The Usefulness of Constraints for Diagram Editing

Michael Wybrow, Kim Marriott and Linda McIver
School of Computer Science and Software Engineering
Monash University, 3800, Australia
`{mwybrow,marriott,lindap}@csse.monash.edu.au`

Peter J. Stuckey
Dept. of Computer Science and Software Engineering
University of Melbourne, 3010, Australia
`pjs@cs.mu.oz.au`

Abstract

This paper examines the usefulness of constraint-based alignment and distribution tools in graphical editors. Currently one-way constraints are used to provide alignment and distribution tools in many commercial editors. In this paper we discuss how limitations of one-way constraints lead to serious usability issues with such tools. To overcome these limitations, we show how to implement alignment and distribution tools using multi-way constraints. We then describe a usability study comparing these two implementations. This is the first usability study we are aware of that examines the relative usefulness of interactive graphical tools based on one-way and multi-way constraints.

1. Introduction

When editing diagrams and other graphical documents, it is often useful to be able to specify geometric relationships between the elements, for instance "left-align these three objects" or "equally space all the selected objects between the outer two". A once-off movement to fulfill this relationship can be done by simply adjusting the positions of shapes. However, in many situations one would like this relationship to be preserved during subsequent editing. Tools that set up such persistent geometric relationships are generally implemented using constraints.

A constraint specifies a relationship among element attributes that should be maintained. For instance, vertical alignment of three boxes A, B and C can be specified by

$$A.x = L.x$$
$$B.x = L.x$$
$$C.x = L.x$$

where L is an "alignment guideline" as in Figure 3. Over the last four decades there has been considerable effort in developing efficient constraint solving techniques for interactive graphical applications (?, ?, ?).

One-way (or data-flow) constraints are the simplest, most widely used approach (?, ?). They form the basis for a variety of commercial products including widget layout engines and the customisable graphic editors Visio and ConceptDraw. A one-way constraint is exactly like a formula in a spreadsheet cell. It has the form $x = f_x(y_1, ..., y_n)$ where the formula f_x details how to compute the value of variable x from the variables $y_1, ..., y_n$. Whenever the value of any of the y_i's changes, the value of x is recomputed, ensuring that the constraint remains satisfied. Thus in the above example they will ensure that if the alignment line is moved then the boxes will follow it. One-way constraints are simple to implement and can be solved extremely efficiently. They are also very versatile since f_x can be any function.

The main limitations of one-way constraints are that constraint solving is directional and that cyclic dependencies are not allowed, i.e. an attribute cannot be defined in terms of itself. Thus for instance in the above example if box B is moved the other boxes and alignment line will not follow it since the constraints only compute values for $A.x$, $B.x$ and $C.x$, and only as a result of changes to the value of $L.x$. For this reason so-called multi-way constraint solving techniques have been developed. These range from local propagation based approaches (?, ?) to simplex-based linear constraint solvers (?, ?, ?). In multi-way constraints, all variables can potentially be output variables—any variable can be calculated from the values of the other variables. With a multi-way constraint solver if B is moved then the other boxes and alignment line will follow.

Despite the large amount of research in the area of constraints and graphical editors, there is little or no empirical evidence to confirm the value of the various constraint-based systems that have been presented. In

1

particular there has been no investigation of the general claims that multi-way constraint-based tools are better than one-way constraint-based tools (?, ?, ?). This is the main contribution of this paper.

Our usability study looks at tools for aligning and distributing shapes. Single-use tools are very easy to implement and offered by practically all editors. Some commercial editors, such as Visio and ConceptDraw, also provide tools which allow the user to create persistent alignment and distribution relationships that are implemented using one-way constraints. As the basis for our study we have designed and implemented a set of alignment and distribution tools based on multi-way constraints and integrated these into Visio. We have designed and conducted a comparative usability study to gather empirical data about the relative usefulness of the different levels of constraint-based tools. We compare tools based on one-way and multi-way constraints, comparing them to similar single-effect tools which do not make use of any constraints.

Section 2 provides motivation for the research, by introducing and discussing shortfalls and limitations of existing single-effect and one-way constraint-based alignment and distribution tools. Section 3 discusses the design of our multi-way constraint-based tools. Section 4 describes the usability study. Finally, section 5 looks at the results of the study.

2. Existing Tools

This section explains the motivation for the research, describing the shortfalls and limitations of the single-effect and one-way constraint-based placement tools provided by many commercially available editors.

2.1. Rarity of constraint-based tools in graphical editors

There are dozens of papers presenting constraint-based techniques and tools, yet most mainstream, commercially available graphical editors do not provide such tools. Editors that do provide constraint-based tools generally provide persistent alignment and distributions using guidelines. This is a popular method making use of one-way constraints as illustrated in the Introduction. Some editors that provide this kind of placement functionality are Visio (?, ?), ConceptDraw (?, ?), and Omni-Graffle (?, ?).

There are a couple of reasons why tools based on other, potentially more powerful constraint solving techniques such as multi-way constraints have not made their way into commercial graphical editors.

Firstly, because of their rarity there is little or no evidence of their value. Secondly, one-way constraint solvers are simple to write and extremely efficient. An efficient multi-way constraint solver is much more complex to write and may require considerable numerical programming abilities. Most authors of graphic editing software are not likely to have the time or knowledge to write their own multi-way constraint solver, at least not without compelling evidence that the tools are going to be of great value to the user.

2.2. Microsoft Visio and its native tools

As stated in the previous section, Microsoft Visio is fairly typical of other leading commercial editors in the placement tools that it provides. It natively offers dynamic connectors, single-effect alignment and distribution tools as well as persistent one-way constraint-based alignment and distribution tools. These are fully implemented and accessible through a clear graphical interface.

Microsoft Visio is commonly recognised as being the leader in diagramming software. In addition to widespread use in industry, many other leading diagramming packages either directly compare their product to Visio or offer support for importing and exporting to the Visio file format (?, ?, ?, ?).

Basic requirements for constraint-based editing tools are fairly easy to determine using well known usability principles. The constraints themselves should be hidden from the user by a good metaphor. They must also provide consistent, predictable behaviour for the user. We now evaluate the tools provided by Visio in light of these principles.

2.2.1. Once-off shape alignment and distribution

Visio provides once-off alignment and distribution tools that effectively re-adjust the positions of the involved objects. There is no lasting relationship created. When shapes have been aligned in this way their physical layout on the page will have changed, but Visio will not treat them any differently. Alignments work by adjusting the positions of all the shapes in the selection (excluding connectors) to align with the lead object. Figure 1 shows changes to a diagram as a result of left-aligning all shapes with Shape B.
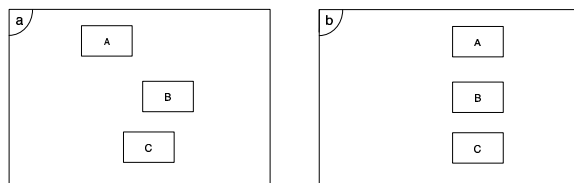

Figure 1. Left-aligning all shapes with lead object Shape B

A distribution applied to a selection will leave the two outer objects where they are and distribute all the other objects in the selection equally between. The user has control over the specific type of distribution used. Once again there is no lasting relationship created from the use of the distribution tool. An example of the distribution action can be seen in Figure 2 where all shapes have been horizontally distributed by their center.
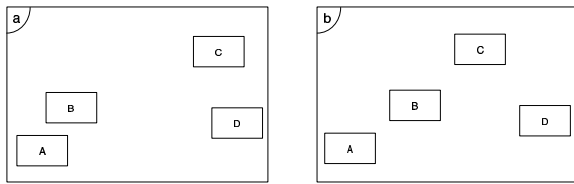
2

Figure 2. Horizontal distribution of all shapes by their center

2.2.2. Persistent alignment and distribution relationships

In addition to its single-effect tools, Visio provides a persistent form of alignments and distributions through the use of guidelines. Guidelines are purely placement aids; they act like normal manipulable objects on the page but are not part of the final diagram (i.e. they will not be visible on printed versions of the diagram.)

The persistent alignment is similar to the single-effect version with the addition of guidelines. The alignment tool works by creating a guideline connected to the lead object in the alignment. It then adjusts the positions of all the other objects to bring them in line and glue them to the guideline, as shown in Figure 3.
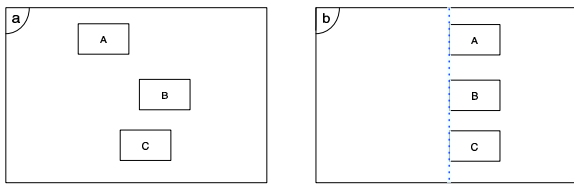

Figure 3. Persistent left-alignment of all shapes with Shape B

Once shapes have been "glued" to a guideline, they can be moved by moving the guideline. Unfortunately using one-way constraints only allows us to specify that the shape is constrained to align with the guideline. What can't be done is to specify that the guideline is contrained to align with the shape. As a result, moving shapes directly, without using the guideline, will always unglue them from guidelines, removing them from any placement relationship they are involved in.

The alignment and distribution tools themselves are required to directly move shapes to set up relationships. Often, such movement will break shapes from their prior alignment or distribution relationships. Thus, often shapes will only be attached to their most recently established placement relationship (and guideline).

For example, if a shape is involved in a vertical alignment, i.e. glued to a vertical alignment guideline, and an action (either manual or tool-based) causes it to be moved in only the vertical plane we expect it to effectively slide up or down the guideline, while staying glued to it. Unfortunately, the shape also gets unglued from the guideline in this case.

Additionally, there is no visible indication that a shape is glued to a guideline unless that shape is currently selected. This means that since the shape has not visually moved away from the guideline, the constraint will be broken without any visual feedback to the user. Such behaviour means the user is unable to fully understand the state of the diagram from its onscreen representation.

These problems are illustrated in Figure 4 where two alignment relationships are set up, both involving Shape B, a vertical alignment in Figure 4(a), followed by a horizontal alignment in Figure 4(b). In creating the second relationship, Shape B's position is altered to be dependent on the position of the horizontal guideline—an action that invisibly removes the shape from the vertical alignment relationship. Moving the most recently created alignment in Figure 4(c) works as expected. Then in Figure 4(d), manipulating the older alignment relationship, we see that shape B is no longer constrained to follow the guideline. This behaviour is undesirable, since it makes it hard for the user to predict the response of the system. Obviously relationships will behave in different ways depending on the order in which they were set up.
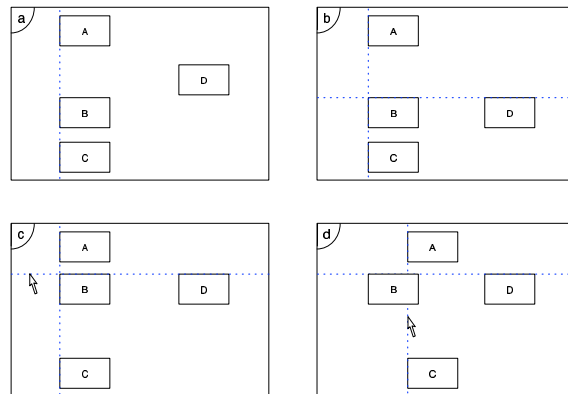

Figure 4. Unexpected behaviour due to limitations of one-way constraints

This particular example quite clearly illustrates problems with shapes being invisibly broken from guidelines. The major weakness of a one-way constraint-based set of tools will be that relationships will often be broken by direct manipulation or other tools that affect the positions of shapes.

It should be noted that in the example presented in Figure 4, the action that caused Shape B to be broken from the vertical alignment—the creation of the horizontal alignment—should not actually require the first constraint to be broken. Using one-way constraints it is possible to have both the x and y position of a shape constrained to follow different guidelines. This particular behaviour appears to be a bad design choice in Visio. It is important since, coupled with the problems of one-way constraints, it almost certainly has a negative impact on the usefulness of Visio's persistent placement tools.

Visio does provide "snap-dragging" as a means of re-

attaching an object to a guideline. Snap-dragging is a technique which uses a gravity metaphor where, as the user drags a shape, it will snap and connect to significant objects (such as guidelines). However, because this method requires quite precise manipulation, being forced to repeatedly make use of it can be tedious for the user.

Like alignment, Visio's persistent distribution tools are similar to the single-effect versions, with the addition of guidelines. Visio considers all the shapes (aside from connectors) in the selection. It creates a guideline for each shape involved and glues the shape to it. The tool then takes the outer two guidelines and distributes the other guidelines (and attached shapes) equally between as seen in Figure 5.
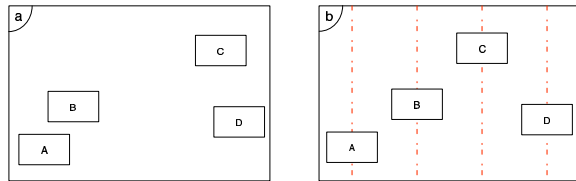

Figure 5. Persistent horizontal distribution of all shapes by their center

As a result of using the tool, we end up with a persistent relationship that can be further manipulated. The outer guideline on each end of the distribution can be dragged, effectively resizing the entire distribution. The other guidelines in the distribution cannot be dragged, because they are dependent on the positions of the outer guidelines.

This behaviour is sufficient for the basic case of distributing shapes, but we again run into the limitations of one-way constraints when trying to distribute shapes involved in alignment relationships. Unless we explicitly select the guidelines themselves for distribution the tool must act on and move the individual shapes, effectively ignoring (and removing them from) any alignment relationships they are a part of. Since this means the user can only distribute aligned groups of shapes by their guidelines, this behaviour violates usability principles that suggest systems should allow the user to arbitrarily substitute equivalent values for each other.

This is unfortunate. While users may consider that the tool creates persistent relationships, these relationships are only truly persistent for as long as they remain untouched by manipulation or the creation of other relationships.

3. Multi-way constraint-based alignment and distribution tools

Thus we have seen that from the user's perspective one-way constraint-based alignment and distribution tools have a serious drawback: alignment and distribution relationships can silently break as a result of direct manipulation of an object in the relationship or because more than one constraint is applied to the same object and the constraints interfere. Although, in part this is to blame on details of the Visio tools, the problem is inherent in basing tools on one-way constraints since each constraint has a fixed direction and an attribute can only have a single formula associated with it.

We hypothesise that placement tools should provide truly persistent alignment and distribution relationships—two shapes put into an alignment relationship should stay aligned through all further editing until the relationship is explicitly removed. The tools can then accurately use the metaphor of an alignment relationship as description, without the user needing to think about them as shapes glued to guidelines. This shifts the interaction with the tools to use of a more familiar, higher level concept.

As one-way constraints cannot support this, we must move to tools that are based on multi-way constraints. In this section we describe multi-way constraint-based alignment and distribution tools that we have integrated with Visio. Our tools are written in C++ and compiled as a Visio add-on Dynamic Link Library (DLL) with Microsoft Visual C++ 6.0.

We chose Microsoft Visio Professional as the platform for development and testing our multi-way constraint-based alignment and distribution tools because it offers a heavily customisable platform for developing diagramming solutions. Most commercial graphical editors do not provide support for developer plug-ins. Visio was chosen over the alternative of modifying an open source editor (such as XFig (?, ?) or Dia (?, ?)) for three reasons. Firstly, it is widely used in industry, which makes the outcome of the research relevant and interesting to a greater group of people. Secondly, it is heavily customisable and provides support for writing add-ons that can neatly extend Visio's own tools and features. Thirdly, being an Office application it shares the common Microsoft Office interface, meaning it will already be partially familiar to anyone who has experience with Office products. The relatively wide exposure of Office applications means that by using Visio for the development and accompanying study, we are less likely to confound the measurement of the tools' usefulness with interface usability issues.

3.1. Design space

Our implementation of these tools makes use of a multi-way constraint solving toolkit, QOCA (?, ?). It allows us to create and solve systems of multi-way linear constraints where the importance of each constraint is given by a constraint hierarchy (?, ?).

Multi-way constraints provide the ability to set up the initial alignment relationship so that moving the guideline moves the group of shapes attached to it, and moving any or all of the shapes also moves the entire group (including the guideline) where this still satisfies any other active constraints—the aligned group will stay aligned throughout all further editing.

4

Constraint hierarchies allow us to express what to do when constraints conflict by attaching an importance to each constraint. If not all constraints can be simultaneously satisfied, then the most important constraints are satisfied and less important constraints are not satisfied. This means we can effectively express a desire for particular variables to be perturbed in preference to others, allowing us to specify preferred solutions and gain some control over the behaviour of the solver.

3.2. Alignment

The creation of an alignment relationship acts in the same way as Visio's existing tools—a guideline (if one does not exist) is created and aligned with the lead object, and all other shapes in the selection will be seen to move to initially align with the guideline. It is only when we begin further manipulation on the diagram that differences between the tools become evident.

The multi-way nature of the created relationship is clearly visible in Figure 6 where when Shape B is moved down and to the left, the two alignment relationships cause both Shape A and C to be moved as a result.
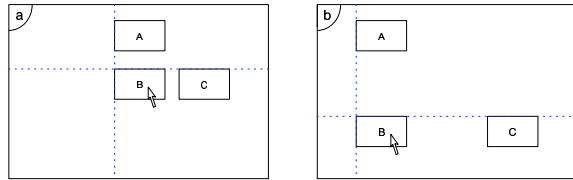

Figure 6. Effects on page due to moving Shape B down and left

An alignment relationship can be removed by deleting the visible indicator of the relationship—the alignment guideline. A shape can effectively be added to an alignment relationship by aligning it with any (or every) shape in the existing relationship.

3.3. Distribution

The basic behaviour of the distribution tool is similar to Visio—it considers all the shapes in the current selection, spacing them all equally (by their center, left or right) between the two outermost shapes. The difference is that the user can distribute shapes involved in alignment relationships and those relationships will stay active. Basically, when the user applies this tool, any group of aligned shapes will remain aligned, appearing to be treated as a single object for the purpose of distribution. This behaviour is easily demonstrated in Figure 7, where all shapes have been selected and distributed horizontally by their center.

Selected shapes without associated guidelines will have guidelines created for them and these guidelines will be the subject of the actual constraints controlling the distribution relationship. A colour change to guidelines
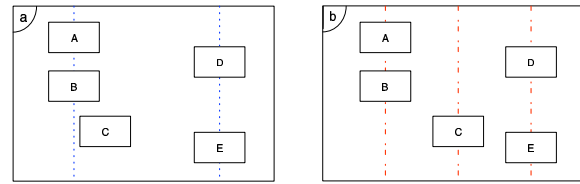

Figure 7. Distribution of all shapes (three effective columns)

is given as a visual indicator that allows distinction between pure alignment guidelines and those also involved in a distribution. The change in colour is indicated in Figure 7 where the alignment guidelines in (a) change when they become part of the distribution in (b).

Once a distribution relationship has been set up, moving the center guidelines involved in the distribution (either by direct manipulation, or movement of a shape "attached" to one) has the effect of moving the entire set of objects involved in the distribution. This is effectively the same result given by selecting all the objects involved and moving them as a group. Conversely, dragging an outer guideline (or "attached" shape) has the effect of growing or shrinking the entire distribution. Movement of the other guidelines involved cause a combination of effects weighted in relation to their proximity to the center or outside of the distribution.

A distribution relationship can be removed by selecting and deleting all the involved guidelines. Like alignment, the distribution relationships stay active until they are explicitly deleted. When the user deletes just a single guideline from the distribution, the distribution relationship is removed. The other guidelines from the relationship remain, but revert back to being plain alignment guidelines, visually changing colour to indicate this.

4. The usability study

This section describes the usability study that was designed and executed for the purpose of collecting empirical data about the comparative usefulness of the different levels of constraint-based placement tools.

4.1. Basic design

The intended purpose of the placement tools is to aid the user in creating and modifying diagrams. Thus, in the study we used both completion time and diagram correctness as measures for the "comparative usefulness" of the tools. We are concerned with how long a participant requires to complete an exercise—obviously we would expect more usable tools to lead to shorter completion times. As an additional measure, we are interested in the relative number of errors found in the "completed" diagrams created by the participants using the different tools. Here we expect more usable tools to result in fewer errors.

The basic design for the study was a set of exercises in which the participants were asked to create, modify

and manipulate diagrams that resembled basic flowcharts. Flowcharts were chosen because they are simple diagrams that share many properties with more complex forms of diagrams.

Since we were evaluating the usefulness of constraints in the use of placement tools, the focus of the exercises are shape placement and overall diagram layout. On one hand, we wanted the exercises to be simple enough not to require any prior knowledge of flowcharts. On the other hand, we did not want the exercises to be so simplistic that they seem contrived. We felt it was important the exercises seemed like realistic tasks. To this end, the diagrams given in the exercises were realistic flowcharts and the layout changes requested in the exercises always increased the aesthetic value of the diagram.

4.2. The three levels

Participants in the study were randomly assigned to one of three groups. Each group was provided with a different level of constraint-based tools for alignment and distribution.

The three levels are described below:

- **Group A**: Single-use alignment and distribution tools are available. These move the involved shapes but do not create a lasting relationship.

- **Group B**: Visio's native form of persistent alignment and distribution tools based on one-way constraints are available. The single-effect tools are not available to the participant.

- **Group C**: Our truly persistent alignment and distribution tools based on multi-way constraints are available. The single-effect and one-way constraints tools are not available to the participant.

All participants were given exactly the same exercises, but made use of only the particular tools offered to their group. Training differed slightly for each group to ensure participants knew how to use the tools available to them.

4.3. Hypothesis

It was hypothesised that the persistent state of the relationships set up by the tools in Group B would make them faster and less error-prone than the single-effect Group A tools. Likewise, we hypothesised that the truly persistent nature of the multi-way constraint tools in Group C would make them faster and less error-prone than the one-way constraint tools of Group B.

4.4. Participants, Environment, Instrumentation

Thirty people were tested; ten in each of the three groups. There were no requirements for the participants other than that they be computer-literate adults. All participants were undergraduate university students, as they were readily available. Thirty individual people were used, participants were not reused across groups. Part of what was being examined were the natural strategies a user takes to solve a problem. It was suspected that people who had already taken part in the experiment once would have learnt the idiosyncrasies of the particular tool set they used and they would then apply this knowledge the next time round.

All tests were carried out in private, the investigator testing a single participant at a time. The environment for the experiment was a usability lab in which the participant sat at a terminal while the investigator sat behind them, observing and taking notes.

A record of each user's interaction with Visio during the tests was obtained by taping a video feed of the test computer's monitor to VHS cassette. A small amount of audio data from post-test debriefing and discussion was also captured to the tape. In addition to this, time data for the start and finish times of each exercise were taken down by the investigator. This includes the time taken to comprehend and complete each exercise. Other notes taken by the investigator highlighted the strategy and method taken by the user to carry out the task, as well as problems they may have experienced.

A short pre- and post-test survey was used as a means of obtaining some additional qualitative and quantitative data about users' experience with related tools, how difficult they found the exercise and suggestions they had for the software's improvement.

At the beginning of each experiment the participant was shown a 15 minute training video. Each participant watched a common introduction to Visio, as well as a specific introduction to the tool set they would be using. Following this, the participant was asked to carry out some training tasks in an informal environment where the investigator would answer questions related to the software. When the participant had completed these tasks and was comfortable with Visio and its tools they proceeded to the timed exercises.

4.5. The exercises

In the exercises, participants edited some simple flowcharts. The exercises also required the participant to make layout changes to the diagrams—spacing the objects on the page or aligning them to make the diagram more aesthetically pleasing. Some of the instructions and final diagrams show a generic representation of alignment or distribution relationships, in this case we required the participant to enforce this relationship but they were free to make use of additional placement relationships if they felt this would make the task quicker or easier.

The exercises where done one at a time, in order. For each exercise, the participant was given a three page instructional handout. The first page showed a typed description of the task, written in point form in plain English. The second page showed the starting-point diagram, a print out of the diagram they would be given to
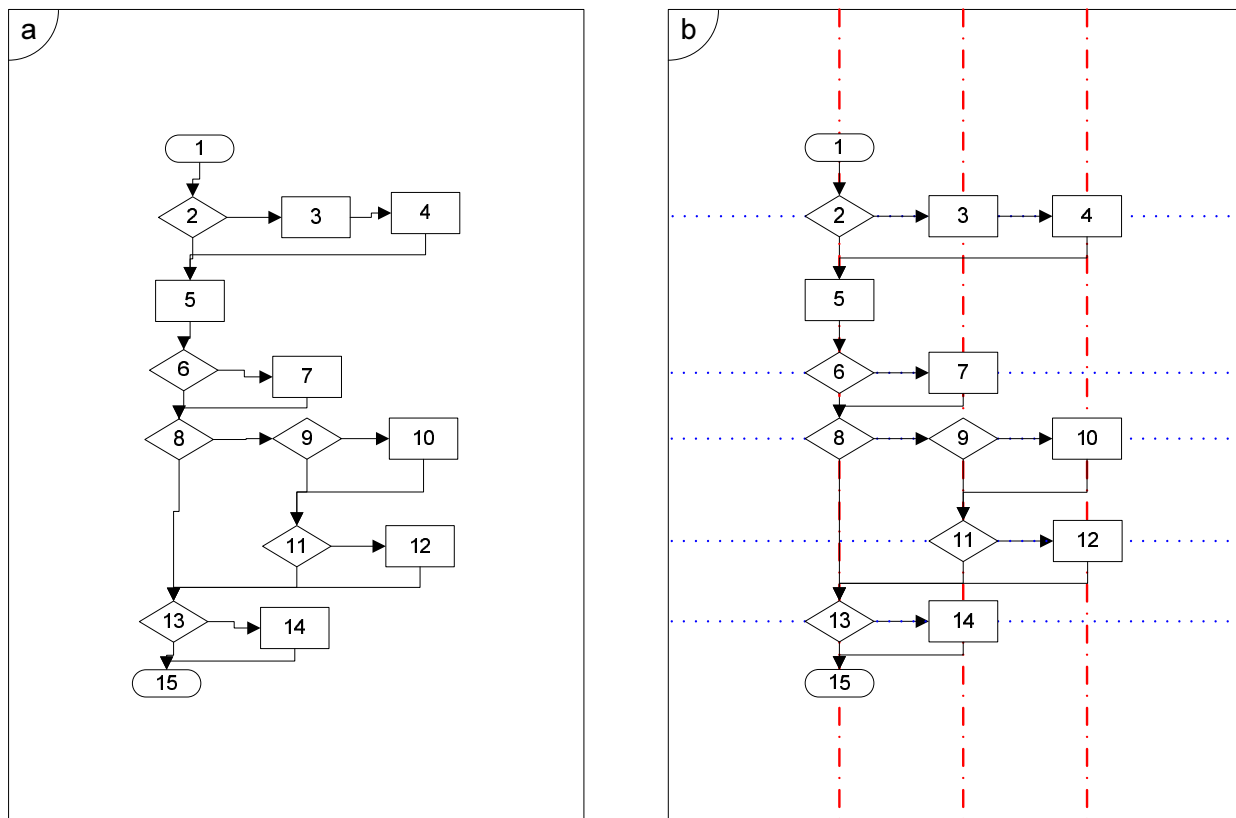
Figure 8. Starting-point and target diagrams for the "Manipulation 1" exercise

work with for the task. The third showed the target diagram, the result of applying the specified instructions to the initial diagram.

The five timed exercises, from which data was collected, are as follows:

- **"Editing"**: A simple exercise aimed at letting the participant get familiar with the timed conditions. It is also useful for the purposes of determining the methods and order of actions taken by the participant to solve the task.

- **"Choice"**: Another general editing exercise. No placement relationships are explicitly mentioned in the instructions or shown on the final diagram but these relationships could be inferred from the final diagram they have been given.

- **"Manipulation 1" and "Manipulation 2"**: These two exercises were designed as a pair. The first exercise requires the participant to add some alignment relationships and a single distribution. As an example, the starting-point and target diagrams for this task are shown in Figure 8.

  The second exercise requires that the diagram be resized to take up all of the available page. In this exercise there is no modification to the objects in the diagram aside from this placement. The alignment and distribution relationships do not change either.

- **"Grid"**: The fifth and final exercise requires modifications to another pre-constructed diagram, specifically the participant must set up vertical and horizontal alignments as well as both vertical and horizontal distributions. The final diagram shows a grid-like arrangement of shapes which makes use of most of the available space on the page.

5. Results

In this section we present and discuss the results of the usability study. We consider completion times for the exercises, as well as errors in the completed diagrams.

For the analysis we use well-known statistical techniques (?, ?). To determine overall statistical significance we use a one-way randomised Analysis of Variance (ANOVA), where we consider $p < 0.05$ to be statistically significant. In the case of unequal group variances, as determined by Levene's test, the comparison of differences between means is instead achieved with a one-way ANOVA using the General Linear Model (GLM) in Minitab. As there has been no prior empirical analysis in this area, we are concerned where among the groups any significant differences (if any) lie. For this reason we use Tukey's HSD test, a form of post hoc comparison, with p set at 0.05.

In our analysis we have excluded the results of exercises where the participant did not finish. It is interesting to note that in total five people quit exercises, four from Group B and one from Group C. Also, participants quit

7

only during the "Manipulation" and "Grid" exercises.
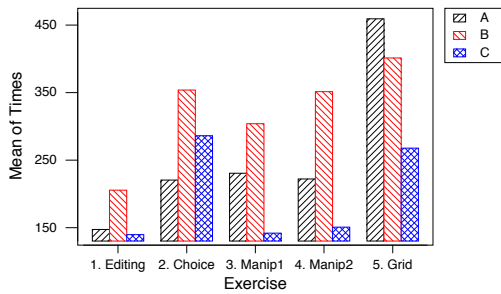
5.1. Completion times

Figure 9. Mean completion times (seconds)

The average completion times for each exercise are shown in Figure 9. To determine where the statistical significance lies we perform an ANOVA for each exercise.

A one-way ANOVA shows borderline significance for the first two exercises. The first exercise ("Editing", $F = 3.48, p = 0.046$) is a basic editing task not requiring any real use of the features offered by the alignment or distribution tools. The second exercise ("Choice", unequal group variances, $F = 3.52, p = 0.045$) involves optional use of the tools. Further analysis, by applying Tukey's HSD test, reveals there to be no significant difference between groups in times for the first exercise, and there to be a significant difference in the second exercise between Group A and Group B. This difference may be explained by participants in Group B who chose to experiment with the use of the tools during this exercise, increasing their completion times. Placement tools would not be expected to have an effect on basic editing (excluding shape placement), it therefore is not surprising that greater statistical significance was not seen in these exercises.

We do find there is significance in the completion times for the exercise "Manipulation 1" (unequal group variances, $F = 7.19, p = 0.004$). Times for this exercise are summarised in Figure 10. Figure 10 is a standard boxplot, showing a measure of spread. The boxes in the figure show the range of the middle 50% of the data, while the whiskers stretch to the largest and smallest values that are not "outliers". Outliers, those points more than 1.5 times outside the range of the middle 50%, are marked with a '*'. We have also added, to the bottom of the boxplot, the mean completion time and standard distribution (in brackets) for each group.

To see exactly where the significance lies we use Tukey's HSD test to consider all pairwise differences between level means. Using this method we find that the only significant difference is between Group B and Group C. In this exercise it is clear that the multi-way constraint-based tools of Group C definitely offer some benefit over the one-way constraints of Group B.
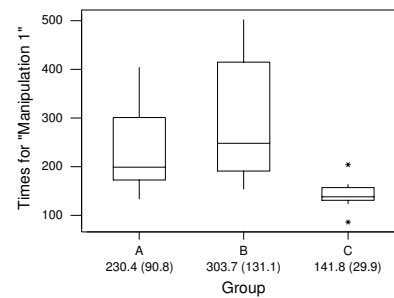
Figure 10. Boxplot of completion times for "Manipulation 1" exercise

We again determine there is significance in the completion times for the exercise "Manipulation 2" ($F = 5.61, p = 0.010$). Times for this exercise are summarised in Figure 11. Once again, using Tukey's HSD test, we find that the only significant difference is between Group B and Group C. This exercise saw the participants manipulating relationships they had set up in the previous exercise. We see that Group C also benefits over Group B in this aspect of editing.
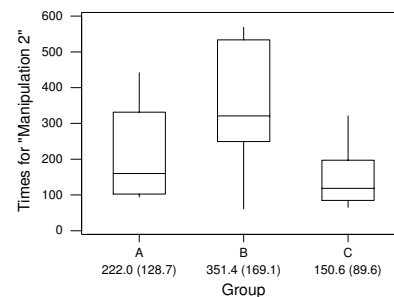
Figure 11. Boxplot of completion times for "Manipulation 2" exercise

In the study we made several observations that might explain why Group B offered no significant benefit over Group A for the "Manipulation" exercises. Participants in Group A participants had to reuse the tools repeatedly to keep objects in the desired relationships. Group B participants tended to have to do the same. Some shapes for them stayed in relationships, but a large number became unglued, leading not only to disassociated shapes but also to disassociated guidelines that no longer carried any meaning. Such objects clutter the page and manipulation of them tended to be misleading and confusing for the participant. In fact, some users found it easier to delete such guidelines and continually recreate the relationships, effectively mimicking the usage of the single-effect Group A tools.

The final exercise ("Grid") also showed a significance in completion times (unequal group variances, $F = 7.01, p = 0.004$). Times for this exercise are summarised in Figure 12. Tukey's HSD test showed that there was significance between times for Group A and Group C, and also between times for Group B and Group C. This shows that the multi-way constraints of Group C are more ben-

eficial for construction of heavily aligned and spaced diagrams than the alternatives of Group A and Group B.
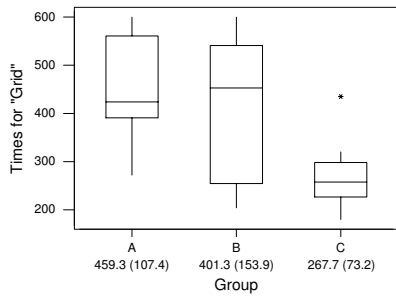

Figure 12. Boxplot of completion times for "Grid" exercise

We are also interested in whether there was any interference between the groups and the exercises. Figure 13 shows group means as an interaction plot with error bars. An absence of interaction is illustrated by the relatively parallel lines of Group B and Group C. This suggests that where we have seen significance, it is not due to the benefit of the tools for the particular individual exercises, but rather it is a benefit seen across all tasks. From the plot we can see that the mean scores from Group C are faster than the scores from Group A.
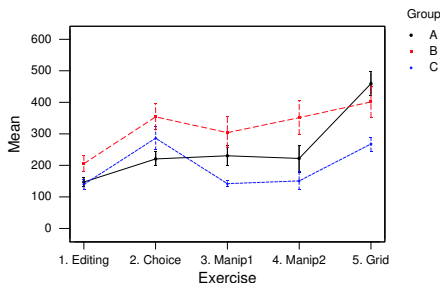

Figure 13. Interaction plot for Groups A, B and C

Perhaps the most interesting is the interaction between Group A and Group B. The plot shows that while Group A out-performs Group B (by means) on most exercises, the result is reversed for the final exercise. Since the Group B tools are a persistent form of the Group A tools, one may naively have expected Group B to out-perform Group A across the tests. We found no significant evidence to support this. In fact, the time values in Figure 13 suggest that Group B tools provide worse performance on all but the final exercise. This supports the observation that these tools suffer from usability problems. Though it is perhaps a surprise exactly how much impact these problems actually have on the tools' usefulness in terms of diagram editing time.

The "Grid" exercise is interestingly the only exercise to look like showing any kind of positive difference between Group B and Group A. An explanation of this might come from the fact that the exercise does not in-

volve any manipulation of relationships once created. We also observed that many participants had learnt the quirks of the one-way constraint-based tools and had devised a particular order in which they could use the tools that would minimise placement relationships from breaking.

5.2. Error rates

We also collected information about the number of errors present in participant's final diagram for each exercise. Diagrams were compared by eye to the target diagram. We classified errors as characteristics such as failure to carry out particular task instructions, as well as shapes not part of required alignment or distribution relationships—easily determined by the presence of kinked connectors.
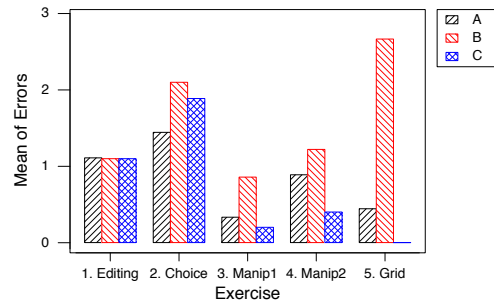

Figure 14. Mean errors in final diagrams

The raw averages for errors in the final diagrams are shown in Figure 14. Apart from Group C having significantly less errors than Group B in "Grid" ($F = 5.79$, $p = 0.009$), these results were not statistically significant. Though by looking at the graph we can see that Group C mostly leads to less errors than Group A and Group B. Here again, in the exercises requiring real use of placement tools, we see that the one-way constraint-based tools of Group B are again more detrimental to performance than their simple single-effect Group A counterparts.

6. Conclusions

We have evaluated the relative usefulness of different levels of constraint-based placement tools for the general task of constructing and editing diagrams. We examined single-effect tools present in most graphical editors, along with one-way constraint-based tools offered by several leading editors, and multi-way constraint-based tools of our own design.

We have presented the design for a set of placement tools based on multi-way constraints. We have implemented and tested this design against existing tools. Our results showed these to offer significant benefit over one-way constraint-based tools for tasks requiring the alignment and distribution of shapes. They were also found to consistently average faster completion times and fewer errors in the completion of such tasks. We have argued

that the usability issues of tools based on one-way constraints can be easily addressed through the use of multi-way constraints instead. Our results support this.

Interestingly, our results show persistent placement tools based on one-way constraints offer no significant advantage over the simple, single-effect tools offered by nearly all commercial editors. The persistent tools can be thought of as an extension of the single-effect tools, yet we have showed they provide no added value to the user for general editing and layout tasks. In fact, from the results, it can be seen that one-way constraint-based tools mostly lead to slower times and more errors in the finished diagram than single-effect tools. We suggest this is due to the very nature of one-way constraints which limits the flexibility of such tools, negatively affecting their usability.

Multi-way constraint-based tools were not found to offer statistically significant advantage over single-effect tools in all tasks, though in tasks requiring alignment and distribution of shapes they consistently resulted in faster average completion times and fewer errors in the final diagram. Given this, it could be argued that significance would be seen given further testing.

However, we are concerned that the questionable design choices with the behaviour of Visio's one-way constraint-based tools (as discussed in section 2.2.2) may have quite severely affected their usability. Further work will involve reimplementing the tools to remove this possible confounding factor, and to conduct additional experiments with the aim of gaining a more accurate comparison of constraint-based tools.

Acknowledgements

References

Badros, G. J. (1998). *Constraints in interactive graphical applications*. Unpublished doctoral dissertation, Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2350.

Borning, A., Freeman-Benson, B., & Wilson, M. (1992). Constraint hierarchies. *Lisp and Symbolic Computation*, *5*(3), 223–270.

Borning, A., Marriott, K., Stuckey, P., & Xiao, Y. (1997). Solving linear arithmetic constraints for user interface applications. In *Proceedings of the 10th annual acm symposium on user interface software and technology* (pp. 87–96). ACM Press.

Computer Systems Odessa. (2002). *ConceptDraw Home Page*. Web Page. (http://www.conceptdraw.com/)

Freeman-Benson, B. N., Maloney, J., & Borning, A. (1990). An incremental constraint solver. *Communications of the ACM*, *33*(1), 54–63.

Hill, R. D. (1993). The Rendezvous constraint maintenance system. In *Proceedings of the 6th annual acm symposium on user interface software and technology* (pp. 225–234). ACM Press.

Hower, W., & Graf, W. H. (1996). A bibliographical survey of constraint-based approaches to CAD, graphics, layout, visualization, and related topics. *Knowledge-Based Systems*, *9*, 449–464.

Larsson, A. (2002). *Dia Home Page*. Web Page. (http://www.gnome.org/projects/dia/)

Marriott, K., Chok, S. S., & Finlay, A. (1998). *A tableau based constraint solving toolkit for interactive graphical applications*. New York: Springer-Verlag.

Microsoft Corporation. (2002, Nov). *Microsoft Visio Home Page*. Web Page. (http://office.microsoft.com/visio/)

Sannella, M., Maloney, J., Freeman-Benson, B. N., & Borning, A. (1993). Multi-way versus one-way constraints in user interfaces: Experience with the deltablue algorithm. *Software - Practice and Experience*, *23*(5), 529-566.

SmartDraw Software. (2003). *SmartDraw Home Page*. Web Page. (http://www.smartdraw.com/)

Snodgrass, J. G., Levy-Berger, G., & Haydon, M. (1985). *Human experimental psychology*. New York: Oxford University Press.

The Omni Group. (2002). *OmniGraffle Product Page*. Web Page. (http://www.omnigroup.com/omnigraffle/)

Vander Zanden, B. T., Halterman, R., Myers, B. A., McDaniel, R., Miller, R., Szekely, P., Giuse, D. A., & Kosbie, D. (2001). Lessons learned about one-way, dataflow constraints in the Garnet and Amulet graphical toolkits. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, *23*(6), 776–796.

xfig.org. (2002). *XFig Home Page*. Web Page. (http://www.xfig.org/)