Mathematical Logic Tutor - Propositional Calculus

Antonio Moreno, Neus Budesca
amoreno@etse.urv.es, nbh.ei@alumne.etse.urv.es
Departament d'Enginyeria Informàtica i Matemàtiques
Escola Tècnica Superior d'Enginyeria
Universitat Rovira i Virgili, Tarragona

1. Introduction

This document describes a program called *Mathematical Logic Tutor - Propositional Calculus* (*MLT-PC*). This program was developed by Neus Budesca, a URV student, in order to obtain an MSc Degree in the studies of Computer Science. She presented the final version of this system in September 1999. This program is intended to help the students of *Introduction to Logic* (*ILO*), which is an optional course in the Computer Science curriculum. The main aim of *MLT-PC* is to provide the *ILO* students with a software tool that assists them in the study of the first part of the course, devoted to Propositional Calculus.

2. Introduction to Logic

ILO is a course which is given in the Fall term. It is optional in the Computer Science curriculum. It is composed of lectures that take 3 hours each week during a whole term (15 weeks). The course is devoted to the basic proof methods in Propositional and Predicate Calculus, as shown in the following description of the covered items:

- Introduction to Propositional Calculus
  Language of the Propositional Calculus. Formalizing natural language statements. Validating arguments.
- Proof theory in Propositional Calculus
  Natural deduction. Derived rules, theorems, metatheorems, logical equivalences. Boolean algebra, normal forms.
- Automating proofs in Propositional Calculus
  Resolution. Types of resolution.
- Introduction to Predicate Calculus
  Predicates, domains, language of the Predicate Calculus.
- Proof theory in Predicate Calculus
  Natural deduction. Derived rules, theorems, metatheorems, logical equivalences.
- Automating proofs in Predicate Calculus
  Adding functions to the Predicate Calculus. Resolution. Undecidibility of the Predicate Calculus.
- First order theories
  Theories: vocabulary, axioms, theorems. Peano's Arithmetic Theory.

3. MLT-PC

MLT-PC is a program that has been developed to help *ILO* students to study the first part of the course, the one devoted to Propositional Logic. The system has been written in Visual Basic, and it may be executed in any standard PC. It may be freely downloaded from the course web page: http://www.etse.urv.es/recerca/banzai/toni/ILO. In its present version, all the messages are displayed in Catalan, although it would be easy to produce versions in any other language. Another student is currently working in a web-based system that may help the students to study the second part of the course, the one devoted to Predicate Calculus.

MLT-PC has the following capabilities:

- Theoretical content

  The program includes a full-text description of all the theoretical aspects covered in the course. The student may browse through this text either sequencially or selecting an specific topic in an index (see figure 1). The main topics are formalization, natural deduction, Boolean algebra and resolution.

- Exercises

  The system includes exercises related to all the propositional proof methods covered in the course: natural deduction, Boolean algebra and different types of resolution. There are also exercises in two other topics: formalization of expressions in natural language and transformation of formulae into Conjunctive or Disjunctive Normal Form. A more detailed description of each topic is given below. The student may consult the correct answers of the exercises at any point (see figures 2 and 3, with examples of correct solutions of natural deduction and resolution exercises). It may also select an exercise and try to do it. There are two ways of selecting an exercise:

  - The student may browse through a list of suggested exercises, and select any of them (see figure 4).
  - The student may select an arbitrary exercise, by just typing the number that identifies it (e.g. it may just select exercise number 25 of the chapter devoted to natural deduction, without having looked at the premises and the conclusion of this exercise previously).

- Statistics:

  A single copy of the program may be used by any number of students. The user of the system has to identify itself with a code when it logs in. *MLT-PC* keeps a small database in Access with the information about each user. It divides all the available exercises into three categories: those that the student has not yet tried to do, those that it has successfully solved and those that the student has tried to do without success. The student may do an exercise any number of times; however, the program alerts it in case it has previously solved that exercise. It may also try to make again a previously failed exercise. The student may consult statistics about its performance. The program may display the number of exercises solved in each topic. It may also show a list with the status of each tried exercise of each topic (solved/unsolved). These statistics are shown in figures 5 and 6.

- Help:

  The student may ask for help at any point of the program, and the system provides indications about how to use the program in its actual location (e.g. if the student is making a natural deduction proof, the program provides some assistance by indicating which options are available to the student, for instance how it can make a conditional proof or a proof by cases).

The following sections give a more detailed description of each of the areas covered by the system.

4. Formalization

This is the first chapter of the course. The student is given some expressions in natural language (in Catalan) and a list of atoms to be used; it has to provide an accurate formalization in Propositional Calculus. An example is given in figure 7. After writing all the formulae, the system checks whether they are correct or not, giving appropriate error messages to the student (e.g. missing atoms, parenthesis mismatch, wrong operator). In the present version, the answer given by the student must match the solution stored by the system (i.e. the program does not take into account logical equivalences).

5. Natural deduction

The student is given a set of premises and a conclusion, and it has to build a natural deduction proof using the standard rules of introduction and elimination of each of the four main logical connectives (negation, conjunction, disjunction and conditional). The system shows a screen as the one displayed in figure 8. The premises are already written at the beginning of the proof by the system. The student must keep adding formulae to the proof, until it reaches the desired conclusion. In each line it must indicate the next formula, the inference rule that it has applied and the positions in the proof of the formulae to which the inference rule has been applied (e.g. it may add the formula *Q* in line 10 as a result of applying Modus Ponens to the formulae in lines 7 -*If P then Q*- and 4 -*P*-). The system checks that the added formula is indeed a correct application of the rule to the given formulae, giving appropriate error messages if the student has made a mistake.
The systems allows the possibility of opening sub-proofs, so the student may perform conditional proofs (to introduce the conditional operator), *reductio ad absurdum* proofs (to introduce the negation operator) or proof by cases (to eliminate the disjunction operator).

6. Normal forms

There are also exercises in which the student must translate a given formula to Conjunctive Normal Form (CNF) or Disjunctive Normal Form (DNF). The student is given the initial formula, and it is also told which property should be used in each of the steps that must be taken to translate it into CNF or DNF. The program checks the correctness of each step. The system considers properties such as idempotence, commutativity, associativity, distributivity, involution, De Morgan laws, etc. An example is shown in figure 9.

7. Boolean Algebra

The user may also make proofs based on Boolean algebra. In order to proof that *B* follows from *A*, it may use either the *positive* method (showing that the disjunction of *B* and the negation of *A* is a tautology) or the *negative* method (showing that the conjunction of *A* and the negation of *B* is a contradiction). It is similar to the transformation into normal form described in the previous section, because the program helps the student by telling which algebraic property should be applied at each step of the proof, and checking its correct application (see figure 10).

8. Resolution

The student may also make proofs using the resolution method. *MLT-PC* asks the student to perform specific types of resolution in each exercise: non-linear resolution (saturation of levels, systematic resolution) or linear resolution (standard linear resolution, linear resolution restricted to the rightmost literal, primary resolution, unitary resolution, resolution with set of support). For instance, an exercise may ask for a linear, primary and non-unitary resolution. The student must initially provide the clauses resulting from the transformation into CNF of the premises and the negation of the conclusion. When the

system checks that the clauses are correct, it allows the student to proceed with the resolution proof. In each step, *MLT-PC* checks that the application of the resolution rule is correct and that the required properties (e.g. using in each step a unitary clause in a unitary resolution) hold. An example of a proposed resolution exercise is given in figure 11.

9. Summary

TLM-PC is a useful tool for teaching the basic proof methods of standard Propositional Logic. It allows the user to make exercises of natural deduction, Boolean algebra and resolution. The system is constantly checking the user's input, and providing meaningful error messages when the student makes a mistake. Other aspects of the course *Introduction to Logic*, such as formalization of statements in natural language and transformation of formulae into normal form, are also covered by the system. The student has also access to all the theoretical content of the course. The system keeps information about all the previous activity of all its users, and thus it can provide full statistics about each student's performance in all its working sessions. The system runs in any standard PC and may be easily installed by the students, who can download it from the course web page when they wish. It has already been used in the Fall term of the 99-00 course at URV, and it has been a very good help for the *ILO* students for their study of Propositional Calculus.

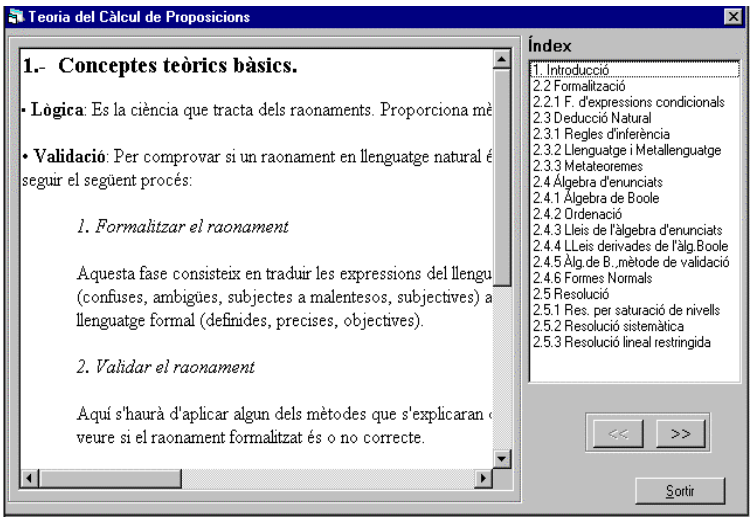

Fig. 1: Browsing through the theoretical concepts..

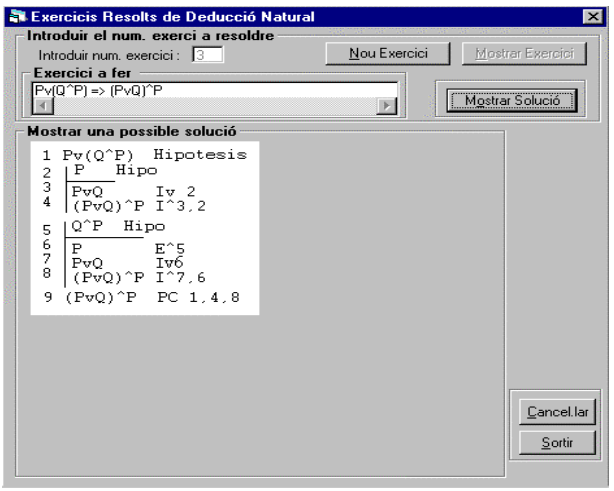

Fig. 2: Example of a natural deduction proof

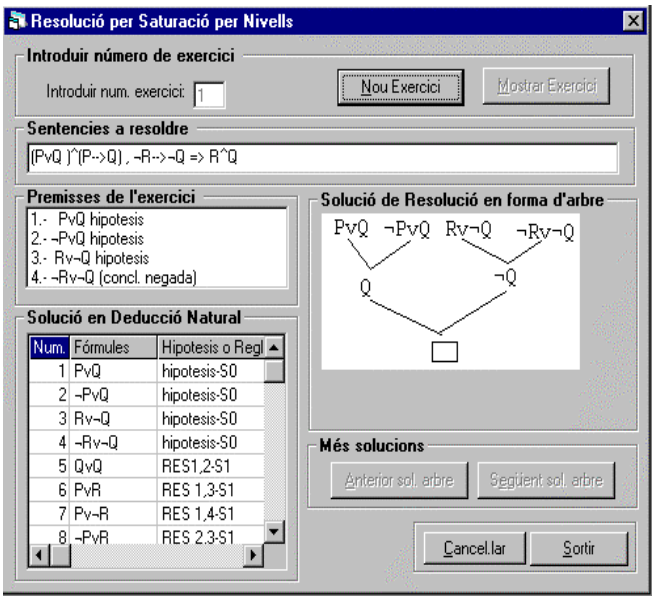

Fig. 3: Example of a resolution proof.

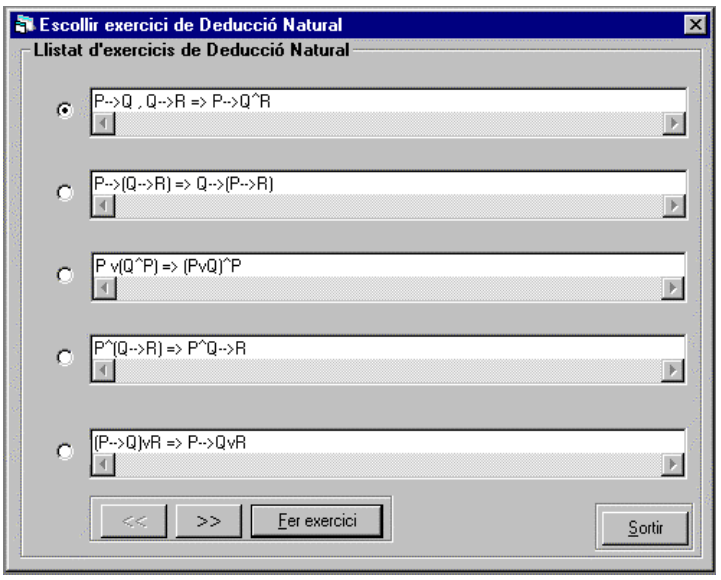

Fig. 4: Selecting a natural deduction exercise

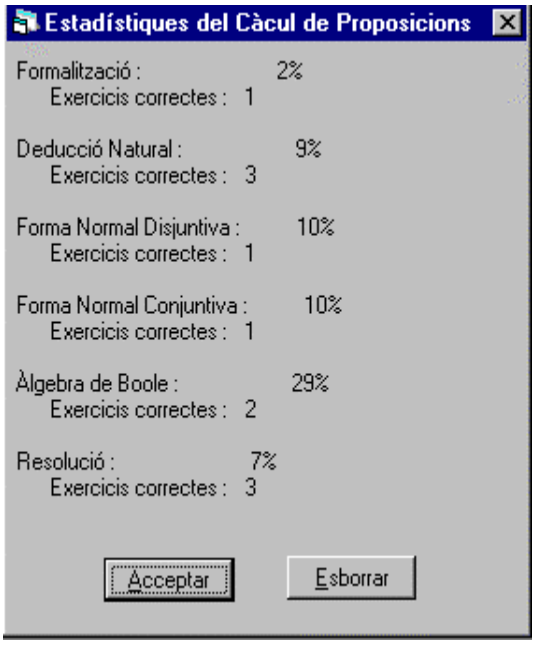

Fig. 5: Statistics: percentage of succesfully solved exercises in each topic

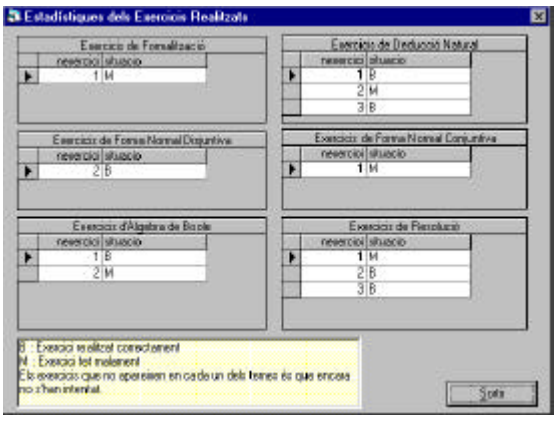

Fig. 6: Statistics: list of exercises tried in each topic (indicating success/failure)

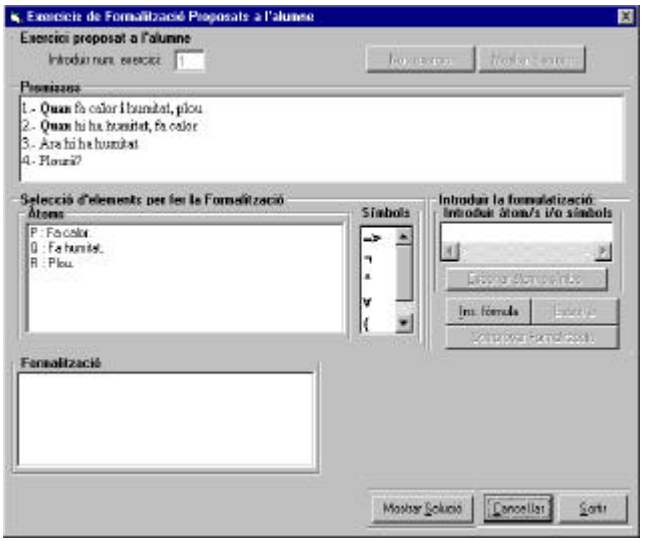

Fig. 7: Formalization exercise

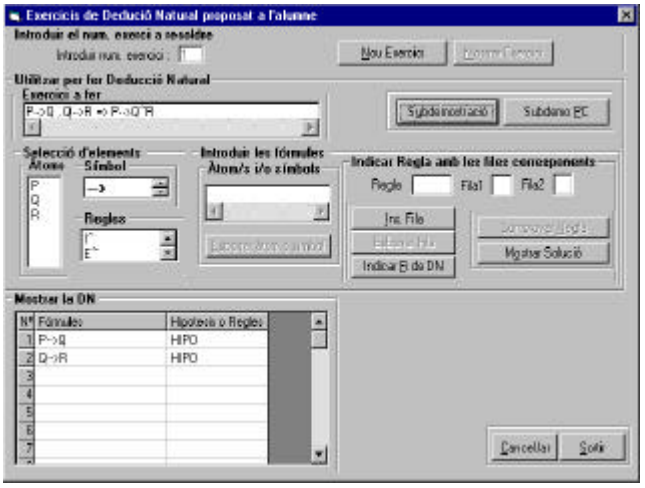

Fig. 8 Beginning of a natural deduction proof

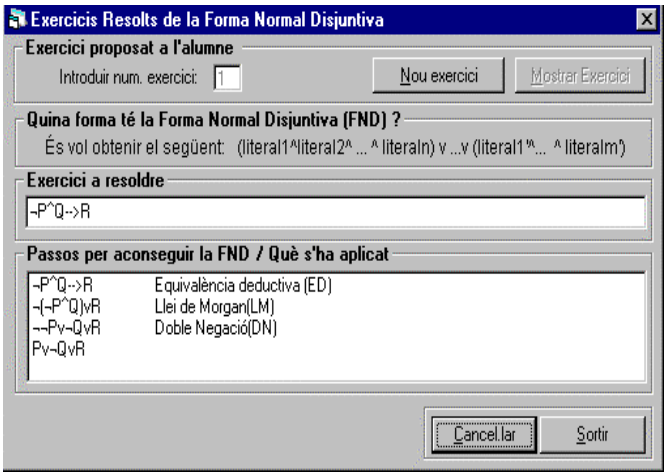

Fig. 9 Transforming a formula into DNF

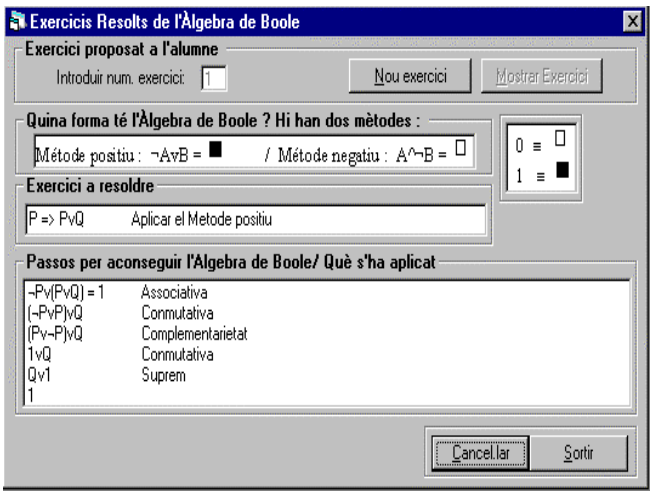

Fig. 10: Example of a proof based on Boole's algebra

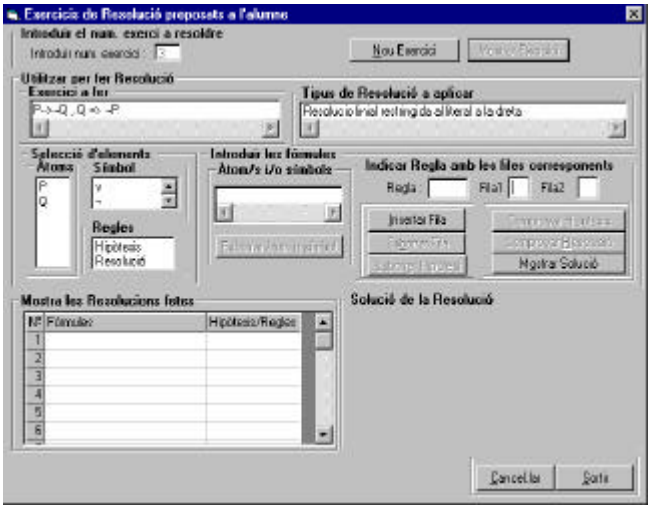

Fig. 11 Beginning of a resolution exercise.