Using a Blackboard Architecture for Assertion Application in Proof Planning

Christoph Benzmüller, Andreas Meier, and Volker Sorge

Universität des Saarlandes, Saarbrücken, Germany
Proof Planning in the OMEGA System:

- considers theorem proving as planning process
- employs methods as planning operators
- applies mathematical facts stored in data base: axioms, theorems, lemmas (so-called assertions)

During proof planning:

Methods access data base to look up and apply Assertions:

\[
\begin{align*}
\frac{Prem \text{s}}{Goal} & \quad \text{Assertion}
\end{align*}
\]
Example

Classifying residue class structures wrt.

- algebraic structure they form (semi-group, monoid etc.)
- isomorphic structures

Proof obligations, e.g., \( \text{Closed}(\mathbb{Z}_5, \lambda x. \lambda y. (x \ast y) \oplus 3) \)

- \( \mathbb{Z}_5 \): set of integer congruence classes modulo 5
- \( \lambda x. \lambda y. (x \ast y) \oplus 3 \): binary operation on \( \mathbb{Z}_5 \)

One approach: apply known theorems from data base
Example

\[\text{ClosedConst} : \forall n : \mathbb{Z} \cdot \forall c : \mathbb{Z}_n \cdot \text{Closed}(\mathbb{Z}_n, \lambda x \cdot \lambda y \cdot c)\]

\[\text{ClosedFV} : \forall n : \mathbb{Z} \cdot \text{Closed}(\mathbb{Z}_n, \lambda x \cdot \lambda y \cdot x)\]

\[\text{ClosedSV} : \forall n : \mathbb{Z} \cdot \text{Closed}(\mathbb{Z}_n, \lambda x \cdot \lambda y \cdot y)\]

\[\text{Closed}\uparrow : \forall n : \mathbb{Z} \cdot \forall op_1 \cdot \forall op_2 \cdot (\text{Closed}(\mathbb{Z}_n, op_1) \land \text{Closed}(\mathbb{Z}_n, op_2)) \Rightarrow \text{Closed}(\mathbb{Z}_n, \lambda x \cdot \lambda y \cdot (x \ op_1 \ y)\uparrow(x \ op_2 \ y))\]

\[\text{Closed}\downarrow : \forall n : \mathbb{Z} \cdot \forall op_1 \cdot \forall op_2 \cdot (\text{Closed}(\mathbb{Z}_n, op_1) \land \text{Closed}(\mathbb{Z}_n, op_2)) \Rightarrow \text{Closed}(\mathbb{Z}_n, \lambda x \cdot \lambda y \cdot (x \ op_1 \ y)\downarrow(x \ op_2 \ y))\]

\[\text{Closed}^\ast : \forall n : \mathbb{Z} \cdot \forall op_1 \cdot \forall op_2 \cdot (\text{Closed}(\mathbb{Z}_n, op_1) \land \text{Closed}(\mathbb{Z}_n, op_2)) \Rightarrow \text{Closed}(\mathbb{Z}_n, \lambda x \cdot \lambda y \cdot (x \ op_1 \ y)^\ast(x \ op_2 \ y))\]
Determining applicable assertions can be difficult

Idea: Separate search for applicable assertions from main proving process

Addressed aspects

- **Concurrency**: parallelizing applicability check to gain efficiency and any-time behavior

- **Flexibility**: parameterize applicability check to employ, for instance, different matching procedures

- **Robustness**: become independent from data base details such as theorem/theory names
Realization – The Idea

- Form clusters of related theorems
- Use two filters (simple & complex)
- Dynamically extend mechanism for new assertions
Realization – The Idea

- Employ the hierarchical blackboard architecture \( \Omega\)-ANTS
- In-built concurrency
- Enables cooperation of knowledge sources (so-called agents)
Realization – The Idea

Interactive User
and/or Selector

Suggestions
- PAI–CallOtter–1
- PAI–=Subst–2
- PAI–ForallE–3
- PAI–AndI–2

AndI
- PAI–AndI–1
- PAI–AndI–2

ForallE
- PAI–ForallE–1
- PAI–ForallE–2
- PAI–ForallE–3

=Subst
- PAI–=Subst–1
- PAI–=Subst–2

CallOtter
- PAI–CallOtter–1

Partial Proof

© Benzmüller, Meier, Sorge, MKM 2001, Linz – p.5
Applicability check performed in three stages:
Applicability check performed in three stages:

- check for suitable goal (first filter)
Applicability check performed in three stages:

- check for suitable goal (first filter)
- try to match theorems (with several algorithms in parallel)
Applicability check performed in three stages:

- check for suitable goal (first filter)
- try to match theorems (with several algorithms in parallel)
- search for possible premises of an applicable theorem
Applicability check performed in three stages:

- Check for suitable goal (first filter)
- Try to match theorems (with several algorithms in parallel)
- Search for possible premises of an applicable theorem

Formation of clusters done automatically

⇒ Special predicate to acquire theorems
Example (continued)

\[ \mathcal{G}\{G\},\{T,P\} = \{ G: G \text{ contains the } \text{Closed} \text{ predicate} \} \]

\[ \mathcal{F}\{T\},\{G\},\{P\} = \{ T: \text{Conclusion matches } G \text{ with first order matching} \} \]

\[ \mathcal{F}\{T\},\{G\},\{P\} = \{ T: \text{Conclusion matches } G \text{ with special algorithm} \} \]

\[ \mathcal{G}\{P\}, \{G,T\},\{\} = \{ P: \text{The nodes matching the premises of } T \} \]
Example (continued)

Goal: \( \text{Closed}(\mathbb{Z}_5, \lambda x. \lambda y. (x \cdot y) + 3_5) \)

Goal contains \textit{Closed} predicate?
Example (continued)

Goal: \( \text{Closed}(\mathbb{Z}_5, \lambda x. \lambda y. (x \cdot y) + 3) \)

<table>
<thead>
<tr>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Goal: Closed(...))</td>
</tr>
</tbody>
</table>

\( \text{ClosedConst} : \text{Closed}(\mathbb{Z}_n, \lambda x. \lambda y. c) \)

\( \text{ClosedFV} : \ldots \lambda x. \lambda y. x \)

\( \text{ClosedSV} : \ldots \lambda x. \lambda y. y \)

with FO matching?

\( \text{Closed}^+ : \ldots \lambda x. \lambda y. (\ldots + \ldots) \)

\( \text{Closed}^- : \ldots \lambda x. \lambda y. (\ldots \mp \ldots) \)

\( \text{Closed}^\ast : \ldots \lambda x. \lambda y. (\ldots \ast \ldots) \)

with special algorithm?
Example (continued)

Goal: \( \text{Closed}(\mathbb{Z}_5, \lambda x. \lambda y. (x \ast y) \overline{\overline{3}^5}) \)

<table>
<thead>
<tr>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Goal:Closed(...))</td>
</tr>
<tr>
<td>(Goal: Closed(...), Thm:\text{Closed}\overline{\overline{+}})</td>
</tr>
</tbody>
</table>

Nodes matching the premises of Thm?
Example (continued)

Goal: \( \text{Closed}(\mathbb{Z}_5, \lambda x. \lambda y. \bar{3}_5) \)

Goal contains \( \text{Closed} \) predicate?
Example (continued)

Goal: \[\text{Closed}(\mathbb{Z}_5, \lambda x. \lambda y. \bar{3}_5)\]

<table>
<thead>
<tr>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Goal: Closed(\ldots))</td>
</tr>
</tbody>
</table>

\[\text{ClosedConst} : \text{Closed}(\mathbb{Z}_n, \lambda x. \lambda y. c)\]
\[\text{ClosedFV} : \ldots \lambda x. \lambda y. x\]
\[\text{ClosedSV} : \ldots \lambda x. \lambda y. y\]

with FO matching?

\[\text{Closed}^+ : \ldots \lambda x. \lambda y. (\ldots + \ldots)\]
\[\text{Closed}^- : \ldots \lambda x. \lambda y. (\ldots - \ldots)\]
\[\text{Closed}^* : \ldots \lambda x. \lambda y. (\ldots \ast \ldots)\]

with special algorithm?
Example (continued)

Goal: \( \text{Closed}(\mathbb{Z}_5, \lambda x. \lambda y. \bar{3}_5) \)

<table>
<thead>
<tr>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Goal: Closed(. . .))</td>
</tr>
<tr>
<td>(Goal: Closed(. . .), Thm: \text{ClosedConst})</td>
</tr>
</tbody>
</table>

Nodes matching the premises of Thm?
General Aspects

1. Interactive theorem proving
   – approach also supports interactive theorem proving
   – $\Omega$-ANTS:
     ranking and suggestion of applicable theorems to user

2. Retrieval from other data bases
   – approach not restricted to particular data base
   – also possible (not implemented yet):
     retrieval from distributed data base via www
Alternative use of $\Omega$-ANTS:

- one (or several) agent for each single theorem
- unique agent to search for matching supports
Discussion

Use of alternative techniques

- for knowledge base retrieval:
  hashing or term indexing techniques
Discussion

Use of alternative techniques

- for knowledge base retrieval:
  hashing or term indexing techniques

- for assertion matching:
  higher-order pattern matching
  higher-order (pre-)unification
  theorem proving
Future Work

Evaluation:

- compare our approach with some of the alternative techniques
- conduct a large case study
Future Work

Evaluation:

- compare our approach with some of the alternative techniques
- conduct a large case study

Integration:

- use our approach in combination with / as part of a more advanced knowledge base