# Escape to Mizar from ATPs

Jesse Alama*

Center for Artificial Intelligence
New University of Lisbon
Portugal
j.alama@fct.unl.pt, http://centria.di.fct.unl.pt/~alama/

**Abstract**

We announce a tool for mapping E derivations to Mizar proofs. Our mapping complements earlier work that generates problems for automated theorem provers from Mizar inference checking problems. We describe the tool, explain the mapping, and show how we solved some of the difficulties that arise in mapping proofs between different logical formalisms, even when they are based on the same notion of logical consequence, as Mizar and E are (namely, first-order classical logic with identity).

## 1 Introduction

The problem of translating formal proofs expressed in different formats is an important research problem for automated reasoning. Proofs today come from many sources, and there are about as many implemented proof formats as there are different systems for interactive and automated theorem proving, not to mention the "pure" proof formats coming from mathematical logic. There is a choice about which axioms and rules of inference to pick. Even natural deduction comes in a number of shapes: Jáskowski, Gentzen, Fitch, Suppes... [17]. It seems likely that as the use of proof systems grows we will need to have better tools for mapping between different formalisms. This need has been recognized for a long time [26, 1], and it still seems we have some way to go. This paper discusses the problem of transforming derivations output by the E [20] automated theorem prover into Mizar texts.[1]

Mizar[2] is a language for writing mathematical texts in a "natural" style combined with a library of reasoning formalized in the Mizar language and verified by the Mizar proof checker. For the purpose of the present paper, the main feature of Mizar is its natural deduction-style proof language, grounded on a notion of "obvious inference" (to be explained below). We will ignore the large Mizar Mathematical Library (MML), an impressive collection going from the axioms of set theory to graduate-level pure mathematics. We will thus treat Mizar as a language and a suite of tools for carrying out arbitrary reasoning in first-order classical logic.

Related work is discussed in Section 2. Section 3 concerns the translation from E derivations to Mizar proofs. Because of the fine-grained level of detail offered by E and the simple multi-premise "obvious inference" rule of Mizar, the mapping is more or less straightforward, save for *skolemization* and *resolution*, neither of which have direct analogues in "human friendly" Mizar texts. Skolemization is discussed in Section 3.2 and our treatment of resolution is discussed

*Supported by the ESF research project *Dialogical Foundations of Semantics* within the ESF Eurocores program *LogICCC* (funded by the Portuguese Science Foundation, FCT LogICCC/0001/2007). Research for this paper was partially done while a visiting fellow at the Isaac Newton Institute for the Mathematical Sciences in the programme 'Semantics & Syntax'. Josef Urban inspired this project and provided many helpful suggestions. Artur Korniłowicz clarified some important details of Mizar proofs.

[1] Our work is available at https://github.com/jessealama/tptp4mizar.

[2] http://mizar.org

in 3.3. The problem of making the generated Mizar texts more humanly comprehensible is discussed in Section 3.4. Section 4 concludes and proposes applications and further opportunities for development. Appendix A is a complete example of a text (a solution to the Dreadbury Mansion puzzle found by E, translated to Mizar) produced by our translation.

## 2 Related work

In recent years there is an interest in adding automation to interactive theorem proving systems. An important challenge is to make sense, at the level of the interactive theorem prover, of solutions produced by external automated reasoning tools. Such *proof reconstruction* has been done for Isabelle/HOL [15]. There, the problem of finding an Isabelle/HOL text suitable for solving an inference problem $P$ is done as follows:

1. Translate $P$ to a first-order theorem proving problem $P^*$.
2. Solve $P^*$ using an automated theorem prover, yielding solution $S^*$.
3. Translate $S^*$ into a Isabelle/HOL text, yielding a solution $S$ of the original problem.

The work described in this paper could be used to provide a similar service for Mizar. It is interesting to note that in the case of Mizar the semantics of the source logic and the logic of the external theorem prover are (essentially) the same: first-order classical logic with identity. In the Isabelle/HOL case, at step (1) there is a potential loss of information because of a mismatch of Isabelle/HOL's logic and the logic of the ATPs used to solve problems (which may not in any case matter at step (3)). In the Mizar context, two-thirds (steps (1) and (2)) of the problem has been solved [19]; our work was motivated by that paper. Steps toward (3) have been taken in the form of Urban's ott2miz[3]. In fact, more than 2/3 of the problem is solved. Our work here builds on ott2miz by accounting for the clause normal form transformation, rather than starting with the clause normal form of a problem. Our translated proofs thus start with (the Mizar form of) the relevant initial formulas, which arguably improves the readability of the proofs. Moreover, our tool works with arbitrary TPTP FOF problems and TSTP derivations produced by E, rather than with Otter proof objects. The restriction to E is not essential; there is no inherent obstacle to extending our work to handle TSTP derivations produced by other automated theorem provers, provided that these derivations (proof objects) are sufficiently detailed, like E's. One must acknowledge, of course, that providing high-quality, fine-grained proof objects is a challenging practical problem for automated theorem provers.

To account for the clausal normal form transformation, one needs to deal with skolemization. This is a well-known issue in discussions surrounding proof objects for automated theorem provers [4]. Interestingly, our method for handling skolemization (to be described below) is analogous to the handling of quantifiers in the problem opposite ours, namely, converting Mizar proofs to TSTP derivations [24] in the setting of MPTP (Mizar Problems for Theorem Provers) [23]. There, Henkin-style implications are a natural solution to the problem of justifying a substitution instance $\varphi(a)$ of a formula given that its generalization $\forall x \varphi$ is justified. Our translation of skolemization steps is virtually the same as this; see Section 3.2 for details.

Exporting and verifying of Mizar proofs by ATPs has been carried out [24]. Such work is an inverse of ours since it goes from Mizar proofs to ATP problems.

[3] See its homepage https://github.com/JUrban/ott2miz and its announcement http://mizar.uwb.edu.pl/forum/archive/0306/msg00000.html on the Mizar users mailing list.

One can reasonably ask to what extent the derivation produced by E and the generated Mizar text are the same proof. We do not intend to enter into a discussion about the proof identity problem. For a discussion, see Došen [6]. Certainly the intension behind the mapping is to preserve the proof expressed by the E derivation. That the E derivation and the Mizar text generated from it are isomorphic will be clarified (but not proved) below. Mappings such as the one discussed in this paper can help contribute to a concrete investigation of the proof identity problem.

It is well-known that derivations carried out in clause-based calculi (such as resolution and kindred methods) tend to be difficult to understand, if not downright inscrutable. An important problem for the automated reasoning community for many years is to find methods of understanding machine-discovered proofs. One approach to this problem is to map resolution derivations into natural deduction proofs. Much work has been done in this direction [12, 13, 8, 7, 10, 11]. The transformations we employ are rather simple. To "clean up" the generated text, we take advantage of the various proof "enhancers" bundled with the standard Mizar distribution [9, §4.6]. These enhancers suggest compressions of a Mizar text that make it more parsimonious while preserving its semantics. In the end, though, it would seem that the judgment of whether an "enhanced" Mizar text is the best representative of a resolution proof is something that has to be left to the reader.

# 3 Translating E derivations into Mizar texts

To construct a Mizar text from a first-order TSTP derivation, one first identifies the function and predicate symbols of the derivation and creates an *environment* for the text. Constructing an environment for a Mizar text amounts to creating a handful of XML files specifying the syntax and semantics of the symbols appearing in the derivation. Normally, one does not create Mizar environments by hand from scratch but rather builds on some preexisting formalizations. Since we do not use the Mizar library, we cannot use the usual Mizar toolchain to construct an environment.

To generate the Mizar text, we exploit recent developments concerning the Mizar parser [2]. We generate XML representations (parse trees) of Mizar texts which can then be rendered as a plain-text Mizar file. The XML representation leaves open the possibility of further manipulation of the text through, e.g., XSL transformations.

The input to our procedure is an E derivation in TSTP format [22].

Section 3.1 discusses the overall organization of the generated proof. In Section 3.2 we discuss the skolemization problem. In Section 3.3 we discuss the problem of resolution.

## 3.1 Global and local organization of the proof

After the first batch of transformations, the refutation is "groomed" in the following ways:

1. Linearly order the formulas.

   In TPTP problems, the order of formulas is immaterial. However, in a natural deduction argument, the order of formulas in Mizar cannot be arbitrary. We topologically sort the input ordered in the obvious way (if conclusion $A$ uses formula $B$ as a premise, then $B$ should appear earlier than $A$) and work with a linear order.

2. Separate reasoning done among the input assumptions from reasoning done with the negation of the conjecture.

To capture the spirit of proof by contradiction we refactor E refutations into so-called diffuse reasoning blocks. We write:

```
theorem φ
proof
  now
    assume ¬φ;
    S1: ⟨conclusion 1⟩ by ...;
    S2: ⟨conclusion 2⟩ by ...;
    ...
    Sn: ⟨conclusion n⟩ by ...;
    thus contradiction by S_{a_1}, S_{a_2}, ..., S_{a_m}
  end;
  hence thesis;
end;
```

This concludes the discussion of the organization of the generated Mizar proof.

## 3.2 Skolemization

E's finely detailed proof output contains not simply the derivation of $\bot$ starting from the clause form of the input formulas. E can also record the transformation of the input formulas into clause normal formal. It is important to preserve these inferences because they give information about what was actually given to E.Accounting for skolemization a well-known issue in generating proof objects [4, 5]. The difficulty is that skolem functions are curious creatures in an interactive setting like Mizar's. Introducing a function into a Mizar text requires that the use can prove existence and uniqueness of its definiens. But what is the definiens of a skolem function, and how can it be justified?

Our solution to the skolemization problem is to introduce axioms. To take a simple example, suppose we have $\forall x \exists y \varphi$, and from this $\forall x \varphi[y := f(x)]$ is "derived". We introduce at this point a new definition (treated as an axiom) whose definiens is:

$$(\forall x \exists y \varphi) \to \forall x \varphi[y := f(x)]$$

Our axiom-based solution to the skolemization problem is admittedly not ideal. Other approaches for dealing with skolemization are available. In principle, one could reconstruct all E's skolemization steps in Mizar using Mizar's choice operator.[4] To do this, given a formula $\psi := \forall x \exists y \varphi$, one can proceed as follows:

1. Introduce a new (non-dependent) type $\tau\psi$ inhabited (by definition) by those objects that satisfy the sentence $\forall x \exists y \varphi$.

2. Prove that $\tau_\psi$ is inhabited by exploiting the fact that the domain of interpretation of any first-order structure is non-empty.

3. Define $f$ outright using Mizar's built-in Hilbert choice operator:

   ```
   definition
     let x;
     func f equals the T;
   end;
   ```

   where `T` is the Mizar type corresponding to $\tau_\psi$.

[4] Unlike in Hilbert's $\varepsilon$-calculus, where the choice operator applies to formulas, the choice operator in Mizar applies to types.

Despite the advantage of being explicit, initial experiments with this "explicit skolemization" approach make clear that the precise details of skolemization steps matter: we have found that skolemization steps in which multiple skolem functions are introduced at once complicates the explicit approach; the algorithm we have just sketched does not apply to such cases. Since E's skolemization procedure can in fact produce such steps, explicit skolemization limits the scope of our tool compared to axiom-based skolemization. Of course, we could implement our own clausifier that provides us the required level of granularity of clausification. However, if we wish to account for every step of an arbitrary E derivation, then the axiom-based solution seems preferable.

### 3.3 Resolution

Targeting Mizar is sensible because it has a single rule of inference, `by`, which takes a variable number of premises. The intended meaning of an application

$$\frac{\varphi_1, \ldots, \varphi_n}{\varphi}\ \texttt{by}$$

of `by` is that $\varphi$ is an "obvious" inference from premises $\varphi_1, \ldots, \varphi_n$. See Davis [3] and Rudnicki [18] for more information about the the tradition of "obvious inference" in which Mizar works. The implementation in Mizar diverges from these proposals [25], but roughly speaking a conclusion in Mizar is obtained by an "obvious inference" in from some premises if there is a derivation of the conclusion from a set of assumptions in which at most one substitution instance of at most one universal premise is chosen.

The main difficulty for mapping arbitrary E derivations to Mizar texts is that Mizar's notion of "obvious inference" overlaps with resolution, but is neither weaker nor stronger than it. The consequence of this is that it is generally not the case that an application of resolution can be mapped to a single acceptable application of Mizar's `by` rule. Consider the following example:

$$\frac{\forall x[\neg A(x) \vee B(x)] \qquad \forall x, y[\neg B(x) \vee \neg B(x) \vee \neg B(y)]}{\forall x, y[\neg A(x) \vee \neg B(y)]}\ \text{Resolution}$$

This application of resolution[5] simply eliminates $B(x)$ from the premises. The difficulty here is that we cannot choose a single substitution instance of the premises such that we can find a Herbrand derivation, and hence the inference is non-obvious even though it is essentially (i.e., at the clause level) a single application of propositional resolution.

The reason for the difficulty is that we are working at the level of formulas rather than clauses. A solution is available: map the application of resolution not to a single application of Mizar's `by` rule, but to a proof:

```
premise1:
for X1 holds ((not A X1) or B X1);

premise2:
for X1, X2 holds ((not A X1) or (not B X1) or (not B X2));

theorem
for X1, X2 holds ((not A X1) or B X2)
proof
  let c1, c2;
  (not A c1) or B c1 by premise1;
  hence thesis by premise2;
end;
```

[5] To be precise, an application of factoring is suppressed in this example.

This Mizar proof has three steps and two applications of `by`. In each application of `by`, there is a single instance of a single universal formula (in the first case the universal formula is `premise1`, and in the second application the universal premise is `premise2`). Note that the substitution instances are not built from constants and function symbols, but from (fixed) variables.

### 3.4 Compressing Mizar proofs

The "epicycles" of resolution notwithstanding, Mizar is able to compress many of E's proof steps: many steps can be combined into a single acceptable application of Mizar's `by` rule of inference. For example, if $\varphi$ is inferred from $\varphi'$ from variable renaming, and $\varphi'$ is inferred by an application of conjunction elimination to $\varphi''$, typically in the Mizar setting $\varphi$ can be inferred from $\varphi''$ alone by a single application of `by`. This is typical for most of the fine-grained rules of E's calculus: their applications are acceptable according to Mizar's `by`, and often they can be composed (sometimes multiple times) while still being acceptable to `by`. Other rules in E's proof calculus that can often be eliminated are variable rewritings, putting formulas into negation normal form, reordering of literals in clauses. More interesting compressions exploit the gap between "obvious inference" and E's more articulated calculus.

It seems to be a hard AI problem to transform arbitrary resolution proofs into human-comprehensible natural deductions. Machine-found proofs seem to have an artificial "flavor" that no rewriting spice can overcome. Still, some simple organizational principles can help to make the proof more manageable.

Compressing proofs helps us to get a sense of what the proof is about. The Mizar notion of obvious inference has been tested through daily work with substantial mathematical proofs for decades, and thus enjoys a time-tested robustness (though it is not always uncontroversial). It seems to be an open problem to specify what we mean by the "true" or "best" view of a proof. When Mizar texts come from E proofs, Mizar finds that the steps are usually excessively detailed (i.e., most steps are obvious) and can be compressed. On the other hand, often the whole proof cannot be compressed into a single application of `by`. We employ the algorithm discussed in [19]: a simple fixed-point algorithm is used to maximally compress a Mizar text. Thus, by repeatedly attempting to compress the proof until we reach the limits of `by`. Yet proof compression is not without its pitfalls. If one compresses Mizar proofs too much, the text can become as "inhuman" as the resolution proof from which it comes. This is a well-known phenomenon in the Mizar community [14]. Experience with texts generated by our translation shows that often considerable compression is possible, but at the cost of introducing a new artificial "scent" into the Mizar text.

## 4 Conclusion and future work

One naturally wants to extend the work here to work with output of other theorem provers, such as Vampire. There is no inherent difficulty in that, though it appears that the TSTP derivations output by Vampire contain different information compared to E proofs; the generic transformations described in Section 3.1 would carry over, but the mapping of skolemization and resolution steps of Sections 3.2 and 3.3 will likely need to be customized for Vampire.

The TPTP language recognizes definitions, but whether an automated theorem prover treats them differently from an axiom is unspecified. In Mizar, definitions play a vital role. After all, Mizar is designed to be a language for developing mathematical theories; only secondarily is it a language for representing solutions to arbitrary reasoning problems, as we are using it in this paper. One could try to detect definitions either by scanning the problem looking for formulas

that have the form of definitions, or, if the original TPTP problem is available, one can extract the formulas whose TPTP role is `definition`. Such definition detection and synthesis has no semantic effect, but could make the generated Mizar texts more manageable and perhaps even facilitate new compressions.

At the moment the tool simply translates E derivations to Mizar proofs. A web-based frontend to the translator could help to spur increased usage (and testing) of our system. One can even imagine our tool as part of the SystemOnTPTP suite [21].

An important incompleteness of the current solution is the treatment of equality. Some atomic equational reasoning steps (specifically, inferences involving non-ground equality literals) in E derivations can be non-Mizar-obvious. One possible solution is to use Prover9's Ivy proof objects. Ivy derivations provide some information (namely, which instances of which variables in non-ground literals) that (at present) is missing from E's proof object output.

For the sake of clarity in the mapping of skolemization steps in E derivation to Mizar steps, we restricted attention to those E derivations in which each skolemization step introduces exactly one new skolem function. The restriction does not reflect a weakness of Mizar; it is a merely technical limitation and we intend to remove it.

We have thus completed the cycle started in [19] and returned from ATPs to Mizar. We leave it to the reader to decide whether he wishes to escape again.

# References


[1] P.B. Andrews. More on the problem of finding a mapping between clause representation and natural-deduction representation. *Journal of Automated Reasoning*, 7(2):285–286, 1991.

[2] Czesław Bylinski and Jesse Alama. New developments in parsing Mizar. 2012. To appear in the proceedings of *Intelligent Computer Mathematics*, 2012.

[3] M. Davis. Obvious logical inferences. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 530–531, 1981.

[4] Hans de Nivelle. Extraction of proofs from the clausal normal form transformation. In *Computer Science Logic*, volume 2471 of *Lecture Notes in Computer Science*, pages 921–951, 2002.

[5] Hans de Nivelle. Translation of resolution proofs into short first-order proofs without choice axioms. *Information and Computation*, 199(1-2):24 – 54, 2005.

[6] Kosta Došen. Identity of proofs based on normalization and generality. *Bulletin of Symbolic Logic*, 9:477–503, 2003.

[7] U. Egly and K. Genther. Structuring of computer-generated proofs by cut introduction. *Computational Logic and Proof Theory*, pages 140–152, 1997.

[8] A. Felty and D. Miller. Proof explanation and revision. In *AAAI-86 Proceedings*, 1987.

[9] A. Grabowski, A. Kornilowicz, and A. Naumowicz. Mizar in a nutshell. *Journal of Formalized Reasoning*, 3(2):153–245, 2010.

[10] C. Lingenfelder. Structuring computer generated proofs. In *International Joint Conference on Artificial Intelligence*. Citeseer, 1989.

[11] Andreas Meier. System description: Tramp: Transformation of machine-found proofs into natural deduction proofs at the assertion level. In David McAllester, editor, *Automated Deduction - CADE-17*, volume 1831 of *Lecture Notes in Computer Science*, pages 460–464. Springer Berlin / Heidelberg, 2000.

[12] Dale Miller. Expansion tree proofs and their conversion to natural deduction proofs. In R. Shostak, editor, *7th International Conference on Automated Deduction*, volume 170 of *Lecture Notes in Computer Science*, pages 375–393. Springer Berlin / Heidelberg, 1984.

[13] Dale A. Miller. A compact representation of proofs. *Studia Logica*, 46:347–370, 1987. 10.1007/BF00370646.

[14] Karol Pąk. The methods of improving and reorganizing natural deduction proofs. In *MathUI10*, 2010.

[15] L. Paulson and Kong Woei Susanto. Source-level proof reconstruction for interactive theorem proving. In Klaus Schneider and Jens Bran, editors, *Theorem Proving in Higher-Order Logics*, volume 4732 of *Lecture Notes in Computer Science*, pages 232–245. Springer, 2007.

[16] F.J. Pelletier. Seventy-five Problems for Testing Automatic Theorem Provers. *Journal of Automated Reasoning*, 2(2):191–216, 1986.

[17] Francis Jeffry Pelletier. A brief history of natural deduction. *History and Philosophy of Logic*, 20(1):1–31, 1999.

[18] P. Rudnicki. Obvious inferences. *Journal of Automated reasoning*, 3(4):383–393, 1987.

[19] Piotr Rudnicki and Josef Urban. Escape to ATP for Mizar. In Pascal Fontaine and Aaron Stump, editors, *PxTP 2011: First International Workshop on Proof eXchange for Theorem Proving*, pages 46–59, 2011.

[20] Stephan Schulz. E-a brainiac theorem prover. *AI Communications*, 15(2):111–126, 2002.

[21] G. Sutcliffe. The TPTP Problem Library and associated infrastructure: The FOF and CNF parts, v3.5.0. *Journal of Automated Reasoning*, 43(4):337–362, 2009.

[22] Geoff Sutcliffe, Stephan Schulz, Koen Claessen, and Allen Van Gelder. Using the TPTP language for writing derivations and finite interpretations. In Ulrich Furbach and Natarajan Shankar, editors, *Automated Reasoning*, volume 4130 of *Lecture Notes in Computer Science*, pages 67–81. Springer Berlin / Heidelberg, 2006.

[23] Josef Urban. MPTP 0.2: Design, implementation, and initial experiments. *Journal of Automated Reasoning*, 37(1-2):21–43, 2006.

[24] Josef Urban and Geoff Sutcliffe. ATP-based cross-verification of Mizar proofs: Method, systems, and first experiments. *Mathematics in Computer Science*, 2(2):231–251, 2008.

[25] Freek Wiedijk. Checker. Available online at `http://www.cs.ru.nl/~freek/mizar/by.pdf`.

[26] Larry Wos. The problem of finding a mapping between clause representation and natural-deduction representation. *Journal of Automated Reasoning*, 6(2):211–212, 1990.

# A Pelletier's Dreadbury Mansion Puzzle: From E to Mizar

```
Ax1: ex X1 st (lives X1 & killed X1,agatha) by AXIOMS:1;

Ax2: lives X1 implies (X1 = agatha or X1 = butler or X1 = charles) by AXIOMS:2;

Ax3: killed X1,X2 implies hates X1,X2 by AXIOMS:3;

Ax4: killed X1,X2 implies (not richer X1,X2) by AXIOMS:4;

Ax5: hates agatha,X1 implies (not hates charles,X1) by AXIOMS:5;

Ax6: (not X1 = butler) implies hates agatha,X1 by AXIOMS:6;

Ax7: (not richer X1,agatha) implies hates butler,X1 by AXIOMS:7;

Ax8: hates agatha,X1 implies hates butler,X1 by AXIOMS:8;

Ax9: ex X2 st (not hates X1,X2) by AXIOMS:9;

Ax10: not agatha = butler by AXIOMS:10;

S1: killed skolem1,agatha by Ax1,SKOLEM:def 1;

S2: agatha = skolem1 or butler = skolem1 or charles = skolem1 by Ax2,Ax1,SKOLEM:def 1;

S3: not hates agatha,(skolem2 butler) by Ax9,SKOLEM:def 2,Ax8;

S4: hates charles,agatha or skolem1 = butler or skolem1 = agatha by Ax3,Ax1,SKOLEM:def 1,S2;

S5: butler = (skolem2 butler) by S3,Ax6;

S6: not hates butler,butler by Ax9,SKOLEM:def 2,S5;

S7: hates butler,butler or skolem1 = agatha by Ax4,Ax7,Ax1,SKOLEM:def 1,Ax5,S4,Ax6,Ax10;

S8: skolem1 = agatha by S7,S6;

theorem
killed agatha,agatha
proof
  now
    assume S9: not killed agatha,agatha;
    thus contradiction by S1,S8,S9;
  end;
  hence thesis;
end;
```

Pelletier's Dreadbury Mansion [16] goes as follows:

> Someone who lives in Dreadbury Mansion killed Aunt Agatha. Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein. A killer always hates his victim, and is never richer than his victim. Charles hates no one that Aunt Agatha hates. Agatha hates everyone except the butler. The butler hates everyone not richer than Aunt Agatha. The butler hates everyone Aunt Agatha hates. No one hates everyone. Agatha is not the butler.

The problem is: Who killed Aunt Agatha? (Answer: she killed herself.) The problem belongs to the TPTP Problem Library (it is known there as PUZ001+1) and can easily by solved by many automated theorem provers. Above is the result of mapping E's solution to a standalone Mizar text and then compressing it as described in Section 3.4. Two skolem functions `skolem1` (arity 0) and `skolem2` (arity 2) are introduced. There are 10 axioms and 8 steps that do not depend on the negation of the conjecture (`killed agatha,agatha`) This problem is solved essentially by forward reasoning from the axioms; proof by contradiction is unnecessary, but that is the nature of E's solution.