Using Animation in Diagrammatic Theorem Proving

Daniel Winterstein[1], Alan Bundy[1], Corin Gurr[1], and Mateja Jamnik[2]

[1] Division of Informatics, University of Edinburgh, 80 South Bridge, Edinburgh, EH1 1HN, U.K.
danielw@dai.ed.ac.uk, bundy@dai.ed.ac.uk, corin@cogsci.ed.ac.uk
[2] School of Computer Science, University of Birmingham, Birmingham, B15 2TT, U.K.
M.Jamnik@cs.bham.ac.uk

Abstract. Diagrams have many uses in mathematics, one of the most ambitious of which is as a form of proof. The domain we consider is real analysis, where quantification issues are subtle but crucial. Computers offer new possibilities in diagrammatic reasoning, one of which is animation. Here we develop animated rules as a solution to problems of quantification. We show a simple application of this to constraint diagrams, and also how it can deal with the more complex questions of quantification and generalisation in diagrams that use more specific representations. This allows us to tackle difficult theorems that previously could only be proved algebraically.

1 Introduction

In adapting diagrammatic reasoning to computers, there is the exciting possibility of developing diagrams in new ways. Diagrams in textbooks are necessarily static. However, if we consider the very real differences between text and hypertext, we see that diagrammatic reasoning on computers need not be just a straight conversion of diagrammatic reasoning on paper. One such new direction is the use of animation. In this paper we describe how animation can be applied to represent and reason about quantification. Other applications are possible: although unrelated to our present work, animated diagrams may also be useful in representing and reasoning about temporal relations. There is an obvious attraction in using time to represent itself.[1]

From the very beginning of mathematics, diagrams have been used to give proofs in subjects such as geometry and number theory. Nevertheless diagrammatic proof is only partially understood today. In particular, we do not have a general theory for handling quantification and the related topic of generalisation.

[1] This would probably not be suitable for domains which involve precise time calculations, as these would be hard to judge in an animation. For qualitative reasoning though, or as part of a mixed system, it seems an interesting line for future research.

We identify three distinct problems in this area: quantifier hierarchy, quantifier type and identifying generalisation conditions. In sentential reasoning, quantifier hierarchy is determined by reading from left-to-right. Quantifier type is determined by the semantics attached to the symbols $\forall, \exists$ (which are part of the common language of scientists and mathematicians), and generalisation is controlled by explicit conditions (e.g. $\forall A, B, C.\mathtt{triangle}(ABC) \wedge \mathtt{angle}(A, B) = 90^o \ \Rightarrow \ ...$). Unfortunately these solutions do not naturally carry over to diagrams.

The use of two dimensions with several spatial relations being significant removes the neat left-to-right ordering on objects that we have in sentential reasoning. For quantifier type, we could try to label objects with the $\forall, \exists$ symbols as in algebra, but it is not always clear which object such a label applies to, especially if the diagram involves composite objects. Fig. 1 gives an example of how this could be less than clear (does the $\forall$ symbol apply to the closest line, triangle or square?). Unlike algebra, where the conditions on a theorem are explicitly stated, diagrams often contain a lot of information that may or may not be relevant. Thus several generalisations are possible, some of which may be false. For example in Fig. 1, we might generalise to 'all triangles', 'all right angled triangles', or 'all similar triangles[2] to the one drawn'. Often there is a clear intuitive generalisation, but this must be formalised if we are to give rigorous proofs (c.f. [9] for a detailed discussion of this). So diagrams require a fresh approach to all three problems.

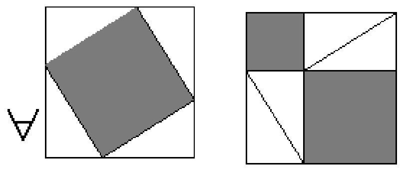

Fig. 1. Problems in Pythagoras' Theorem: It is not clear what the $\forall$ symbol applies to, and we have not specified which aspects of the triangle are important.

1.1 Related work

Several approaches have tackled the problem of quantification in diagrams by introducing new notation. This can produce systems as powerful as predicate logic. However, it generally leads to more complex diagrams, and great care is required if these are to retain their intuitive feel. For example, in 1976 Schubert, starting from semantic nets, developed (by adding more and more notation) a diagrammatic representation that is as expressive as modal lambda calculus (see Fig. 2) [12]. Unfortunately, the resulting diagrams are extremely difficult to read and, to the best of our knowledge, were not generally used.

[2] Two triangles are similar if they differ only in scale or left-right orientation.

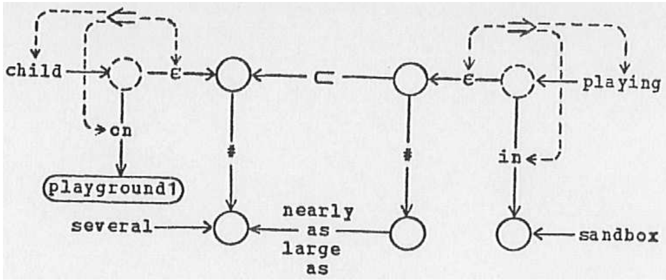

Fig. 2. A Schubert diagram for "Several children are on the playground. Most of them are playing in the sandbox."

A more recent example is [8], which extends Venn diagrams with extra notation and some inference rules to give a heterogenous system[3] as expressive as first-order predicate logic. It is shown to be sound and complete, but also seems too difficult to use. As Ambrose Bierce's inventor said: "I have demonstrated the correctness of my details, the defects are merely basic and fundamental" [2]. By extending diagram systems on purely logical criterion without considering ease-of-use issues, the final systems lost the very qualities that make diagrams attractive.

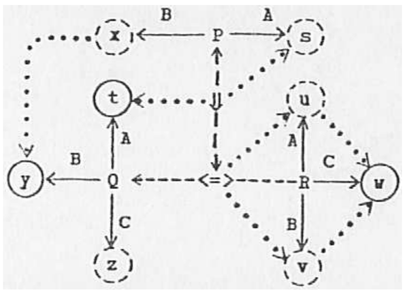

Fig. 3. We are unsure what this Schubert diagram means.

A more successful diagrammatic system with quantifiers is Gil *et al*'s *constraint diagrams*, which are an extension of *spider diagrams* [5]. The objects in spider diagrams are oval contours (representing sets with overlap=intersection), points (representing members of sets) and 'spiders' (linked points representing statements of the form $x \in A \cup B$). Constraint diagrams introduce arrows (representing relationships) and quantifiers. Only points and spiders are quantified over. Quantifier type is handled by drawing points in different ways. That is, they have different primitive objects for the different quantifiers: $\bullet$ = $\exists$point,

[3] A system whose representations mix diagrams and text [1].

$\star = \forall$points. Quantifier ordering problems are dealt with by labelling quantifiers with numbers.

The solutions in [5] for handling quantification have similarities to the ones we will present here. However the abstract indirect nature of constraint diagrams makes them 'closer' to algebra, and thus the problems are easier than for more specific diagrams (c.f. §2.2).

The generalisation problem arises only in a limited form for [5]. This is because, except for set membership, all conditions must be made explicit – as in algebra. Set membership conditions are handled by fixing the interpretation: a point x is interpreted as belonging to the smallest region containing it. 'Spiders' provide a means for over-riding this default interpretation. We shall look at how this idea of a default reading with extra syntax to give other readings can be extended to cover diagrams where a wide range of conditions can be represented.

Using different objects to represent quantifier type would not be suitable for our analysis diagrams, where we variously wish to quantify over points, sets, functions, lengths, etc., as it would involve introducing multiple primitives for each type of object. This would quickly get confusing.

2 Prerequisites: some definitions

2.1 Real analysis

Our work is in the domain of real analysis (which gives a rigorous underpinning for calculus, and leads on to fields such as topology). Analysis is a form of geometry, but one whose dry algebraic formalism can make it hard to learn. This makes it an attractive area for applying diagrammatic reasoning. We have implemented a prototype interactive theorem prover using a diagrammatic logic [14]. Our aim is to produce a teaching system based on this, so issues of comprehension and understanding are paramount.

We follow Cauchy's $\epsilon - \delta$ analysis (also known as *standard analysis*), based on arbitrarily small error terms [11]. Definitions often involve arbitrarily small open balls. We write $B_r(x) = \{x' : |x - x'| < r\}$ for the ball of radius r with centre x (this notation is common but not universal). Our examples in this paper will be based on open sets, which are defined as follows:

Definition 21. If X is open... $\mathtt{open}(X) \Rightarrow \mathtt{set}(X) \wedge \forall x \in X, \exists \epsilon > 0. B_\epsilon(x) \subset X$

Definition 22. X is open if... $\mathtt{set}(X), \forall x \in X, \exists \epsilon > 0. B_\epsilon(x) \subset X \;\Rightarrow\; \mathtt{open}(X)$

2.2 Diagrams

Diagrammatic representations are by their nature quite specific, however the level of specificity varies. We introduce the term *direct* to informally describe the degree of this. A more direct diagram is one where the representation used is closely linked to its meaning. For example drawing a triangle to reason about triangles. By contrast, an indirect diagram is one where the relation between sign and meaning is arbitrary, and based on convention. Constraint diagrams are an example of this, where a dot can represent anything from a spatial point to a person. Textual representations are always indirect.

In general, the closer the link between signifier and signified,[4] the more specific the representations are (i.e. more direct diagrams tend to be more specific). Specific representations lead to the generalisation problem outlined in §1. Also, it seems that the more specific the representations are, the harder it is to properly perform universal quantification. This is because there is extra information that the user must ignore. For example, it is easier to reason with *'let X represent any man...'* than *'let the late Jon Barwise, who had fading brown hair and contributed so much to diagrammatic reasoning, represent any man...'*. Nevertheless, specific representations do seem to have strong advantages. In particular, more direct diagrams give representations for geometric objects which are both very natural, and seem to lend themselves well to diagrammatic reasoning. See [7] or [13] for discussions of this issue. Our domain is in geometry, hence we have adopted a system based on fairly direct diagrams. This gives us quite natural representations for many of the objects in the domain, but makes quantification a difficult issue.

Our diagrams consist of labelled graphical objects with relations between them. Relations may be represented either graphically or algebraically (this makes our representations heterogenous, although we will continue to refer to them as diagrams).

2.3 Proof system

Often diagrammatic reasoning is presented as a question of interpreting static diagrams. Here we consider dynamic diagrammatic reasoning, where the process of drawing is important, as opposed to just the finished diagram.

Our logic is defined using redraw rules, which are similar to rewrite rules but transform diagrams rather than formulae. This reflects our belief that diagrammatic reasoning is often linked to the drawing process, rather than just the finished diagram. These rules are expressed diagrammatically by an example transformation. A *simple redraw rule*, $D_0 \hookrightarrow D_1$, consists of an initial diagram (D_0, the antecedent or pre-condition) and a modified diagram (D_1, the consequent, or post-condition). Fig. 4 gives an example redraw rule.

Theorems are stated as rules rather than statements (e.g. $sin^2\theta + cos^2\theta \Rightarrow 1$, $1 \Rightarrow sin^2\theta + cos^2\theta$ rather than $\forall\theta. sin^2\theta + cos^2\theta = 1$), and are also expressed diagrammatically (i.e. as redraw rules). A proof consists of a demonstration that the antecedent of the theorem can always be redrawn to give the consequent diagram using an accepted set of rules (i.e. the axioms). Hence a proof is a chain of diagrams, starting with the theorem antecedent and ending with the theorem consequent. We refer to an incomplete or complete proof as a *reasoning chain*.

Informally, the procedure for applying a simple rule is:

[4] A *signifier* is the method (e.g. a word or picture) used to represent a concept (the *signified*); together they make up a *sign* [4].

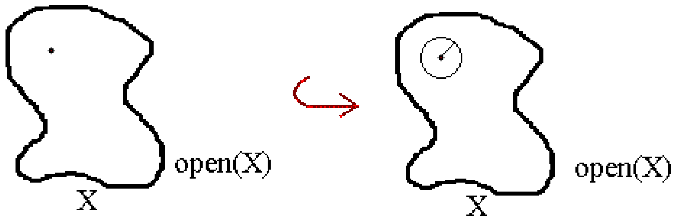

Fig. 4. Definition 21 as a redraw rule.

1. The antecedent diagram is matched with some part of the working diagram (i.e. the last diagram in the reasoning chain).
2. The working diagram is modified in an equivalent way to the modification between the antecedent and consequent diagrams. This modified diagram is added to the end of the reasoning chain.

The principal differences from rewrite rules are:

– There can be an infinite number of valid (but equivalent) redrawings for a given diagram, a given rule and a given matching (e.g. a rule may specify that a point should be drawn, but leave open the choice of which point to draw).
– Due to the problem of multiple possible generalisations, there is no clear choice for how the matching algorithm should work.

Fig. 4 shows how Definition 21 can be implemented as a redraw rule. The antecedent will match any point y in any open set Y; the consequent guarantees the existence of a ball $B_\epsilon(y) \subset Y$.

Consider implementing the converse rule (Definition 22). This definition can be read as 'X is open if, given any point x in X, we can find an $\epsilon > 0$ such that$B_\epsilon(x) \subset X$'. Note the verb *'we can find...'* – this condition can be thought of as dynamic: it gives a type of behaviour which we must demonstrate to show that X is open (by contrast, the conditions in Definition 21 can be thought of as *adjectival*). Static diagrams are not well suited to representing behaviour. They are better suited to adjectives than verbs. Instead, we introduce *animated redraw rules*. An *animation* here is a chain of diagrams. Animated redraw rules have an animation as their pre-condition. Where simple redraw rules match the last diagram in the reasoning chain, animated rules must match a section of the reasoning chain.[5] Fig. 5 gives an example redraw rule with an animated antecedent.

[5] The full system includes two further types of rule for case-split introduction and case elimination.

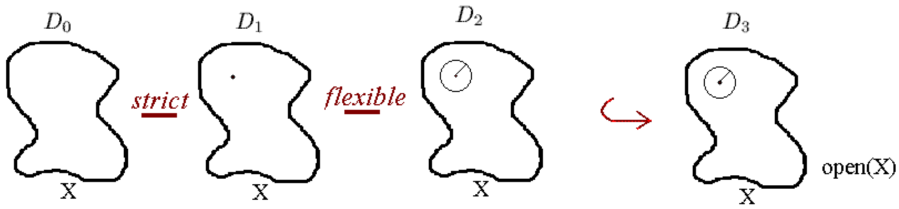

Fig. 5. Definition 22 as an animated redraw rule. The terms *strict* and *flexible* are explained in section 3.3.

3 Using animation for quantification

3.1 Quantifier hierarchy

As with sentential reasoning, quantifier order can be important. Animation gives a reliable and intuitive ordering without introducing extra notation. This is because of causality: it is obvious that object A cannot depend on object B if B was drawn after A. Fig. 6 gives an example based on the joke "Every minute, somewhere in the world a woman gives birth. We must find this woman and stop her." This shows how animation in a very simple way eliminates the ordering ambiguity which allows two conflicting interpretations of the first sentence. Structurally this is equivalent to quantifier numbering in [5], although in presentational terms it is very different.

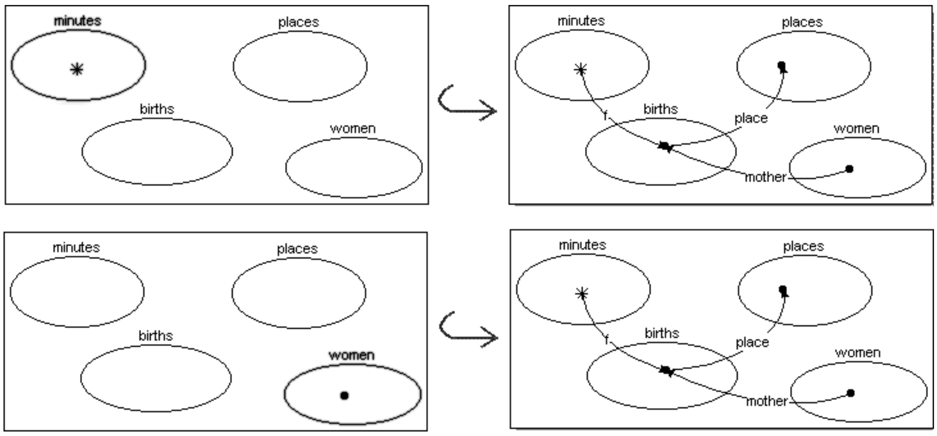

Fig. 6. 'For any minute, there is a woman who gives birth' vs. 'There is a woman who gives birth every minute'. The second diagram is identical for both cases, but the starting diagram shows which object comes first.

3.2 Generalisation

In our logic, a matching algorithm allows redraw rules to be applied to a wide range of diagrams. Thus the matching algorithm determines generalisation of the rule (and vice-versa: specifying generalisations would establish a matching criterion) [14]. As discussed in §1, there are often several possible generalisations, hence several matching algorithms are possible. It seems unlikely that there will be a canonical answer to the question of what aspects of a diagram should be read as generalisation conditions.

We could simply make all the relevant information explicit in the diagram, and assume everything else is unimportant. However this would make for cluttered, less legible, diagrams. A more sensible approach is to have a default interpretation that certain aspects of a diagram are assumed to be important. Ideally, this should be the same as the intuitive reading of the diagram. The conditions specified by this default interpretation can be strengthened or relaxed, but only through explicit conditions in the diagram. Spider diagrams show how this can be used for representing set membership, where spiders are used to override the default reading. It can be extended to cover other relations. Some of the default readings we use are:

- Lengths are considered unimportant (to be generalised), unless a statement of the form $\texttt{length}(A) < \texttt{length}(B)$ or $\texttt{length}(\texttt{A}) = \texttt{B}$ is added.
- Right angles are considered salient,[6] unless the angle is tagged with a $\measuredangle$ symbol indicating an aribtrary angle.

3.3 Quantifier behaviour

Statements in our logic are expressed as rules, hence the question of 'how do quantifiers behave?' becomes 'when should a rule antecedent match a reasoning chain?' That is, the question of what does a diagram/animation mean is recast as 'what diagrams/reasoning chains does it match?'

We have to be careful working in direct diagrams, as a quantified object is also a specific example.[7] For example, when reasoning about an abstract universally quantified point, we must nevertheless draw a particular point, and this point will have properties that do not hold universally. However, as long as such properties are not used in the reasoning that follows, they will not affect the generality of the proof. The reasoning that follows would work for any point, so it does not matter which point was actually drawn. The specific case that is drawn comes to represent a class of equivalent cases. What matters is that the reasoning is generic (with indirect representations, this is automatically enforced by using generic objects; with direct representations the generality of the reasoning must be checked).

Consider again Fig. 5, where there is a universally quantified point x in the middle of the rule antecedent. Suppose we wish to apply this rule to show that

[6] Assumed to be an intentional feature and therefore not to be generalised.

[7] To be precise, it is interpreted as a specific example.

the set $Y = B_1((0,0))$ is open. First we introduce an arbitrary point $y \in Y$ to match the point x in the rule antecedent. We still have further reasoning to do before the rule will match: we have to find an ϵ-ball about y that lies within the set Y. The reasoning that follows must be universally applicable, which means that it must not use the specific nature of the point y, only the fact that $y \in Y$. For example, suppose we concluded $B_{0.1}(y) \subset Y$ from $y = (0.7, 0.8)$. The reasoning is sound, but it does not apply to other values of y. Our chain of reasoning finds an ϵ for $y = (0.7, 0.8)$, but this reasoning could not be applied to any point. Hence the rule – which requires that such an ϵ exists for any point – is not applicable If the reasoning that follows draws on non-universal aspects of the point, then we say that the point y has been *compromised*; it is no longer universally quantified. This is simple to check: an object is compromised if later reasoning alters its generalisation (by adding extra conditions).

This leads us to the following method for reliably enforcing generic reasoning: suppose an animated redraw rule has the antecedent $D_0 - D_1 - ... - D_n$, where the D_i are diagrams. When a universally quantified object is introduced into the proof, it must be done exactly as shown in the rule. The interpretation of the object introduced into the reasoning chain must be equivalent to the interpretation of the object introduced in the rule antecedent. If diagram D_i introduces a universally quantified object, we call the transition $D_{i-1} - D_i$ a *strict* transition, since it will only match a transition $P_j - P_{j+1}$ if $P_j - P_{j+1}$ shows equivalent modifications to $D_{i-1} - D_i$ *and no other modifications* (i.e. no extra constructions or conditions). Moreover, subsequent reasoning on P_{j+1} must not compromise the new universally quantified object drawn. This ensures that the reasoning that follows will be as general as the rule requires.

When the rule antecedent contains an existentially quantified object, all that must be shown is that some matching object can be constructed in the reasoning chain. How this is done does not matter (as long as it does not compromise a universally quantified object). Hence an existentially quantified object can be drawn in any manner using several redraw operations, since all we require for the rule antecedent to match is that some such object exists. If diagram D_j introduces an existentially quantified object we call the transition $D_{j-1} - D_j$ a *flexible* transition. A flexible transition allows arbitrary other constructions to be drawn in the reasoning chain when moving from one diagram in the rule to the next.

For example, to prove the theorem $\texttt{open}(B_r(x))$ takes 11 steps in our logic. A sketch of this proof is given in Fig. 7. We start with the set $B_r(x)$ (diagram P_0 in Fig. 7). The first step is to introduce an arbitrary point in $B_r(x)$ to match the universally quantified point in diagram D_1, Fig. 5. It then takes three steps to construct a suitable ϵ-ball (P_5 in Fig. 7) and five more steps to show that it lies inside $B_r(x)$ (diagram P_{10}). All these steps are performed using simple redraw rules. The final step is to apply the animated rule shown in Fig. 5. Let $P_0 - ... - P_{11}$ be the proof. Then D_0 matches P_0, D_1 matches P_1 and D_2 matches P_{10}, as shown in Fig. 8.

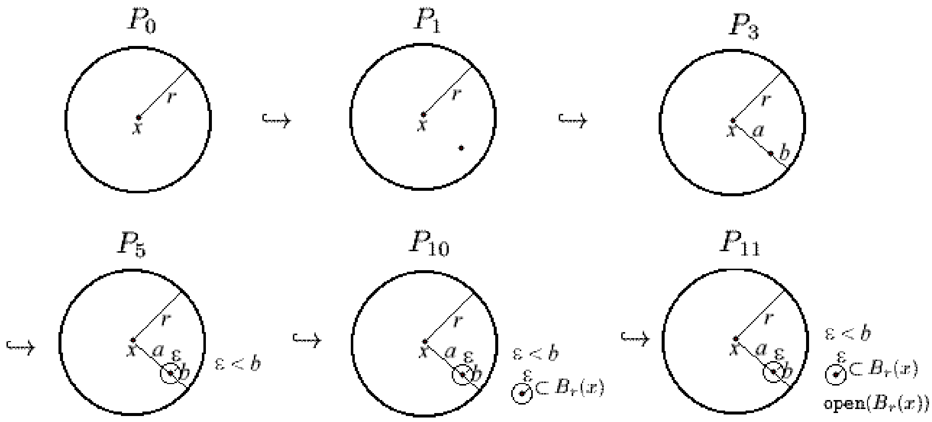

Fig. 7. Sketch proof for $\mathtt{open}(B_x(r))$. Space limitations prevent us giving the full proof or the rules used.

$$\begin{array}{lllllll}
D_0 - \mathit{strict} - D_1 - \mathit{flexible} - D_2 & \hookrightarrow & D_3 \\
\downarrow \qquad\qquad\quad \downarrow \qquad\qquad\qquad\quad \downarrow & & \downarrow \\
P_0 \quad \hookrightarrow \quad P_1 \hookrightarrow \quad \ldots \quad \hookrightarrow P_{10} & \hookrightarrow & P_{11}
\end{array}$$

Fig. 8. Antecedent matching with strict and flexible transitions.

The method described above syntactically prescribes quantifier type in rule antecedents in terms of the matching criterion regarding transitions.

3.4 Quantification in the rule consequent

Rule consequents are always a single diagram containing new objects or new conditions.[8] New objects are assumed to be existentially quantified. More complex inferences can be expressed in this formalism by two rules linked with a syntactic tag. For example, the statement $p \Rightarrow \exists x.\forall y.q(x,y)$ would be converted into two rules: $p \Rightarrow \exists x.r(x)$ and $r(x), y \Rightarrow q(x,y)$, where r is the syntactic tag created to link the two.

3.5 Representing quantifier type

A side-benefit of animation is that it clears up labelling ambiguities. When dealing with emergent or composite objects (i.e. objects formed as a result of drawing other objects, such as some of the squares in Fig. 1), it is not necessarily clear what object a label applies to. However if objects are introduced one at a time

[8] If we allowed objects/conditions to be deleted or changed it would give a non-monotonic logic.

with their labels, then this ambiguity vanishes. In our logic we require a reasoning step to recognise a composite object, so composite objects are 'drawn' (and labelled) after their parts.

With labelling ambiguity removed, we could simply reinstate the algebraic symbols $\forall, \exists$ and represent quantifier type by labelling objects with them. Diagrams also allow other, potentially more interesting, possibilities. These include some form of drawing convention, such as colour-coding or using different shaped objects. Or – since quantifiers are introduced one at a time – quantifier type can be represented by having different transitions between frames in the animation.

Any of these representation methods would be sufficient to distinguish the two quantifier types, but they have different advantages. Using the established symbols gives the user something they may already be familiar with. Colour-coding is 'cleaner' since it does not introduce extra labelling, and this may aid comprehension.

Both of the above methods rely purely on convention for their meaning. However, since quantifier behaviour (i.e. their syntactic meaning) comes from the diagram transitions, we could also represent quantifiers by labelling the transitions. A more interesting option is to use special transitions that can attempt to convey the difference in meaning. These special transitions are animations of a different kind. They are independent of the reasoning rather than a part of it. For example, a universally quantified point could 'roam it's habitat', indicating that it is not a specific point. With an existentially quantified object in a rule antecedent, the transition could indicate 'miscellaneous drawing' to illustrate that, when applying the rule, other unspecified constructions will be necessary at this stage. Such an approach would not be of interest to professional users, but could be helpful in teaching applications. However the quantifier type is not visible in the final diagram. To a certain extent, the strengths and weaknesses of these representation methods complement each other, and so they can be combined. We currently use a combination of colour-coding and special transitions, although in the future the system will be customisable to a user's preferences. Colour-coding is identical at the syntactic level to the different primitive objects used in [5], plus it can be used uniformly across types of object that are drawn in quite different ways – although it does restrict the use of colour for representing other properties.

4 Open issues

Consider the rule in Fig. 5. The natural way of using this rule requires at least one point in the set - so it cannot be applied to the empty set. This introduces a dilemma: our definition of an open set is slightly different from the standard one in that it excludes the empty set. The cause of this discrepancy is that our diagrammatic universal quantifier has existential import. That is, the statement $\forall x \in X$ implies $\exists x \in X$. This is also true in aristotelian logic [10] and natural language, but of course false in predicate calculus. Currently we handle the empty set as a special case. However, since correspondence with conventional logic is desirable in mathematical domains, this is not ideal.

5 Converting animated rules into quantified rules

So far we have described how sentential rules correspond to animated diagrammatic rules. Now we look into how animated diagrammatic rules correspond to the sentential ones. The concept of well-formed formulae (wff) can be translated to diagrams. We assume the following loose definition of a well formed diagram (wfd) here:

Definition 51. Suppose we have a diagram language L consisting of graphical objects, labelling constants and first-order predicates, and for each property that can be inferred from the diagrams, there is a corresponding predicate in L (i.e. any diagram in L can also be described purely algebraically in L). Then say:

- The empty diagram is a wfd.
- If A is a wfd, X an object within L, X has label l_X and l_X is not already used in A, then $A \cup X$ is a wfd.
- If A is a wfd, X some objects in A and $p(X)$ a predicate in L, then $A \cup p(X)$ is a wfd, where $p(X)$ could be drawn either graphically or algebraically using object labels.

Are any conjunctions of predicates allowed? Since we are using heterogenous diagrams, any combination can be stated, but there are sensible restrictions that could be made (e.g. disallowing $point(X) \wedge line(X)$).

5.1 Well formed formulae

Let us assume we have an interpretation function I that maps single diagrams to unquantified algebraic statements in a suitable domain. Let X denote a vector of variables, let p, q denote any conjunction of predicates in L (e.g. $\texttt{equal\_area}(X, Y)$, $\texttt{point}(X) \wedge \texttt{in}(X, Y)$, etc.)
Given diagrams D_0, D_1 the simple rule $D_0 \hookrightarrow D_1$, is well-formed if D_0, D_1 are wfd and $D_0 \subset D_1$. Simple rules correspond to statements of the form:

$$\lambda X.p(X) \;\Rightarrow\; q(X)$$

Here, $p(X) = I(D_0), q(X) = I(D_1) \backslash I(D_0)$ (with the natural mapping between labels and variables).
Given any animated antecedent $D_0 - ... - D_n$, we can add another diagram D_{n+1} to it in two ways:

1. With a flexible transition, introducing existentially quantified objects:
   If $A \;\Rightarrow\; B$ is a wff, $\texttt{var}(A)$ are the variables (free and bound) in A, and $X \cap \texttt{var}(A) = \emptyset$ then

$$A, \exists X.p(var(A), X) \;\Rightarrow\; B \text{ is a wff}$$

2. With a strict transition, introducing universally quantified objects:
If $A \ \Rightarrow \ B$ is a wff, $\mathtt{var}(A)$ are the variables (free and bound) in A, and $X \cap \mathtt{var}(A) = \emptyset$ then

$$A, \exists X'.p(var(A), X'), \forall X.p(var(A), X) \ \Rightarrow \ B \text{ is a wff}$$

5.2 Example animated redraw rule (Fig. 5) revisited

Following the correspondences given above and assuming the behaviour of interpretation function I, we get:

$$\begin{aligned}
I(D_0) &= \lambda X.\mathtt{set}(X) \\
I(D_0 - D_1) &= I(D_0), \exists x'.\mathtt{point}(x'), x' \in X, \forall x.\mathtt{point}(x), x \in X \\
I(D_0 - D_1 - D_2) &= I(D_0 - D_1), \exists \epsilon.\mathtt{real}(\epsilon)\epsilon > 0, B_\epsilon(x) \subset X \\
I(D_0 - D_1 - D_2 \hookrightarrow D_3) &= I(D_0 - D_1 - D_2) \Rightarrow \mathtt{open}(X)
\end{aligned}$$

Hence the redraw rule converts to the algebraic rule:

$$\mathtt{set}(X), X \neq \emptyset, \forall x \in X, \exists \epsilon > 0.B_\epsilon(x) \subset X \ \Rightarrow \ \mathtt{open}(X)$$

which is almost Definition 22. As explained in §4 we currently need a seperate redraw rule to cover the case $X = \emptyset$.

6 Conclusions & future work

Real analysis is a domain where great care is required to avoid mistakes, and where quantifier ordering is often important. As part of our project to produce a diagrammatic formalisation for this subject, we have developed rules with animated pre-conditions. This paper demonstrates how these rules work, showing how they allow us to perform generic quantified reasoning with specific representations. We hope that this has applications to other fields; any domain that considers dynamic behaviour seems promising. A second type of animation (special transitions for representing quanifier type) is also introduced to give more meaningful representations.

The treatment given in §5 is quite loose. We intend to develop a formal semantics for animated redraw rules, plus a formal definition of our interpretation function and matching criterion (whose behaviour we have assumed here). This should then allow us to show equivalence between redraw rules and algebraic rules.

There are drawbacks to our method of representing and reasoning about quantification. The issue of existential import raises serious questions. The use of colour to represent quantifier type severely limits the way in which colour can be used elsewhere in the diagram. Also, the extra dimension of time in the representations might prove to be harder for users because of 'overloading' understanding through extra demands on working memory. Perhaps the greatest drawback of animation is that it is not suited to being printed (e.g. in textbooks or papers), except as cumbersome comic strips where the simplicity of the representation is lost. Note that to a certain extent, this does not apply to its use on blackboards, where animation can be performed, albeit a little crudely.

However the advantages are a logic that is, we hope, more elegant and natural to use. Using animation to extend diagrams avoids extra labelling and should be more intuitive, since it draws on cause-and-effect for meaning rather than requiring conventions. Moreover rules with animated pre-conditions focus attention on the reasoning used in a proof. This could be beneficial from an educational point of view. Using extra notation it is possible to avoid the use of animation within our system. For example, Fig. 9 gives a non-animated version of the redraw rule in Fig. 5. We feel that by compressing all the information into one diagram, the non-animated version obscures the relations between the objects. We will test student responses to this difference as part of our system evaluation. We are also investigating how expressive our analysis diagrams are (i.e. how good a coverage of theorems we can achieve). Our preliminary work suggests that diagrammatic reasoning can be successful in teaching this domain [14].

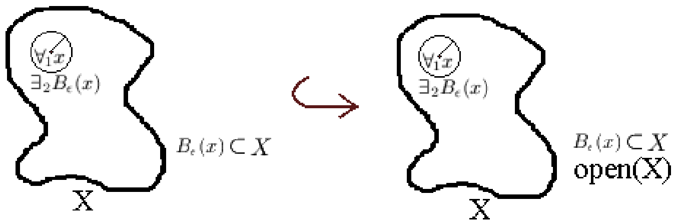

Fig. 9. Static redraw rule defining an open set.

References

1. J.Barwise & J.Etchemendy "Heterogeneous Logic" in Diagrammatic Reasoning: Cognitive and Computational Perspectives. AAAI Press/The MIT Press, 1995.
2. A.Bierce "Fantastic Fables: The Flying Machine" 1899. Project Gutenberg e-text 1995, ftp://ftp.ibiblio.org/pub/docs/books/gutenberg/etext95/fanfb10.txt
3. J.Howse, F.Molina & J.Taylor "On the Completeness and Expressiveness of Spider Diagram Systems" proceedings of the 1st International Conference on Theory and Application of Diagrams. pp26-41, Springer-Verlag, 2000.
4. B.Fraser "Structuralism" course notes, Cambridge University. Available online at http://www.classics.cam.ac.uk/Faculty/structuralism.html, 1999.
5. J.Gil, J.Howse & S.Kent "Towards a Formalization of Constraint Diagrams" Symposium on Visual Languages and Formal Methods. IEEE, 2001.

6. J.Gil & Y.Sorkin "The Constraint Diagrams Editor" Available online at http://www.cs.technion.ac.il/Labs/ssdl/research/cdeditor, 1999 – 2000.
7. C.Gurr, J.Lee & K.Stenning "Theories of Diagrammatic Reasoning: Distinguishing Component Problems" in Minds & Machines 8(4) pp533-557, Kluwer Academic, 1998.
8. E.Hammer "Reasoning with Sentences and Diagrams" in "Working Papers on Diagrams and Logic" editors G.Allwein & J.Barwise, Indiana University Logic Group, 1993.
9. P.J.Hayes & G.L.Laforte "Diagrammatic Reasoning:Analysis of an Example" in "Formalizing Reasoning with Visual and Diagrammatic representations: Papers from the 1998 AAAI Fall Symposium" editors G.Allwein, K.Marriott & B.Meyer, 1998.
10. G.Klima "Existence and Reference in Medieval Logic" in "New Essays in Free Logic" editors A.Hieke & E.Morscher, Kluwer Academic, 2001.
11. T.W.Körner "Analysis" lecture notes, Cambridge University. Available online at http://ftp.dpmms.cam.ac.uk/pub/twk/Anal.ps, 1998.
12. L.K.Schubert "Extending the Expressive Power of Semantic Networks" in Artificial Intelligence. 7(2) pp163-198, Elsevier, 1976.
13. A.Sloman "Interactions Between Philosophy and A.I.: The Role of Intuition and Non-Logical Reasoning in Intelligence" proceedings 2nd International Joint Conference on Artificial Intelligence, 1971.
14. D.Winterstein "Diagrammatic Reasoning in a Continuous Domain" CISA Ph.D. Research Proposal, Edinburgh University. Available online at http://www.dai.ed.ac.uk/homes/danielw/academic, 2001.