except Graph 4 in the 'Human 1st' column of Fig. 3. In particular unnecessary bend points appear to have been introduced to emphasise symmetry or to ensure that if a node has two edges they are on opposite sides of the node, perhaps to ensure continuity in path following.



Furthermore we found correlations (Table 1) showing that user preference is positively correlated with:

R3 compactness (ratio of the area occupied by the nodes to the total area of the graph, the latter given by the bounding box of all nodes and bend points);

R4 grid-like node placement (percentage of nodes that participated in at least one alignment of three or more nodes);

R5 symmetry (largest set of nodes which pair off completely by reflection over a horizontal or vertical axis);

and negatively correlated with: **(R6) edge crossings**; **(R7) edge bends**; standard deviation of **(R8) edge segment length**; and **(R9) stress**. These accord with the conclusions of earlier studies [34, 12, 30].

In the answers to the post-task question about their aims when manually laying out the graphs, many participants mentioned untangling the graph and removing crossings, this seemed to be the most basic aim. Some mentioned symmetry, overall shape, balance, and trying to lay it out on a grid. Another mentioned "layout the less connect[ed] node[s] on [the] outside in a diagram."

Our experiment confirmed that manual layout leads to quite a different style of layout than that computed by the most widely used orthogonal layout algorithm. Furthermore humans significantly prefer the best human layout ($\bar{\mu} = 0.621$) to that produced by yFiles ($\bar{\mu} = 0.415$), as confirmed by a Wilcoxon signed rank test with $p = 1.286\text{e-}9$. We believe this is because current algorithms for orthogonal layout are designed to first minimise edge crossings, then bend points, and finally area, whereas humans are more flexible and will keep edge crossings and bends if this reveals symmetries, separates clusters, reduces segment length or improves compactness. They also like nodes to be organised on a grid.

4 Human-like Orthogonal Layout Algorithm (HOLA)

Given the widespread use of orthogonal networks there is a clear need for a new kind of layout algorithm, one that produces more human-like layout. Here we describe our new Human-like Orthogonal Layout Algorithm (HOLA). This has been carefully designed in accord with the aesthetics (R1-9) identified in the above user study. To give some feel for the effectiveness of our new algorithm we show the layout produced by HOLA for the 8 input graphs in the last column of Fig. 3.

1. Topological decomposition of graph into *trees* and *core* (Fig. 5a)
2. Layout of the core (5b,5c):
   (a) Stress-minimising layout of core (P1:R3,5,6,8,9)
   (b) Greedy orthogonalisation of layout (P2:R2,4,9)
   (c) Orthogonal edge routing (P2:R6,7,8)
3. Tree layout and placement (5d,5e):
   (a) Symmetric layout of each tree (P3:R5)
   (b) Planarisation of core
   (c) Insertion of trees into the core (P3:R1)
   (d) Stress-minimising layout (P1:R9)
4. Opportunistic improvement (5e,5f):
   (a) Opportunistic alignment (P2:R4)
   (b) Node distribution by neighbour stress (P2:R3,8)
   (c) Rotation (P2)
   (d) Remove dummy nodes and final orthogonal edge routing

Fig. 4: Steps in the HOLA Algorithm, with cross-reference to our design principles P1-3 and aesthetic goals R1-9

HOLA has 4 main steps as listed in Fig. 4 and illustrated in Fig. 5(a–f). Since symmetry (R5) is easy in tree layout (Step 3a), and users like trees on the outside (R1), we decided to decompose the graph into trees and *core* (Step 1), also reasoning that stress-minimisation could better reveal symmetries in the core once the trees had been removed. The flexibility of the constrained optimisation approach allowed us to then combine many and varied ideas in Steps 2, 3, and 4 to further address the aesthetic goals (R1-9), as examined in greater depth below.

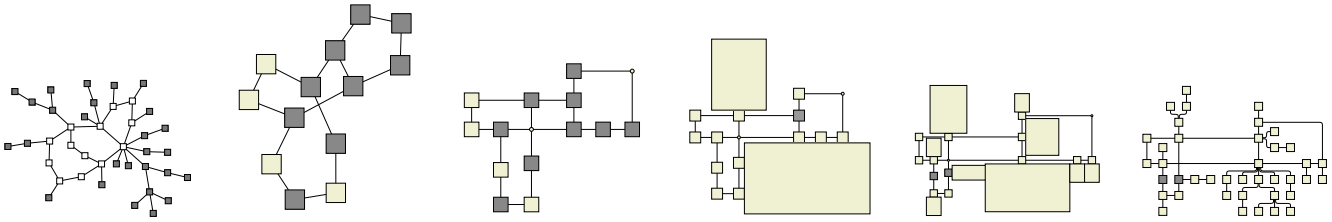
The algorithm is fundamentally different to previous orthogonal layout algorithms (i.e. those based on the Topology-Shape-Metrics (TSM) approach) [7]. TSM first fixes an embedding for the planarised input graph, then solves bend minimisation for this embedding to achieve an orthogonal shape, and in the final compaction step computes node positions. In contrast our approach is based on the following Principles:

P1 Use stress-minimisation (R9) to untangle the graph (R6) and reveal underlying symmetries (R5) as well as encouraging uniform edge length (R8) while keeping the layout compact (R3). Also, stress is always considered as one of the optimisation criteria throughout further modifications (R9).

P2 Apply incremental extensions/improvements to the existing layout, like an opportunistic human editor. In particular this is used to "tune" bend points (R2) and to achieve grid-like alignments where possible (R4).

P3 Subgraphs with tree structure are treated specially such that they can be arranged symmetrically (R5) and their placement can be controlled precisely (R1).

With respect to P1, this process is essentially opposite to that of Didimo *et al.* [8], in which a diagram is first given an orthogonal layout, and then that layout is made increasingly "force-directed". HOLA



(a) Step 1: Trees (dark grey nodes) will be pruned.

(b) Step 2: Stress-minimal layout of core. Dark grey nodes are the roots of trees.

(c) Step 2 (cont.): Core orthogonalised, edges routed, graph planarised.

(d) Step 3: Two trees placed, the tree rooted at the dark grey node is next. It could be placed NW, SW, or E.

(e) Step 3 (cont.) and 4: All trees placed. Note: the two dark grey nodes are almost but not quite aligned.

(f) Final layout. Nodes that were nearly aligned have been exactly aligned. Tree nodes have been reinserted.

Fig. 5: HOLA applied to a small example graph illustrating the four main steps.

instead gives primacy to low-stress layout by beginning with a (locally) stress-minimal layout then gradually making it orthogonal. The second point means that at virtually all stages of the pipeline there is an actual layout, i.e. positions for nodes and edges. This differs from TSM based approaches in which earlier stages in the pipeline only consider abstract topology. P2 reflects our desire to produce human-like layout and so we are employing a more human-like layout method.

The most closely related approach is that of Kieffer *et al.* [20] which explores the use of greedy heuristics to "gridify" a force-directed layout by snapping nodes to a grid or adding vertical or horizontal alignment constraints between nodes, and Rüegg *et al.* [31] which extends this to handle port constraints and directed edges laid out using orthogonal connectors. While stages 2b and 4a of HOLA utilise similar greedy heuristics, these are a small part of the overall approach. HOLA has been designed from the beginning to take into account aesthetics (**R1-9**) and, for instance, employs a completely different approach to layout of the graph's subtrees.

We now look at the steps in HOLA in more detail. This algorithm assumes a *connected* graph $G = (V,E)$, with node set V and edges E.

4.1 Topological decomposition

Step 1 is a *peeling* process [6, 5, 32] where all leaves are removed from the graph G repeatedly until none remain. The leaf nodes are added to a new graph H and reconnected to one another as they are added. When no leaves are left in G the remaining subgraph C is called the *core*. In Fig. 5a the dark grey nodes will be pruned by this process. Finally if L is the set of nodes in H, and $\rho : L \to V$ maps each pruned leaf node to the unique node to which it was attached at the time that it was pruned, then we form the set $R = \{\rho(\ell) : \ell \in L\} \setminus L$ of *root nodes*, and add to H a copy r' of each node $r \in R$, connecting it to each ℓ for which $\rho(\ell) = r$. The connected components of H then constitute some t trees $T_1, T_2, \ldots, T_t$, t a nonnegative integer equal to the size of the set R, and each tree T_i has root node $r'_i \in H$ which is a copy of a node $r_i \in C$. If we have $t = 1$ and $C = \emptyset$, i.e. the graph G is in fact a tree, then we simply apply our symmetric tree layout procedure (see below) and terminate. (See e.g. the HOLA layouts for Graphs 3 and 6 in Fig. 3.) Otherwise we proceed with Step 2, layout of the core graph.

4.2 Layout of the core

(a) Layout of the core graph C begins with a simple unconstrained stress-minimising layout [11], followed by the application of overlap removal constraints [13]. The first of these steps gives the nodes a natural and low-stress distribution in the plane (Fig. 5b). The user study of Dwyer *et al.* [12] has shown that users like low-stress layouts, and as we begin to orthogonalise the layout we will try to keep the stress low, in service of P1.

(b) Orthogonalisation proceeds in two steps, which we call *node configuration*, and *chain configuration*. In node configuration we sort all nodes v of degree 3 or higher by falling degree, and visit them in order. For each node v, we will align at most one of its neighbours in each of the four cardinal compass directions, NORTH, SOUTH, EAST, and WEST relative to v, and we call this a *configuration* of v. We have determined by experiment that configuring the highest degree nodes first tends to result in more favourable configurations, i.e. in a greater total number of alignments. Degree-2 nodes, meanwhile, which we call *links*, are left out of this process entirely. The maximal subgraphs consisting entirely of links are called *chains*, and these are configured in the following step, according to different principles.

We attempt to assign as many as possible of the neighbours of v to the four compass directions, favouring an assignment that minimises the total angular displacement of these nodes relative to v. The configuration is achieved by projecting onto appropriate alignment and separation constraints. Guided by P1, we prohibit any configuration that would reverse the orthogonal ordering of any node with respect to v. E.g. if u was a neighbour of v with $u_x < v_x$ before configuration, then we would require $u_x \leq v_x$ after configuration; in other words, u could not be assigned EAST. If in addition we had $u_y > v_y$, then u also could not be assigned NORTH. We also prohibit any assignment that would alter the cyclic order of the neighbours of v.

After greedily assigning the best configurations we can to each non-link node, we use gradient-descent to minimise the stress in the graph. During node configuration the stress will in general have climbed steadily with each projection onto the configuration constraints. The chains however remain unconstrained, and stress-minimisation will now cause them to settle into balanced, well-distributed arrangements, smoothing any jagged corners that may have arisen.

This permits us to again honour P1 as we move on to chain configuration, the second step in the orthogonalisation process. Let a chain be given, consisting of the links $v_1, v_2, \ldots, v_k$, and let nodes u and w be the other neighbours of the first and last links v_1 and v_k, respectively. Edges $e_0 = (u, v_1)$ and $e_k = (v_k, w)$ may or may not have been aligned by the node configuration process. For example, v_1 might be aligned EAST of u, and v_k NORTH of w. In such a case, choosing where to enforce bends in the chain is similar to the process of routing an orthogonal connector when the connection directions at each endpoint are given. As shown in Fig. 2 of Wybrow et. al. [36], the minimal number and types of bends required depend on the relative positions of u and w as well as the connection directions. If the edge e_0 is not yet aligned, we simply consider both of the possible compass assignments of v_1 relative to u that preserve their orthogonal ordering, and likewise with e_k.

For a given chain, there may be different minimal-length bend sequences, e.g. (R,R,L) or (L,L,L). We evaluate each potential bend sequence by a greedy process that chooses locally optimal points at which to create each bend in the chain.

Here we depart from the orthodoxy that bend points are always bad, as we consider both nodes and edges as potential bend points in the chain. We believe that a bend point may be deliberately created in an edge in service of other aesthetic principles, namely stress-minimisation and symmetry. This is one example of our attempt to follow P2, and work like an opportunistic human editor.

The basic idea of our greedy process is simple: in order to gauge

how suitable an edge is for becoming a bend point we measure how close its slope is to ± 1 in the plane. For a node we likewise consider the slope of the base of an isosceles triangle with apex at the node's centre and with one leg parallel to each of the node's edges.

The cost of creating a bend at a given edge or node is measured as a certain increasing function of the difference of the slope at that place from ± 1 (sign chosen according to bend direction), and the cost of a bend sequence is the sum of these costs. We greedily attempt to choose a minimal-cost bend sequence, and enforce it, aligning each edge of the chain vertically or horizontally with new alignment constraints, onto which we project. After thus processing each chain, the chain configuration step is complete.

As a configurable option, the ACA algorithm [20] may be applied to the chains instead of this process. We have found that this gives better results on large graphs as in our second user study (see Sec. 5), but not on small graphs as in the first study.

(c) For the low-density graphs at which our algorithm is targeted, most of the connectors will now be mere straight axis-aligned segments, as a result of the orthogonalisation. However, some diagonal connectors may remain, and in Step 2c we compute an orthogonal routing for each of these [36]. By construction every node in the core C has degree at least two, and we take care to ensure that connectors attach to at least two of the four sides of each node, lest the node become a leaf in the planarisation of Step 3b.

4.3 Tree layout and placement

In this step we now incrementally add the trees back into the core in accordance with P3.

(a) We first determine a layout for each tree T_i. The symmetric tree layout of Step 3a uses the algorithm of Manning and Atallah [21]. Each tree is provisionally given SOUTH growth direction, meaning that each rank is horizontally aligned and appears below, i.e. with greater y-coordinate, than the prior rank, starting with the root node r_i'. If the tree structure is in fact symmetric with respect to the root node then the layout will be symmetric about a vertical axis through the root node; otherwise it will be as close to symmetric as possible about this axis in a certain well-defined sense (namely the "c-trees" [21] are paired off about this axis to the extent possible). The edges of each tree are routed orthogonally [36] with all edge connections on the sides facing the opposite rank. Our initial user study suggests that this is a desirable layout for a tree as it clearly shows the structure and emphasises symmetries.

(b) Next we determine how to place each tree in the core. We begin by planarising the core in order to give it a well-defined set of faces into which the trees can be placed. This is achieved in two passes, each of which is implemented as a "sweep" algorithm [13]. In the first, edge overlaps are removed by introducing a dummy node for each bend point of each connector, and then eliminating all but one edge between each pair of nodes. In the second pass, edge crossings are removed by introducing a dummy node at each crossing. Crossing detection can be achieved by this $O(n \log n)$ sweep process because all connector segments are axis-aligned.

(c) We are now left with graph P, which is a planarisation of the core graph C, and whose set of nodes contains that of C as a subset (Fig. 5c). In particular each root node r_i for the trees T_i belongs to P. Since P is planar we may compute its set F of *faces*. Step 3c of the layout process now determines how to reattach the trees laid out in Step 3a to the root nodes r_i in P. This means choosing for each tree T_i a *tree placement* (f, d_p, d_g, b), where $f \in F$ is a face to which the root node r_i belongs, d_p and d_g are the *placement direction* and *growth direction*, and b is the *flip bit*.

The placement direction d_p is one of the eight compass directions including the four cardinal directions discussed already, as well as the four *ordinal* directions, SE, SW, NW, and NE. The growth direction d_g is one of the four cardinal compass directions. If d_p is cardinal then d_g must equal d_p; if d_p is ordinal then d_g must be one of the two cardinal components of d_p. The flip bit b is a Boolean saying whether the tree should be flipped over the axis of its growth direction d_g.

Tree placement is a greedy process. The trees are considered in descending order of the perimeter of their bounding box, and as we examine each possible placement (f, d_p, d_g, b) we attempt to make a choice which will minimise the increase in stress. In general placement into face f may require the generation of separation constraints to expand the available space inside f to make room for the tree. The expansion can usually be performed in several different ways (there is at least the decision whether to operate in the x- or the y-dimension first), and we evaluate each option by computing the necessary separation constraints, projecting, measuring the change in the stress of the graph, and backtracking. In this way tree placement is guided by P1.

Besides consideration of stress, however, our choice of tree placements is governed by two configurable flags, one saying whether we will favour cardinal placement directions d_p, and one saying whether we will favour placement in the external face f_{ext}, i.e. the unique unbounded face. Under our default configuration both flags are set to true, with cardinal placement taking highest precedence, followed by external placement, and with stress minimisation being considered last.

Favouring of cardinal placement is motivated by the desire for symmetry, and to make the routing of connectors from the root node to the tree nodes of the first rank easier (i.e. so that the connectors are shorter overall, and so that there is more room in which to route them). Favouring of placement in the external face is motivated by findings from our initial user study. Thus tree placement is also guided by P2.

In Fig. 5d two trees have been placed already, and their bounding boxes inserted into the graph temporarily as place holders. The tree rooted at the dark grey node is to be placed next, and three placement directions d_p are possible: SW or E into interior faces, or NW into the exterior face. Under our default configuration the E placement will be chosen, since it is the only cardinal direction available.

(d) During the tree placement process the stress of the graph will in general climb as we project onto each new set of face-expansion constraints. Therefore after all trees have been placed (Fig. 5e) we perform gradient-projection (Step 3d) to dissipate the accumulated stress as much as possible.

4.4 Opportunistic improvement

One of the hallmarks of (good) human layouts is that they obey the NONO principle: "Nothing is Obviously Non-Optimal." The final stage in our algorithm is designed to tweak the layout, ideally leaving no obvious small changes that would improve the layout.

(a) We begin in Step 4a with a process which searches for nearby pairs of nodes which are almost (but not quite) aligned, and it applies constraints to align them. For example in Fig. 5e the dark grey nodes will be aligned, as in Fig. 5f.

(b) After addressing alignments we consider the even distribution of nodes. This time instead of making local refinements we rely on a global stress minimisation step (Step 4b) in which we modify the stress function to include only those terms corresponding to neighbouring nodes. We refer to this as *neighbour stress*, and this step tends to space sequences of adjacent nodes more evenly by ignoring the effects of long-range forces.

(c) Next, with a view to the aspect ratio of most existing display devices, we rotate the layout by ninety degrees if its width is less than its height. To make the rotation direction deterministic we prefer that a majority of trees wind up with SOUTH growth direction after the rotation, rather than NORTH. Since we must not rotate the nodes themselves (they may have labels), rotation means that the node positions (but not dimensions) as well as the constraints holding amongst them, are rotated, and after this we apply another gradient-projection with neighbour stress.

(d) Finally we remove dummy nodes in passing from the planarised graph back to the original, and perform a final orthogonal routing of the edges. In this step we ensure that edge routes pass through any bend points deliberately created in the chain configuration step, but otherwise take this as a final chance for edge routes to be optimised relative to node positions which in general will have changed since the previous routing.

5 Final Evaluation

We performed a second user study[3] to provide a summative evaluation of the algorithm and comparison with the state-of-the-art orthogonal layout algorithm. This study extended our first evaluation (Sec. 3.2) in three ways.

First, from our first study we had eight user-generated layouts that were most highly ranked in the tournament. We could now compare these with two styles of automatic layout: the new HOLA algorithm and the state-of-the-art yFiles TSM implementation. Since the best manual layouts had already been shown to be (on average) preferred to yFiles, we expected them to again by preferred by participants in a direct three-way comparison; however, since HOLA is designed to be "human-like" in its approach to layout, we hypothesised (**H1**) that its output would also be preferred on average over yFiles on the corpus of small graphs.

Second, the correlations between layout attributes and preference revealed by our first study (R1-9) were observed in small graphs, limited by the size of graph that could reasonably be manually arranged by our participants. Since both HOLA and yFiles are completely automated we could now directly compare the "human-like" layout approach with the existing layout algorithm on much larger graphs. We hypothesised (**H2**) that HOLA layouts would be preferred over yFiles layouts by most participants on larger graphs.

Third, these larger and more complex graphs make readability much more challenging, and so for these graphs we could conduct non-trivial readability tests. We examined user performance on two standard tasks: finding a shortest path between two nodes and finding the neighbours of a node. We hypothesised (**H3**) that participants would have greater speed and accuracy in completing these tasks with HOLA layouts compared to yFiles layouts.

Apparatus & Materials: The second user study used two graph corpora. The first contained three layouts of each of the eight graphs from the original study: layouts computed with HOLA, yFiles (default settings, as discussed previously) and the highest-voted human layout of each graph. The second corpus contained HOLA and yFiles layouts for six larger graphs. We chose one SBGN graph (the Glycolysis-Gluconeogenesis pathway), one metro map graph (Sydney), and we generated four random graphs, including one small (60 nodes, 65 edges), one large (120 nodes, 126 edges), and two medium-sized graphs of different densities, (90 nodes, 100 edges) and (90 nodes, 110 edges). See Fig. 6. We could not include human layouts for these larger graphs because manual layout would have taken prohibitively long.

Participants: The study was advertised on *Monash Memo* as well as within our Faculty. 89 participants started the survey and completed Part 1, 84 completed Part 2, and 83 continued through Parts 3 and 4. Due to a display error in the first part, we had to reject the first 13 participants' answers on three of the eight graphs. A $50 gift card was offered as incentive, and it was explained that the winner would be the participant whose accuracy and speed were the best in Parts 2 and 3, and whose choices were the closest to the aggregate choices in Parts 1 and 4 and thus, in order to win, your best strategy was to choose the layouts which you thought would be preferred by most other people.

Procedure: The survey was conducted online. It had 4 parts.

Part 1: Participants ranked the three layouts in the first corpus. The order of the graphs was randomised, as was the order of the three layouts on each page. Users were asked to drag gold, silver, and bronze medal icons onto the three layouts. (The icons also said '1st', '2nd', and '3rd'.)

Part 2: Participants were asked to find a shortest path between two nodes in the six larger graphs. The two nodes were chosen systematically: one was a highest-degree node (chosen at random) and the other was any node either four or five links away (again chosen randomly). Users were shown each graph in two layouts, HOLA and yFiles, with the two chosen nodes coloured red, and they were asked to click on the nodes of a shortest path between them, turning the intermediate nodes green. They were first given a training task of the same kind,

[3]Full results available online [3]

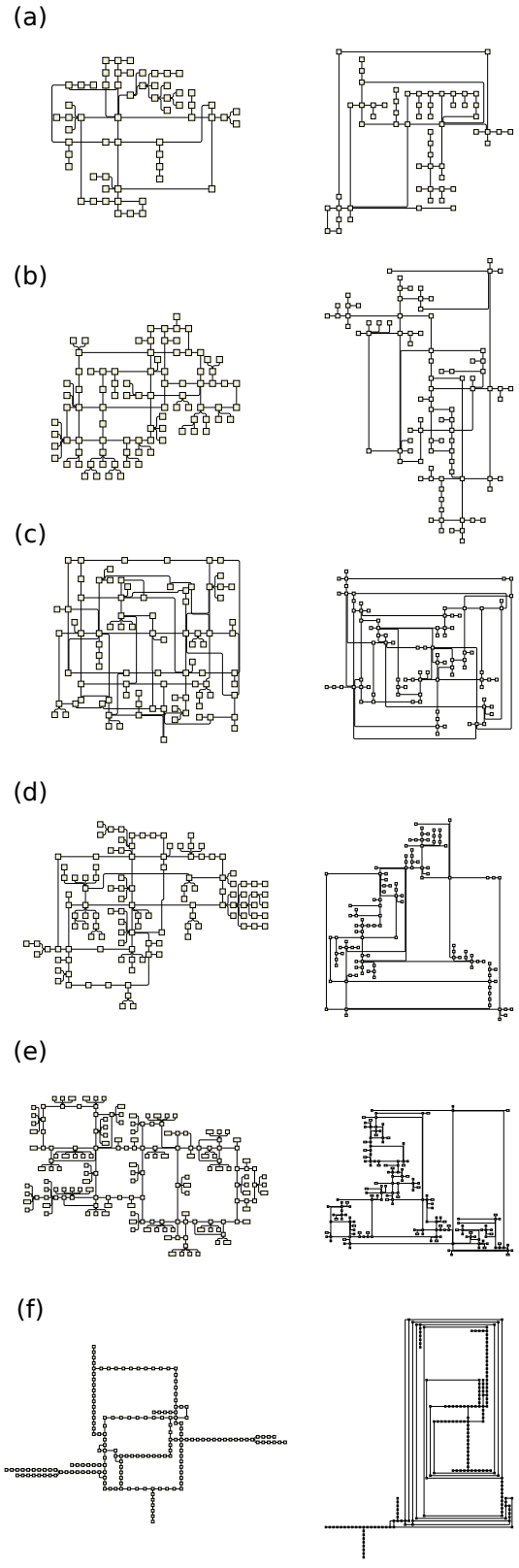


Fig. 6: Left column: HOLA layouts for the (a) small, (b) medium, lower density, (c) medium, higher density, (d) large, (e) SBGN, and (f) metro map graphs from Parts 2, 3, and 4 of the second study. Right column: yFiles layouts for the same graphs.

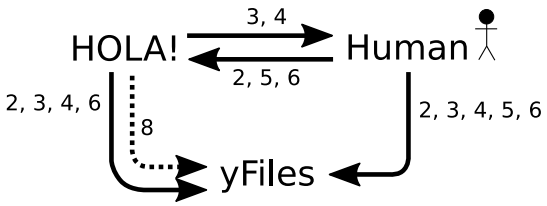


Fig. 7: Participants' preferences for drawings by human, yFiles or HOLA of graphs from Fig. 3. A solid arrow $a \rightarrow b$ indicates a significant preference for condition a over condition b at the $p < 0.01$ confidence level. The dotted arrow indicates a preference for layout of Graph 8 by HOLA over yFiles at the $p < 0.05$ level.

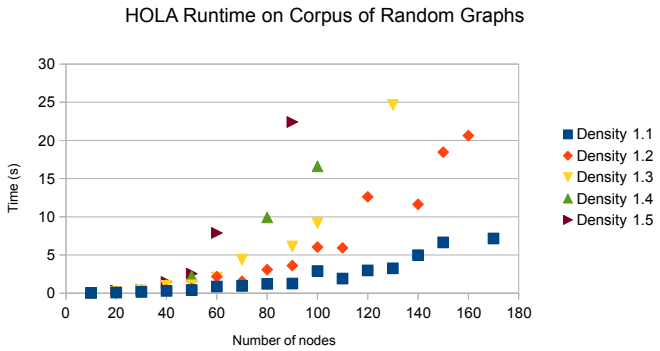


Fig. 8: HOLA running time on a collection of random graphs with between 10 and 170 nodes, and with densities between 1.1 and 1.5 edges per node. yFiles layout takes less than 2 seconds on each of these graphs. On the small graphs from our first study HOLA runtimes range from 9 to 97 milliseconds.

in which they could not advance until they had correctly identified a shortest path.

Part 3: This was similar to Part 2 but with a single red node and the task being to click on all neighbours of that node. There was again a training task. The red node was chosen to be a node of maximal degree but different from both of the red nodes chosen for Part 2, to avoid familiarity effects.

Part 4: The large graphs were shown in the two layouts HOLA and yFiles side by side, and users were asked to say which layout was better, and to briefly explain why in a text box.

Results & Discussion

Part 1: User rankings assigned with the gold, silver, and bronze medal icons in the online study were mapped onto the numbers 1, 2, 3 respectively, so that 1 was the best possible rank and 3 the worst. The rankings were analysed using Friedman's test. The aggregate mean rank across all graphs was 1.75 for Human, 1.80 for HOLA, and 2.46 for yFiles, and the null hypothesis was rejected (p = 2.607e-15). Thus, we can accept H1 and conclude that HOLA is of comparable quality to human layout for the eight small networks considered.

Post-hoc pairwise Nemenyi tests showed that both HOLA and Human layout were preferred over yFiles and hence, are an improvement over the best existing orthogonal layout algorithm (p = 2.9e-12 and p = 5.0e-11 resp.). On average over all graphs, neither HOLA nor Human layout was clearly preferred one over the other, as hoped in our intent to match human layout quality. The significant pairwise preferences for the eight individual graphs amongst the three layout methods are shown in Fig. 7.

Part 2: This compared user performance when finding the shortest path between two nodes on the larger graphs laid out using yFiles and HOLA. Both error rates and timings showed a high degree of leptokurtosis, so Wilcoxon's signed rank test was used. The mean error rates 0.162 for HOLA and 0.548 for yFiles differed significantly (p = 1.916e-14). The mean times 12.27s for HOLA and 29.15s for yFiles also differed significantly (p = 2.085e-15).

Part 3: This compared user performance when counting a node's neighbours on the larger graphs laid out using yFiles and HOLA. Again both error rates and timings showed a high degree of leptokurtosis, so Wilcoxon's signed rank test was used. The mean error rates 0.159 for HOLA and 0.349 for yFiles differed significantly (p = 4.364e-09). The mean times 10.10s for HOLA and 12.98s for yFiles also differed significantly (p = 2.848e-12).

The significant results for Part 2 and 3 together allow us to reject any null-hypothesis for H3 and accept that overall, participants were significantly faster and more accurate with HOLA layout than yFiles.

Part 4: Finally we compared user preferences on the larger graphs. User rankings were analysed using Wilcoxon's signed rank test. The aggregate mean rank across all graphs was 1.20 for HOLA and 1.80 for yFiles, which differed significantly (p = 5.123e-12).

Among individual graphs only the higher density graph on 90 nodes (Fig. 6c) showed no significant preference. For all others HOLA was preferred (p < 1.0e-5). This reinforces our decision to target lower density graphs, and shows that HOLA produces better layouts in those cases (H2).

Feedback: After giving their preference for the large graphs, users left comments in a free-text field. From 83 participants looking at six graphs, 359 comments were collected. There were 65 comments concerning proximity of connected nodes, e.g., *"Connected nodes are generally closer together, making their association more obvioius[sic]."* There were 12 comments mentioning *"structure,"* 8 of them favouring the "structure" of HOLA layout, e.g., *"Clearer to see, more structured layout."* Sixteen comments gave favourable mention to *"trees"* in HOLA, e.g., *"More familiar - like a family tree."* Briefly, other significant terms and the number of times they were mentioned were: *"Compactness"* (13 times), *"Symmetry"* (3 times), *"grid"* (4 times), and *"cross[ings]"* (7 times).

6 Conclusion and Future Work

We have presented a new "human-centred" methodology for automatic network layout algorithm design that uses both formative and summative user studies. We have used this new methodology to develop HOLA, an orthogonal network layout algorithm that achieves a layout quality comparable to that of a human and significantly better than previous automatic layout algorithms.

As part of the formative studies for HOLA we conducted a "human-authored layout" study in which users manually created an orthogonal layout for small graphs by moving nodes and manipulating the edge routes. In accord with previous studies we found that humans did not like crossings or bends, liked symmetry and regular grid-like node placement, uniform segment length and separated clusters. We also found that compactness was important. The most surprising result was most of the highly ranked human layouts contained unnecessary bendpoints whose purpose appeared to be to emphasise symmetry or ensure contiguous edges.

HOLA has potentially wide application in engineering and biology where orthogonal layout of networks is commonplace. However there are some ways in which we could improve it. The first is speed (see Fig. 8); parts of the algorithm are currently still implemented in Python which is sub-optimal. After porting to C++, HOLA will be released as part of the open-source Adaptagrams layout library [1].

The second possible improvement is extending HOLA to better handle dense graphs through integration of automatic cloning [15], edge-bundling [17] and compression techniques [10].

Acknowledgments

We acknowledge the support of the ARC through DP140100077.

References

[1] adaptagrams open-source constraint-based network layout software: `http://www.adaptagrams.org`.

[2] Page with our detailed results for the formative study http://data.graphlayout.net/HOLA/formative.

[3] Page with our detailed results for the summative study http://data.graphlayout.net/HOLA/summative.

[4] yFiles automatic network layout software: http://www.yworks.com/en/products/yfiles/.

[5] J. Abello, F. van Ham, and N. Krishnan. ASK-GraphView: A large scale graph visualization system. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):669–676, 2006.

[6] D. Archambault, T. Munzner, and D. Auber. TopoLayout: Multilevel graph layout by topological features. *Visualization and Computer Graphics, IEEE Transactions on*, 13(2):305–317, 2007.

[7] C. Batini, E. Nardelli, and R. Tamassia. A layout algorithm for data flow diagrams. *Software Engineering, IEEE Transactions on*, (4):538–546, 1986.

[8] W. Didimo, G. Liotta, and S. A. Romeo. Topology-driven force-directed algorithms. In *Graph Drawing*, pages 165–176. Springer, 2011.

[9] K. Dinkla, M. A. Westenberg, and J. J. van Wijk. Compressed adjacency matrices: untangling gene regulatory networks. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2457–2466, 2012.

[10] T. Dwyer, N. Henry Riche, K. Marriott, and C. Mears. Edge compression techniques for visualization of dense directed graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2596–2605, 2013.

[11] T. Dwyer, Y. Koren, and K. Marriott. IPSep-CoLa: An incremental procedure for separation constraint layout of graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):821–828, 2006.

[12] T. Dwyer, B. Lee, D. Fisher, K. I. Quinn, P. Isenberg, G. Robertson, and C. North. A comparison of user-generated and automatic graph layouts. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):961–968, 2009.

[13] T. Dwyer, K. Marriott, and P. J. Stuckey. Fast node overlap removal. In *Graph Drawing*, pages 153–164. Springer, 2006.

[14] C. J. Fisk and D. Isett. "ACCEL" automated circuit card etching layout. In *Proceedings of the SHARE design automation project*, pages 9.1–9.31. ACM, 1965.

[15] N. Henry, A. Bezerianos, and J.-D. Fekete. Improving the readability of clustered social networks using node duplication. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1317–1324, 2008.

[16] M. Himsolt. Comparing and evaluating layout algorithms within GraphEd. *Journal of Visual Languages and Computing*, 6(3):255–273, 1995.

[17] D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):741–748, 2006.

[18] W. Huang. Using eye tracking to investigate graph layout effects. In *Visualization, 2007. APVIS'07. 2007 6th International Asia-Pacific Symposium on*, pages 97–100. IEEE, 2007.

[19] W. Huang, S.-H. Hong, and P. Eades. Effects of sociogram drawing conventions and edge crossings in social network visualization. *J. Graph Algorithms Appl.*, 11(2):397–429, 2007.

[20] S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow. Incremental grid-like layout using soft and hard constraints. In *Graph Drawing*, pages 448–459. Springer, 2013.

[21] J. Manning and M. J. Atallah. Fast detection and display of symmetry in trees. *Congressus Numerantium*, (64):159–169, 1988.

[22] K. Marriott, H. Purchase, M. Wybrow, and C. Goncu. Memorability of visual features in network diagrams. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2477–2485, 2012.

[23] C. McGrath, J. Blythe, and D. Krackhardt. The effect of spatial arrangement on judgments and errors in interpreting graphs. *Social Networks*, 19(3):223–242, 1997.

[24] H. Purchase. Which aesthetic has the greatest effect on human understanding? In *Graph Drawing*, pages 248–261. Springer, 1997.

[25] H. C. Purchase. Performance of layout algorithms: Comprehension, not computation. *Journal of Visual Languages & Computing*, 9(6):647–657, 1998.

[26] H. C. Purchase, J.-A. Allder, and D. A. Carrington. Graph layout aesthetics in UML diagrams: user preferences. *J. Graph Algorithms Appl.*, 6(3):255–279, 2002.

[27] H. C. Purchase, D. Carrington, and J.-A. Allder. Empirical evaluation of aesthetics-based graph layout. *Empirical Software Engineering*, 7(3):233–255, 2002.

[28] H. C. Purchase, R. F. Cohen, and M. James. Validating graph drawing aesthetics. In *Graph Drawing*, pages 435–446. Springer, 1996.

[29] H. C. Purchase, R. F. Cohen, and M. I. James. An experimental study of the basis for graph drawing algorithms. *Journal of Experimental Algorithmics (JEA)*, 2:4, 1997.

[30] H. C. Purchase, C. Pilcher, and B. Plimmer. Graph drawing aesthetics–created by users, not algorithms. *Visualization and Computer Graphics, IEEE Transactions on*, 18(1):81–92, 2012.

[31] U. Rüegg, S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow. Stress-minimizing orthogonal layout of data flow diagrams with ports. In *Graph Drawing*, pages 319–330. Springer, 2014.

[32] S. B. Seidman. Network structure and minimum degree. *Social networks*, 5(3):269–287, 1983.

[33] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *Systems, Man and Cybernetics, IEEE Transactions on*, 11(2):109–125, 1981.

[34] F. van Ham and B. E. Rogowitz. Perceptual organization in user-generated graph layouts. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1333–1339, 2008.

[35] C. Ware, H. Purchase, L. Colpoys, and M. McGill. Cognitive measurements of graph aesthetics. *Information Visualization*, 1(2):103–110, 2002.

[36] M. Wybrow, K. Marriott, and P. J. Stuckey. Orthogonal connector routing. In *Graph Drawing*, pages 219–231. Springer, 2010.