Planning Diagonalization Proofs

Lassaad Cheikhrouhou
Fachbereich Informatik
Universität des Saarlandes
D-66041 Saarbrücken, Germany
lassaad@cs.uni-sb.de

1 Introduction

Proof planning [Bun91] is the search for a sequence of tactics (a proof plan) which can be applied to construct an object level proof. The used operators (methods) are specifications of tactics represented in a meta-language. A method specifies when the associated tactic can be applied and what its effects are. In our approach, we extend methods that represent well-known proof techniques, such as induction or diagonalization, with control knowledge and call them strategies.

The point of proof planning is to analyze proof techniques in order to determine their typical proof steps and to find a suitable control to perform these steps within the proof planning process. This paper \(^1\) is a first attempt of formalizing the diagonalization proof technique. After introducing the diagonalization principle in the next section, we describe how this strategy can be realized in the proof planning environment of ΩMEGA [BCF+97].

2 The Diagonalization Principle

In [DSW94] a proof by diagonalization is described as follows:

The diagonalization method turns on the demonstration of two assertions of the following sort:

1. A certain set \( E \) can be enumerated in a suitable fashion.
2. It is possible, with the help of the enumeration, to define an object \( d \) that is different from every object in the enumeration, i.e. \( d \notin E \).

We empirically studied some diagonalization proofs [Che96] to elaborate this proof technique:

- The first task is carried out by searching for a surjective function \( f \) from some set \( N \) into a certain set \( E \), the set to be enumerated. The indexing property of \( f \) \( [\forall x \in E (x) \rightarrow \exists y \in N (y \land x = f(y))] \) guarantees for each element of \( E \) the existence of an index in \( N \).
- The central point of diagonalization is the construction of the object \( d \), the diagonal element. On the one hand, it must be different from every object in the enumeration: For each element \( x \) of the set \( N \), the object \( f(x) \) must be different from \( d \). As \( d \) is a function, we achieve this by enforcing that the application of \( d \) to the element \( z \) (\( d(z) \)) differs from the diagonal term \( f(z)(z) \) in some property: On the other hand, the function \( d \) is defined in such a way, that it belongs to the enumerable set \( E \). Consequently, it has an index \( i \) (\( d = f(i) \)) and this leads apparently to a contradiction.

\(^1\) This paper is a short version of [Che97].
3 A Diagonalization Proof Strategy

In this section, we give a declarative representation of the diagonalization strategy and explain how it can be applied.

Representation of the Strategy

To represent the diagonalization proof strategy, we use the declarative framework for the representation of proof methods in [HKRS94] with some extensions. These extensions allow the reasoning with metavariables and the representation of some control knowledge within methods, e.g. the ordering of method subgoals. The diagonalization strategy is represented in Figure 1.

<table>
<thead>
<tr>
<th>Strategy : Diag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declarations</td>
</tr>
<tr>
<td>Precondition</td>
</tr>
<tr>
<td>Goal</td>
</tr>
<tr>
<td>Postcondition</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Proof Schema

1. $1 \vdash n(i) \land D = F(i)$  \hspace{5mm} (Hyp)
2. $1 \vdash n(i)$  \hspace{5mm} (AndEL 1)
3. $1 \vdash D = F(i)$  \hspace{5mm} (AndER 1)
4. $1 \vdash \bot$  \hspace{5mm} (MEC(D, IP) 2)

5. $\vdash E(D)$  \hspace{5mm} (Open)
6. $\vdash \forall x. \exists y. \bar{E}(x) \rightarrow \exists y. \bar{n}(y) \land x = \bar{F}(y)$  \hspace{5mm} (Open)
7. $\vdash \exists y. \bar{n}(y) \land D = \bar{F}(y)$  \hspace{5mm} (6 5)
8. $\vdash \bot$  \hspace{5mm} (ExistsE 7 4)

Fig. 1. The Diag strategy

The proof schema consists of ND (natural deduction) lines whose formulae are schematic, i.e. propositions with metavariables. Each ND line has a justification which can be either open (annotated by Open) or closed. A closed justification consists of a tactic, e.g. a ND rule, an argument list, and a list of ND lines (the premises). This means, the ND line can be proven from the premises by applying the tactic with the given arguments.

The goal of a strategy should match an open ND line at the object level, which is to be proven by the strategy. Before applying the strategy, its precondition

\footnote{Higher order metavariables (HOV), denoted in capital letters, can be instantiated with lambda expressions, whereas first order metavariables (FOV) can only be instantiated with constant symbols or FOVs. An over-lined metavariable can be bound to an object term with free variables, i.e. metavariables, and an un-annotated occurrence of a metavariable stands for a closed object term.}
must be fulfilled, where a precondition consists of ND lines from the proof schema
and constraints. The ND lines have to match support lines of the goal and the
constraints are evaluated. The support lines of an open ND line consist of its
hypotheses and their derived consequences.

The application of the strategy would reduce the goal to new subgoals to be
closed and additional constraints to be satisfied. Both subgoals and constraints
are given in the postcondition list. The ND lines and the constraints should be
considered sequentially from the left to the right, if they are separated by
commas in the list, and simultaneously by grouping them in a list marked with
∥.

Application of the Strategy
The strategy Diag can be used to prove a contradiction, i.e. ⊥. This strategy can
be chosen, among other methods with the same goal ⊥, either by the user or by
the control module of the planner which classifies available methods according
to additional information about the problem. The precondition list of Diag is
empty. Thus, it can directly be applied by considering its postcondition list:

The indexing property is determined by closing the subgoal 6, i.e. proving
the formula schema of the ND line 6 by assertion application from the support
lines. Hereby, the metavariables α, β, E, n, and F must be fully instantiated.
The next postcondition is the constraint newconst(γ, α) whose evaluation binds
the metavariable i to a new constant, the index of the diagonal element D.

The rest of the postcondition list has to be evaluated simultaneously and
leads to the construction of D. The function D belongs to the enumerable set
E and inverts some property wrt. the diagonal term F(i)(i). The first property
can be stated by closing the subgoal 5. The second property can be fulfilled by
some propositions which depend on β, the type of F(i)(i), and on the instantia-
tion of D. This is the reason why we represent the inverting property as a
constraint. The satisfaction of the inverts constraint would deliver the proofs
IP to the propositions that guarantee the inverting property of D. To prevent
nonsense instantiations of D, we use the restriction constraints: occurs(x,⊤(x)),
diffs(D, λxα, x), and diffs(D(i), F(i)(i)).

A vague specification of D can be given by D = λxα ⊤(F(x), x), and requir-
ing the inverting property ⊤(⊤(F(i), i))) ↔ ¬⊤(F(i)(i))). Some alternative
instantiations of these schemata, that we obtained by investigating examples in
[Che96], would make this task easier, as they provide more control:

1. The diagonal term F(i)(i) denotes a proposition: we consider the formula
   schema ¬F(i)(i) ↔ ⊤(i) as inverting property.
2. Otherwise: two important possible instantiations of D are distinguished:
   2.1. D(x) can be defined according to some condition U(F(x), x): D(x) equals
       Y(x), if U(F(x), x) holds, and it is Z(x) otherwise. Thus, D must be
       instantiated by the schema λxα if(U(F(x), x), Y(x), Z(x)), where the
       constraint differs(Y(x), Z(x)) must hold.

       The inversion of the term F(i)(i) is obtained, if we prove the sub-
       goals: ⊤(F(i)(i)) → ¬⊤(F(i), i), ⊤(Y(i)), ⊤(F(i)(i)) → ⊤(F(i), i), and
       ⊤(Z(i)).
2.2. After proving the subgoal 5, \( D \) can be instantiated with the function 
\[ \lambda x \alpha \cdot C(F(x), x) \): the inverting property of \( D \) can be either the inequality 
\[ C(F(i), i) \neq F(i)(i) \] or the formula schema \[ \exists \exists(C(F(i), i) \leftrightarrow \lnot F(i)(i)) \].

While closing subgoals represented by formula schemata, metavariables are
incrementally instantiated. This is done especially for HOVs by middle-out
reasoning [KBB93].

**Execution of the Strategy**
The instantiated proof schema of \( \text{Diag} \) is inserted into the ND proof. Lines, that
are not justified by ND rules, can be expanded further by applying their justification
tactics. The expansion of line 4 in the proof schema of \( \text{Diag} \) corresponds
to making the implicit contradiction of the diagonal element explicit. The tactic
\text{EC} generates a contradiction proof at ND level according to the instantiation
of \( D \) and of the proven properties that guarantee the inversion of the diagonal
term \( F(i)(i) \).

**References**

[BCF97] C. Benzmüller, L. Cheikhrouhou, D. Fehrer, A. Fiedler, X. Huang, M. Kerber,
M. Kohlhase, K. Konrad, E. Melis, A. Meier, W. Schärschmidt, J. Siek
mann, and V. Sorge. \textit{\Omegamega: Towards a Mathematical Assistant.} In
William McCune, editor, \textit{Proceedings of the 14th Conference on Automated

[Bun91] Alan Bundy. \textit{A Science of Reasoning.} In \textit{Computational Logic: Essays in
honour of Alan Robinson.} MIT Press, 1991. also presented at the 10th CADE
1990 as extended abstract.

[Che96] Lassaad Cheikhrouhou. \textit{The Mechanization of the Diagonalization Proof
Strategy.} SEKI Report SR-96-14, Fachbereich Informatik, Universität des
Saarlandes, Im Stadtwald, Saarbrücken, Germany, 1996.

[Che97] Lassaad Cheikhrouhou. \textit{Planning Diagonalization Proofs.} SEKI Report SR-
97-06, Fachbereich Informatik, Universität des Saarlandes, Im Stadtwald,
Saarbrücken, Germany, 1997.

[DSW94] Martin D. Davis, Ron Sigal, and Elaine J. Weyuker. \textit{Computability,
Complexity, and Languages: Fundamentals of Theoretical Computer Science.}

Mathematical Proofs with Methods.} Journal of Information Processing and

[KBB93] I. Kraan, D. Başın, and A. Bundy. \textit{Middle-Out Reasoning for Program
Synthesis.} In P. Szeredi, editor, \textit{Proceedings of the 10-th International