WinKE: A Proof Assistant for Teaching Logic

Marcello D'Agostino and Ulrich Endriss

Department of Human Sciences, University of Ferrara,
Via Savonarola 38, 44100 Ferrara, Italy
dgm@dns.unife.it
http://dns.unife.it/dgm/WinKE/

Abstract. WinKE is a new interactive theorem proving assistant based on the KE calculus, a refutation system which combines features from Smullyan's analytic tableaux and Gentzen's natural deduction. It has been developed to support teaching logic and reasoning to undergraduate students. The software is supportive of and complementary to an introductory textbook on classical logic [Mondadori and D'Agostino, 1997]. This paper provides a short introduction to the KE calculus and describes the main features of the WinKE system.

1 The KE Calculus

In this paper we present an interactive pedagogical system for teaching classical logic based on the system KE (see [D'Agostino and Mondadori, 1994] and [Mondadori and D'Agostino, 1997]). This is a refutation system which, in our view, supersedes both natural deduction and Smullyan's tableau method, namely the most popular systems currently used in teaching. One drawback of the method of analytic tableaux is that its rules do not capture one of the essential semantic features of classical logic, namely the bivalent character of the underlying notion of truth (see [Broda *et al.*, 1995]). This problem could be solved by augmenting the tableau rules with the rule given below:

$$\frac{}{A \mid \neg A}$$

We call this rule *PB*, from *Principle of Bivalence*. The formula A which appears in the two nodes generated by an application of this rule is called *PB*-formula. It is well-known that in a tableau environment this rule corresponds to the cut rule of the sequent calculus. Therefore, its addition allows tableaux to represent the use of "lemmas" in proofs. However, this would be only an *ad hoc* adjustment. The tableau rules are complete *without* the *PB*-rule, so there is no natural way of incorporating this rule in the tableau method (or, if you wish, there is a natural way which amounts to "simulating" the system that we are going to present below). A more interesting approach consists in *weakening* the operational rules so that the system becomes complete only with the addition of the *PB*-rule. There is (under very plausible assumptions) a unique system of rules of this kind: the system KE. Its rules are given in Table 1. The crucial difference compared with the traditional tableau method is that *PB* is the *only* branching rule. All the rules are tree-expansion rules, like the tableau rules, and the notions of closed tree, refutation and proof are defined in the same way as for tableaux.

Table 1. The KE Rules for Propositional Logic

Alpha Rules	$\dfrac{A \wedge B}{\begin{array}{l}A\\ B\end{array}}$	$\dfrac{\neg(A \vee B)}{\begin{array}{l}\neg A\\ \neg B\end{array}}$	$\dfrac{\neg(A \rightarrow B)}{\begin{array}{l}A\\ \neg B\end{array}}$	$\dfrac{\neg\neg A}{A}$
Beta Rules	$\dfrac{\begin{array}{c}A \vee B\\ \neg A\end{array}}{B}$	$\dfrac{\begin{array}{c}\neg(A \wedge B)\\ A\end{array}}{\neg B}$	$\dfrac{\begin{array}{c}A \rightarrow B\\ A\end{array}}{B}$	$\dfrac{\begin{array}{c}A \rightarrow B\\ \neg B\end{array}}{\neg A}$
PB	$\dfrac{}{A \mid \neg A}$			

Given these operational rules, the *PB*-rule is no longer redundant, and is indeed necessary to obtain classical completeness. So, in the system KE, the inferential power pertaining to the logical operators is separated from the inferential power pertaining to the classical (bivalent) interpretation of "true" and "false". As a result, there is a "natural" strategy governing the applications of *PB* which consists in applying it on a branch only when none of the operational rules is (non-redundantly) applicable. But what formulas should be chosen as *PB*-formulas? Let us call *analytic restriction* of KE the system which results from KE by restricting the applications of the *PB*-rule to subformulas of the formulas occurring above. Such a restriction clearly obeys the subformula principle and can be easily shown to be complete. Therefore, contrary to the Gentzenian tradition, the subformula property for KE is obtained *not* by *eliminating* "cuts" (i.e. the applications of *PB*), but by *restricting* them to subformulas. The analytic restriction of KE is amenable to systematic proof-search procedures of the same kind as the ones used for the tableau method.

We haven't discussed first order versions of KE. It suffices here to say that the quantification theory of KE is identical to that of tableaux, i.e. it results from the propositional fragment by adding the tableau quantifier rules or any of their improvements (see [Mondadori and D'Agostino, 1997]).

2 Overview of WinKE

In this section we present an overview of WinKE's interface and functionality. Its design was strongly inspired by the work described in [Pitt, 1995]. WinKE runs under Windows 95 and has been implemented using LPA WinProlog (by Logic Programming Associates, London, 1995). A detailed account on design and implementation issues (of a preliminary version of the tool) is given in [Endriss, 1996].

Using the software refutations for formulas in either propositional or first order logic can be performed. The WinKE process of constructing a proof tree is as faithful as possible to the pen-and-paper procedure. Using the same method taught during lectures should make it easier to learn how to use the tool. The WinKE system features an interactive interface via menus, dialogues, and graphic tools, a graphic window for constructing and displaying a KE proof tree, on- and off-line proof checking, and is completed by the option to automatically perform proofs (or parts of a proof), and to build up countermodels. User support is provided by bookkeeping facilities, hints, different ways of retracting proof steps, and a detailed on-line help system. The system provides several files with example problems. New ones can be edited directly within WinKE.

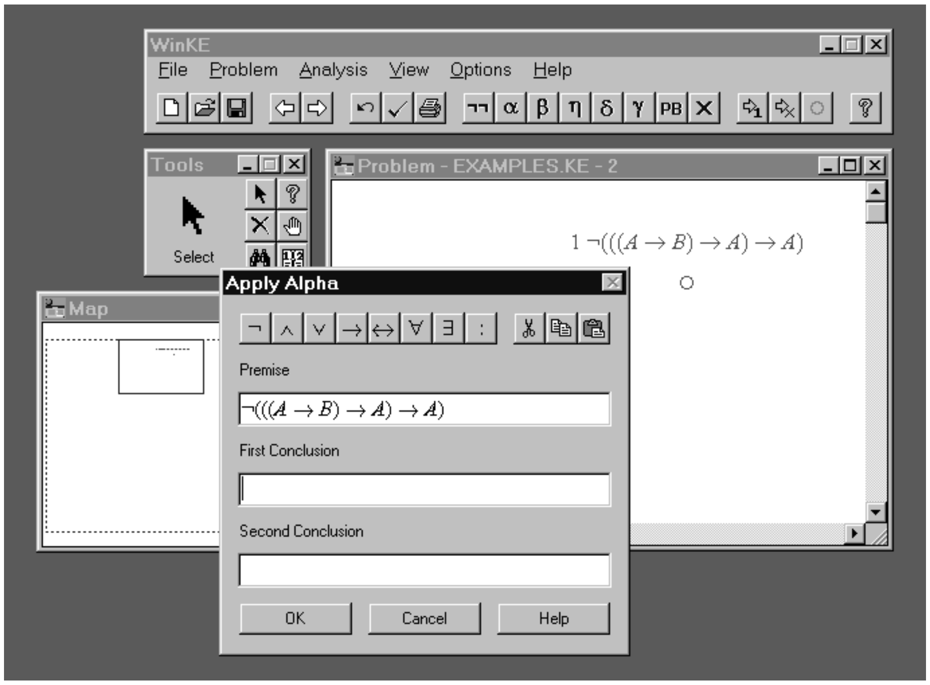
2.1 Interface and Graphic Tools

The interface of WinKE consists of four windows: the main window providing the menus, the graphic window to display and manipulate KE proof trees, a viewer, which displays a scaled-down view of the entire drawing board, and a tool box with several graphic tools (see Figure 1). The design of the interface is close to that of Windows standard software. For example the use of the graphic tools works similarly to standard graphics programs. All user actions are either invoked by that tools or via menus. Note that the buttons in the menu window provide shortcuts to menu options likely to be used frequently.

Proof trees displayed in the graphic window consist of graphical objects, which are either formulas or so-called branch markers. Those markers are used to refer to a certain branch. They are either represented by a circle (in the case of an open branch) or a cross (for closed branches) just below the last formula of that branch. Every formula is associated with a certain number, which can be used to refer to parent formulas. The small rectangle in the window captioned "Map" corresponds to the part of the drawing area visible in the graphic window. Using the mouse it can be moved around when working on large trees.

A particular graphic tool is chosen by clicking on it. It then can be used in the graphic window. The default graphic tool is the select tool, which is used to construct the tree. Clicking on formulas or branch markers with the select tool will highlight them. Then particular actions may be selected from the menus to be applied to the selected objects. This will typically be an application of a KE rule. Depending on the actual menu item chosen a dialogue requesting further input from the user (i.e. the conclusion(s) of the rule) may be invoked, and finally the tree is expanded accordingly. Figure 1 shows an example alpha rule application. After having selected formula number 1 and the branch marker beneath it the user pressed the "α"-button to invoke the dialogue. S/he then has to enter the correct conclusions.

There are also a delete and a retract tool, both of them to take back (wrong) proof steps. The difference between them is, that the delete tool simply prunes the tree at the clicked formula, whereas the retract tool only deletes the formulas



1 $\neg(((A \to B) \to A) \to A)$

Premise

$\neg(((A \to B) \to A) \to A)$

First Conclusion

Second Conclusion

Fig. 1. Applying an Alpha Rule

that logically depend on the clicked one, i.e. that could not have been derived without that formula being on the same branch. This is completed by a standard undo-option available from the menus. The remaining two tools don't manipulate the tree, they are only used for gathering information. The hint tool applied to an open branch marker will highlight all formulas that have not yet been analysed on that (open) branch. Vice versa clicking a formula will highlight all open branch markers denoting a branch which that formula has not yet been analysed on. Finally the bookkeeping tool will display the bookkeeping information available for each node. If that node is a formula, the bookkeeping information consists of the KE rule used to derive it, the parent formula(s), and possibly the sibling formula. In addition formulas that are either analysed or subsumed on all open branches are marked. If the node clicked on is a closed branch marker, the bookkeeping tool reveals which pair of formulas has been used to close that branch.

2.2 Modes, Checking, and Automated Deduction

WinKE provides three different levels of supervision and assistance respectively, the teaching modes. They are called supervisor, pedagogue, and assistant. In supervisor mode within the rule application dialogues any (syntactically correct) input is accepted, whereas in pedagogue mode the correctness of the rule application is checked on-line. The same is true for the assistant, but here the user input is reduced to a minimum. That means, for the simple rules (the propositional ones apart from *PB*), no input of the conclusion(s) is required as their derivation is straight-forward given the premise(s). For the other rules the system gives a list of possible inputs to choose from (alternatively the user can of course also type in a formula). In case the supervisor mode has been used WinKE also provides off-line checking. This will display all errors on a tree in turn and offer the possibility to retract the associated formulas directly.

For the on- as well as for the off-line checking the user may choose the level of error reporting. Only the very basic KE rules are checked in any case, in addition you may or may not add checking for beta simplification (subsumption), analytic application of *PB*, and/or checking of the order of rule applications (like for example: analyse an alpha formula before you split a branch using *PB*, etc.).

In particular to make the system a more convenient assistant, but also to be able to demonstrate proofs to novice users, the option to automatically derive (parts of) proofs has been added. You can either ask WinKE to perform the next proof step on a selected branch, to finish a branch, or to complete an entire proof. For consistent sets of formulas, i.e. if there are open branches that cannot be closed, WinKE can automatically derive the description of a countermodel.

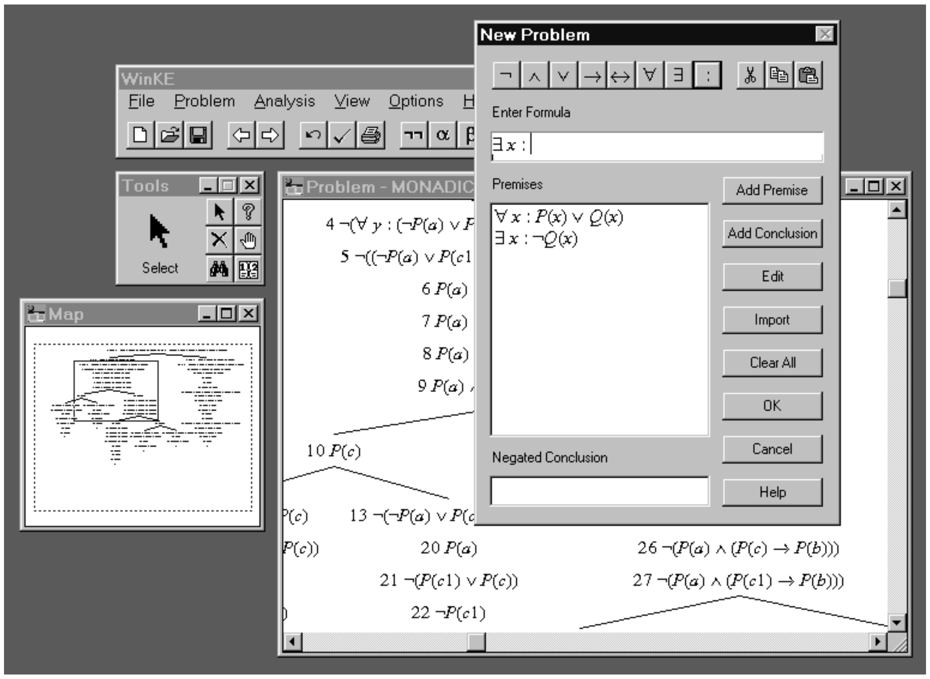
2.3 Additional Features

KE problems are saved in files, either as problems, proofs, or incomplete proofs. Within the program you can jump between different problems of the same file. Problem files are edited in the same environment as they are worked on (see Figure 2 for an example). You have the option to cut and paste from existing problems when defining new ones. This offers a comfortable way for teachers to write up and test new exercises. Students could be encoraged to make their own experiments trying different sets of formulas.

Every problem is associated with a text of arbitrary length. Also that text can be edited and read directly within WinKE. In the context of a student exercise it might contain hints for finding a solution or a reference to a page of a textbook. Other features available include printing and generating LaTeX descriptions of proof trees. Parts of the functionality of WinKE can be made password protected, for example to disable automated proving, the assistant mode, or the proof checker. The tool is completed by a comprehensive on-line help system.

3 Conclusion

We have presented the pedagogic tool WinKE. Its principal task is to support teaching in the context of an introductory course on elementary classical logic. The software is complementary to the logic textbook [Mondadori and D'Agostino, 1997] which is based on KE. Given the automated deduction feature the system can also be used as a proof assistant. It is easy


New Problem
WinKE
File Problem Analysis View Options
Tools
Select
Map
Problem - MONADIC
Enter Formula
Premises
Negated Conclusion
Add Premise
Add Conclusion
Edit
Import
Clear All
OK
Cancel
Help
4 $\neg(\forall y : (\neg P(a) \vee P$
5 $\neg((\neg P(a) \vee P(c1$
6 $P(a)$
7 $P(a)$
8 $P(a)$
9 $P(a)$
10 $P(c)$
13 $\neg(\neg P(a) \vee P(c$
20 $P(a)$
21 $\neg(P(c1) \vee P(c))$
22 $\neg P(c1)$
26 $\neg(P(a) \wedge (P(c) \rightarrow P(b)))$
27 $\neg(P(a) \wedge (P(c1) \rightarrow P(b)))$

$\exists x :$

$\forall x : P(x) \vee Q(x)$
$\exists x : \neg Q(x)$

Fig. 2. Editing a KE Problem

to learn, comfortable to use, and visually satisfying. Teachers will particularly benefit from WinKE when preparing and testing exercises for their students.

Other logic tutors include for example popular programs like Tarski's World [Barwise and Etchemendy, 1991]. Using Tarski's World students are asked to verify first order formulas stating propositons about simple three-dimensional worlds inhabited by geometric objects. Though it might be useful in teaching the very basics (in particular how to translate "real world" situations into logic), it does not deploy a systematic proof procedure and therefore cannot be called a proof assistant. The Hyperproof program [Barwise and Etchemendy, 1994] is a derivative of Tarski's World. It is used to construct proofs of sentences on that same geometric world applying a natural deduction like calculus. As it is restricted to examples of that particular domain it is difficult to be compared with WinKE. The highly sophisticated interface of Hyperproof definitely makes it very attractive to students, but at the same time it also makes it more difficult to learn how to use the tool. WinKE has been designed to simulate an existing proof procedure. In that sense it is supportive of the teaching process. For Hyperproof on the contrary teaching has to be centered around the software. The Tableau II program [Potter and Watt, 1988] is based on semantic tableaux and therefore is much closer to WinKE than the other two systems mentioned. As far as interface and usability are concerned WinKE clearly offers noticeable advantages over Tableau II.

Further information about WinKE as well as the KE calculus can be found at the URL: `http://dns.unife.it/dgm/WinKE/`.

References

[Barwise and Etchemendy, 1991] Jon Barwise and John Etchemendy. *Tarski's World.* CSLI Publications, Stanford, 1991.

[Barwise and Etchemendy, 1994] Jon Barwise and John Etchemendy. *Hyperproof.* CSLI Publications, Stanford, 1994.

[Broda *et al.*, 1995] Krysia Broda, Marcello D'Agostino, and Marco Mondadori. *A Solution to a Problem of Popper.* In *The Epistemology of Karl Popper*, Kluwer, 1995.

[D'Agostino and Mondadori, 1994] Marcello D'Agostino and Marco Mondadori. *The Taming of the Cut. Classical Refutations with Analytic Cut.* Journal of Logic Computation, 4(3):285–319, 1994.

[Endriss, 1996] Ulrich Endriss. *A KE Based Theorem Proving Assistant.* Master's thesis, Imperial College, London, 1996.

[Mondadori and D'Agostino, 1997] Marco Mondadori and Marcello D'Agostino. *Logica.* Edizioni Scolastiche Bruno Mondadori, Milan, 1997.

[Pitt, 1995] Jeremy Pitt. *MacKE: Yet Another Proof Assistant & Automated Pedagogic Tool.* In P. Baumgärtner, R. Hähnle, and J. Possegga, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, Springer Verlag, 1995.

[Potter and Watt, 1988] Michael Potter and Duncan Watt. *Tableau II: A Logic Teaching Program.* Oxford University Computing Services, Learning and Resource Centre, Oxford, 1988.

[Prawitz, 1965] D. Prawitz. *Natural Deduction. A Proof-Theoretical Study.* Almqvist & Wilksell, Uppsala, 1974.

[Smullyan, 1968] Raymond Smullyan. *First-Order Logic.* Springer Verlag, 1968.