A Theorem Prover for a Diagrammatic Blocks World

Michael Wollowski

Computer Science and Software Engineering Department
Rose-Hulman Institute of Technology
Terre Haute, IN 47803, USA
Overview

• Introduction
• Syntax of diagrams
• Types of proofs
• Sample consequence proof
• Primary rules of inference
• Implementation
• Conclusions
Introduction

• Developed a system in which to use diagrams to give proofs in the blocks world.
• Proven the system sound and complete
• Thereby shown that one can reason validly with diagrams
• Implemented it, showing that this reasoning can be automated
Sample Diagram

move-size(1)     table-size(2)
Diagrammatic Symbols

• Block Square: 

• Table line: 

• Situation connectors:  ->  …

• Boundary rectangle:
Well-Formed Frame

- Exactly one boundary rectangle
- Exactly one table line drawn horizontally near the bottom of the boundary rectangle
- Block squares are drawn on the table line or squarely on top of each other. They do not overlap with each other or the boundary rectangle.
- Each block square contains exactly one block constant in a non-overlapping fashion.
- No two block squares are labeled by the same block constant
Types of Proofs

- Consistency
- Non-consequence
- Consequence
- Inconsistency
Consistency Proofs

• Used to show that the given information is consistent
• This means that there is a plan
• Traditional planning problems fit this category best
• In traditional blocks world, a solution can always be found, but an optimal solution is desired, as such, it is more complex than a consistency proof
Consequence Proofs

• Used to show that some information follows from the given information
• Something holds for all plans
• Most complex proof
• Detailed example momentarily
Non-Consequence Proof

• Used to show that some information does not follow from the given information
• This means that there is an alternative plan
• With this proof, we show that a counterexample is consistent, hence, it is a consistency proof
Inconsistency Proofs

• Used to show that some information is inconsistent
• This means that there is no plan
• It is a consequence proof in which no path leads from the start state to the final state, such that it satisfies all constraints.
Sample Consequence Proof

• Let this diagram be the given information

move-size(1)  table-size(2)
Sample Consequence Proof

- Suppose we want to show that the following information is a consequence.
- It states that in each plan we have to move B onto of C.

```
<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td></td>
</tr>
</tbody>
</table>
```

move-size(1)  table-size(2)
Abbreviated Proof
Rules of Inference

• \textit{Given}. Accept the given information

• \textit{Move}. A frame is added, and linked by an arrow

• In the new frame, either a single block or a single tower of blocks has been redrawn
Rules of Inference

• **Cases exhausted.** This rule is used to state that diagram which were derived by the “move” rule exhaust all possible moves which can be made

• This rule takes the sentential constraints into account

• In our system, we give proofs by cases
Rules of Inference

- *Cycle*. This rule is used to make explicit that there are cycles in this domain.
- Introduce a copy of a frame, link it by triple dots and label it the last such frame before the next frame.
Rules of Inference

• Close. Used to put the rule “cycle” to good use.
• If a frame has been labeled “last” and an identical frame appears between it and the referenced frame, then the diagram can be closed.
Representing Diagrams

- Towers of blocks are stored as lists.
- `(define-structure (frame number info last next))`
- `(make-frame 'f8 '((a b) (c)) nil nil)`
Implementation

• Rules of inference are a direct implementation of the ones presented.
Conclusions

• Well-formed definitions of diagrams can be given
• Rules of inference can be shown sound and complete
• Diagrams are designed for particular domains
• As such a diagrammatic system is a special-purpose system