KRRT: Knowledge Representation and Reasoning Tutor System*

Jos A. Alonso[1], Gonzalo A. Aranda[2], and Francisco J. Martn–Mateos[1]

[1] Departamento de Ciencias de la Computacin e Inteligencia Artificial.
Escuela Tcnica Superior de Ingeniera Informtica, Universidad de Sevilla
Avda. Reina Mercedes, s/n. 41012 Sevilla, Spain
{jalonso,fmartin}@us.es,
[2] Dpto Ingeniera Electrnica, Sistemas Informticos y Automtica.
Escuela Politcnica Superior de La Rbida
Universidad de Huelva
Crta. Palos de la Frontera, Palos de la Frontera (Huelva), Spain
gonzalo.aranda@diesia.uhu.es

Abstract. Knowledge Representation & Reasoning (KR&R) is a fundamental topic in Artificial Intelligence. A basic KR language is First–Order Logic (FOL), the most representative logic–based representation language, which is part of almost any introductory AI course. In this work we present KRRT (Knowledge Representation & Reasoning Tutor). KRRT is a Web–based system which main goal is to help the student to learn FOL as a KR&R language.

1 Introduction

Knowledge Representation & Reasoning (KR&R) is a fundamental topic in Artificial Intelligence. A basic KR language is First–Order Logic (FOL), the most representative logic–based representation language, which is part of almost any introductory AI course. Usually students find difficulties in different aspects of using FOL as a knowledge representation and reasoning language. Two of them are about how to translate from natural language sentences into FOL formulae and how to construct formal proofs.

On the web "Logic software and logic education" [1], we can found several systems classified as logic educational software: Jape [4], "Logic Tutor" [2], Logic–ITA [6], "P–Logic Tutor" [7], OLIVER [10] among others. However, most of them are focused on how to construct formal proofs using natural deduction rules [4] or restricting themselves to propositional logic [2,6,7,10]. So, these systems are not concerned with how to use FOL as a KR&R language.

In spite of the amount of existing tools, there is an important subject on teaching logic for which these tools couldn't be used: formalization. Formalization is the process to translate arguments from natural language to logic. This is

* This research was funded in part by the Spanish Ministry of Education and Science under grant TIN2004–03884 and Feder funds.

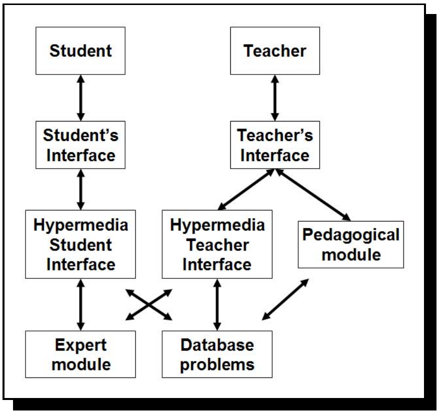
the initial step on “Computational Logic” subject and the main one when logic is used to represent knowledge. Our experience shows us that this is the most difficult process for students, and the way to learn it is solving exercises.

Until now, the only tools which help on teaching the formalization process are interactive forms as “Logic Tutor” by Huth and Ryan [5]. In these forms, the argumentations are presented and the student has to choose among the different proposals. As the forms have fixed answers, the possibilities of alternative and equivalent formalizations are excluded.

In this work we present KRRT (Knowledge Representation & Reasoning Tutor). KRRT is a Web–based system which main goal is help the student to learn FOL as a KR&R language. At knowledge representation stage, KRRT uses the FITS architecture's [3], based on the use of automated reasoning systems (OTTER [8], Vampire [9]) to prove the correctness of the formalization written by users. At reasoning stage, our tool makes a generalization of OLIVER [10] procedure, extending its reasoning power, from propositional logic to first order logic with equality.

2 Architecture and Implementation

KRRT uses the FITS architecture [3] showed in the following figure



The problem database is stored on MySQL. The teacher introduce new problems by an hypermedia application that generates the HTML forms displayed on the teacher interface.

The student interface is a set of HTML forms dynamically generated from the problem database by Apache web server.

The expert module is the section where the system intelligence is located. This module is the responsible to check all the teacher argumentation and the student answers.

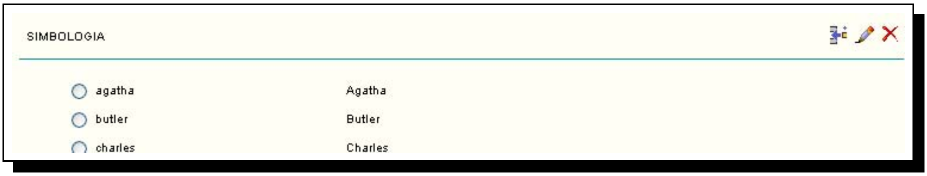
2.1 Login into the System

KRRT is developed for being used on Internet. To enter it, the students must connect to the web page located at `http://www.cs.us.es/clg/krrt`. In the registration form the student is asked about her start level. Then, to analyze the student progress, information about the errors frequently repeated and the learning evolution is stored in the system. This is obtained maintaining the connection user and registering all the activities in the application on the time. This information could be used to estimate the hardness of the problem.

3 Problems Edition

The first KRRT service for teachers is to help on problem edition. To create a new problem, KRRT presents a page with 3 sections. The symbol section, the premise section and conclusion section; all of them with new, edit and delete button.

The teacher can create arguments and formalize them. The main window shows a list of all the arguments in natural language, together with the right formal equations written in first order logic language.

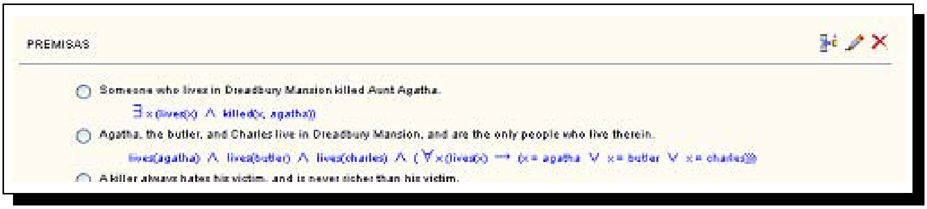


SIMBOLOGIA

○ agatha Agatha

○ butler Butler

○ charles Charles



PREMISAS

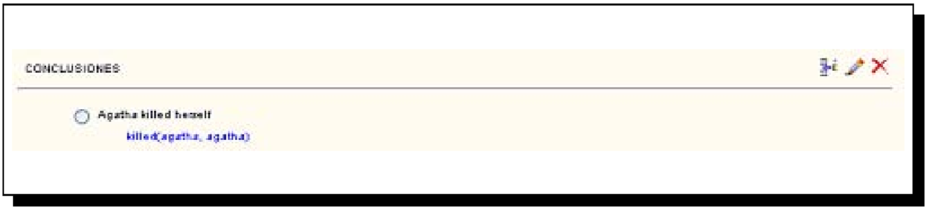
○ Someone who lives in Dreadbury Mansion killed Aunt Agatha.

$\exists x\,(\mathrm{lives}(x) \wedge \mathrm{killed}(x, agatha))$

○ Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein.

$\mathrm{lives}(agatha) \wedge \mathrm{lives}(butler) \wedge \mathrm{lives}(charles) \wedge (\forall x\,(\mathrm{lives}(x) \rightarrow (x = agatha \vee x = butler \vee x = charles)))$

○ A killer always hates his victim, and is never richer than his victim.



CONCLUSIONES

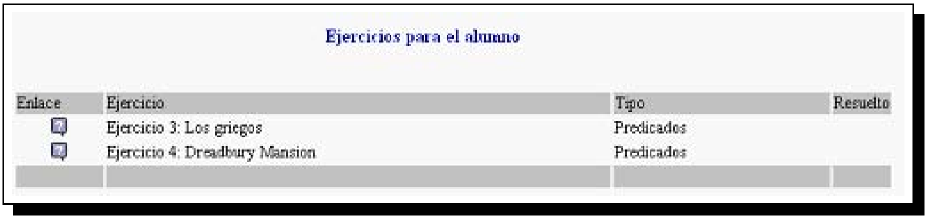
○ Agatha killed herself

$\mathrm{killed}(agatha, agatha)$

In the edition process, the expert module validates syntactically the formalization of each one of the sentences, and the correction of the whole argumentation. At the end, KRRT adds the new problem to its database to make it available for students.

4 Knowledge Representation Task

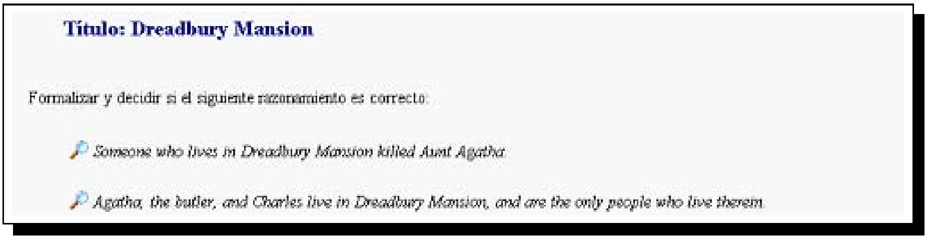
The first kind of task supported by KRRT is Representation. The system offers a collection of argumentation exercises (as it is shown in the following figure), and the student selects the one she wants to solve.



Ejercicios para el alumno

Enlace	Ejercicio	Tipo	Resuelto
	Ejercicio 3: Los griegos	Predicados	
	Ejercicio 4: Dreadbury Mansion	Predicados	

Once a problem is selected, a list of its hypothesis and conclusion is presented. If the student wants to solve the exercise, it must select, one by one, all of them and write the answers.

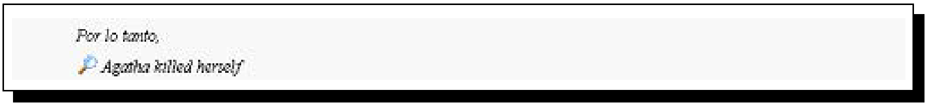


Título: Dreadbury Mansion

Formalizar y decidir si el siguiente razonamiento es correcto:

Someone who lives in Dreadbury Mansion killed Aunt Agatha

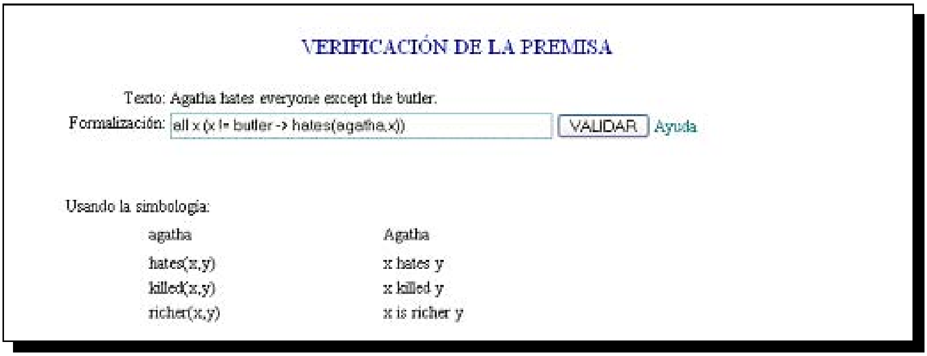
Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein



Por lo tanto,

Agatha killed herself

When an hypothesis is selected, the system shows the verification form. This form contains the hypothesis and a blank box where the student should write the formalization using OTTER syntax.



VERIFICACIÓN DE LA PREMISA

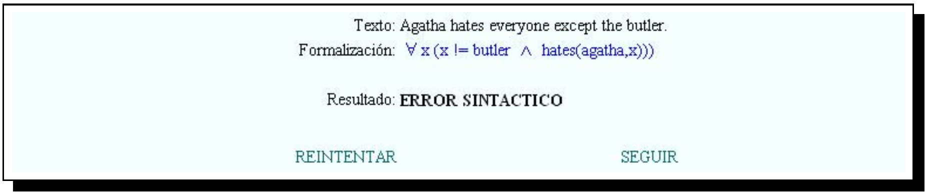
Texto: Agatha hates everyone except the butler.

Formalización: all x (x != butler -> hates(agatha,x))

Usando la simbologia:

agatha	Agatha
hates(x,y)	x hates y
killed(x,y)	x killed y
richer(x,y)	x is richer y

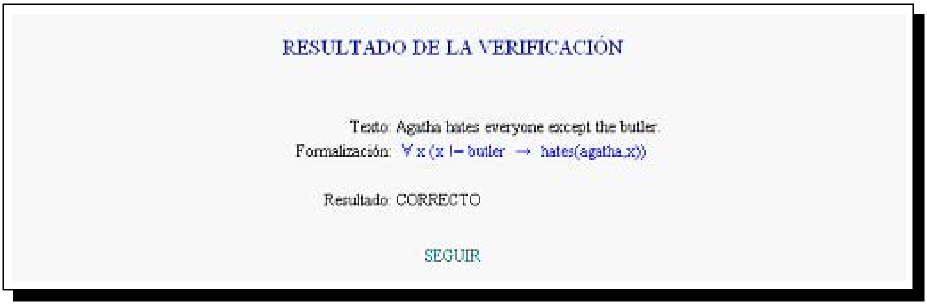
To check the formalization, the Validar button should be pressed and a new window appears with the result. The system reports syntax errors in the proposed formalization:



Texto: Agatha hates everyone except the butler.
Formalización: ∀ x (x != butler ∧ hates(agatha,x)))

Resultado: **ERROR SINTACTICO**

as well as successful formalization modulo semantic equivalence:

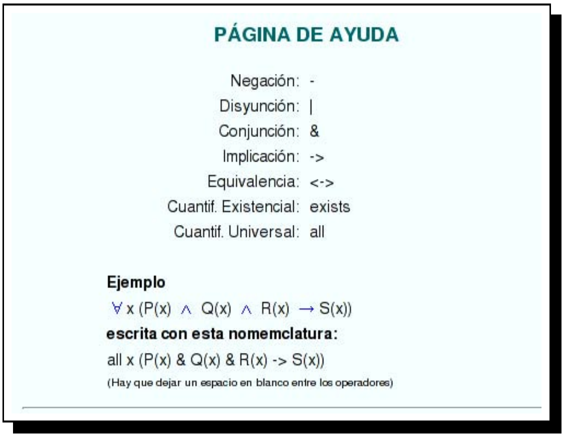


RESULTADO DE LA VERIFICACIÓN

Texto: Agatha hates everyone except the butler.
Formalización: ∀ x (x != butler → hates(agatha,x))

Resultado: CORRECTO

We note that the student formalization is showed in mathematical notation translating OTTER syntax. This happens whenever the system shows the equations.



PÁGINA DE AYUDA

Negación:	-
Disyunción:	|
Conjunción:	&
Implicación:	->
Equivalencia:	<->
Cuantif. Existencial:	exists
Cuantif. Universal:	all

Ejemplo

∀ x (P(x) ∧ Q(x) ∧ R(x) → S(x))

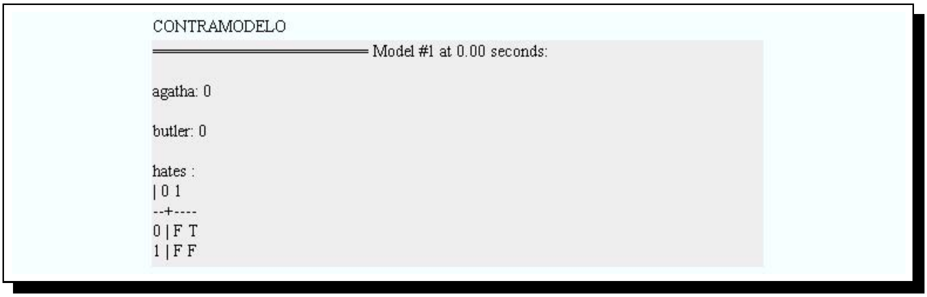
escrita con esta nomemclatura:

all x (P(x) & Q(x) & R(x) -> S(x))

(Hay que dejar un espacio en blanco entre los operadores)

The newest aspect of KRRT is its semantic nature; that is, the system accepts as right answer any formal equation logically equivalent to the solution, using the problem symbolization. In this example, the student could formalize the conclusion using the formula $\forall x\ [x = butler \vee hates(agatha, x)]$ or any other equivalent.

But if the conclusion is formalized as $\forall x\ [x \neq \mathit{butler} \rightarrow \mathit{hates}(\mathit{agatha}, x)]$ then the system indicates that it is an incorrect formalization and shows a counter-model.

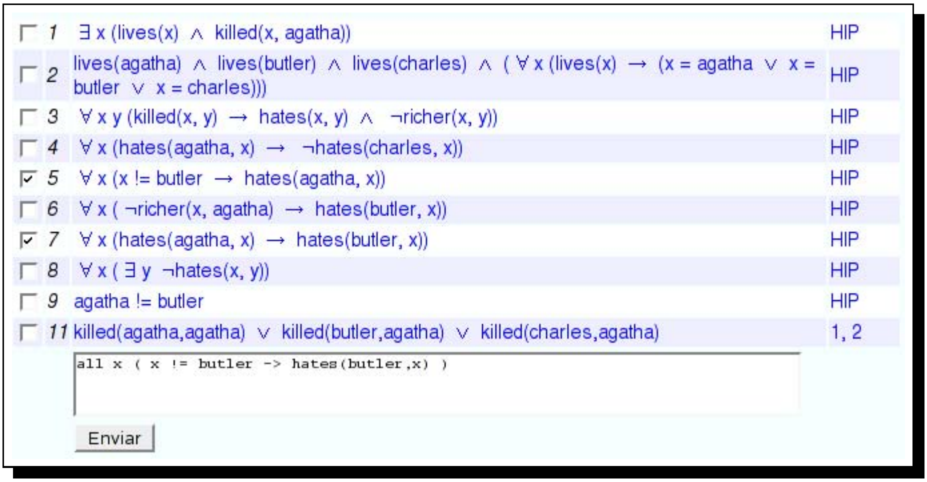


```
CONTRAMODELO
============================ Model #1 at 0.00 seconds:

agatha: 0

butler: 0

hates :
| 0 1
--+----
0 | F T
1 | F F
```

5 Reasoning Tasks

The second kind of task supported by KRRT is Reasoning. When the student finishes to correctly formalize an exercise, then it is added to the reasoning exercises list available for her.

For example, suppose that the student has correctly formalized the Agatha exercise and she wants to start to reasoning on it. In this way, she chooses it from the exercises list and the system shows a form page with the whole formalization and a text box. Here, the student have to click a maximum of two choices and write the formula which she thinks is a logic consequence from the clicked ones.



□	1	∃ x (lives(x) ∧ killed(x, agatha))	HIP
□	2	lives(agatha) ∧ lives(butler) ∧ lives(charles) ∧ (∀ x (lives(x) → (x = agatha ∨ x = butler ∨ x = charles)))	HIP
□	3	∀ x y (killed(x, y) → hates(x, y) ∧ ¬richer(x, y))	HIP
□	4	∀ x (hates(agatha, x) → ¬hates(charles, x))	HIP
☑	5	∀ x (x != butler → hates(agatha, x))	HIP
□	6	∀ x (¬richer(x, agatha) → hates(butler, x))	HIP
☑	7	∀ x (hates(agatha, x) → hates(butler, x))	HIP
□	8	∀ x (∃ y ¬hates(x, y))	HIP
□	9	agatha != butler	HIP
□	11	killed(agatha,agatha) ∨ killed(butler,agatha) ∨ killed(charles,agatha)	1, 2

```
all x ( x != butler -> hates(butler,x) )
```

When the formula is proved, KRRT add it to the premise list, indicating what formulae had been used to deduce it. Also this new formula can be used to deduce new formulae; in the other hand, if the formula cannot be proved, the system shows an counter-example.

To finish the exercise, the student must press the Finalizar button, then KRRT checks semantically if the last formula written is equivalent to the exercise conclusion. If it does, a message is showed notifying that the answer is right. Otherwise, a counter-example is presented.

6 Other Services of KRRT

KRRT offers other services for teachers, as the student history and its evaluation. Also it can do an evaluation about the hardness of the problems. These services are being incorporated currently.

To students, the system offers services like exercise status, which it can be one of these: Not touched, started with errors (syntax or semantics ones) and corrects (finished or not)

Also a pedagogical module is being connected, to extract conclusions that help us to identify the student difficulties in a more detailed way.

7 Conclusions and Future Works

In this work, we present KRRT, the first intelligent tutor for teaching formalization with semantics correction.

Our future works are leaded in two lines: to incorporate KRRT during the present course 2006–07 on teaching the subject of "Computational logic" and to enlarge their services, from the point of view of their automatization, adaptation and evaluation.

References

1. Logic software and logic education, `http://www.cs.otago.ac.nz/staffpriv/hans/logiccourseware.html`
2. Abraham, D., Crawford, L., Lesta, L., Merceron, A., Yacef, K.: The Logic Tutor: A multimedia presentation. Electronic Journal of Computer–Enhanced Learning (October 2001)
3. Alonso, J.A., Aranda, G.A., Martín.-Mateos, F.J.: FITS: Formalization with an Intelligent Tutor System. In: IV International Conference On Multimedia And Information And Communication Technologies In Education (2006)
4. Bornat, R., Sufrin, B.: Jape: A calculator for animating proof-on-paper. In: McCune, W. (ed.) Automated Deduction - CADE-14. LNCS, vol. 1249, pp. 412–415. Springer, Heidelberg (1997)
5. Huth, M., Ryan, M.: LICS web tutor, `http://www.cs.bham.ac.uk/research/projects/lics/tutor`
6. Lesta, L., Yacef, K.: An intelligent teaching assistant system for Logic. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 421–431. Springer, Heidelberg (2002)
7. Lukins, S., Levicki, A., Burg, J.: A tutorial program for propositional logic with human/computer interactive learning. In: SIGCSE 2002, pp. 381–385. ACM, New York (2002)

8. McCune, W.: OTTER 3.3 reference manual. Argonne National Laboratory (2003)
9. Riazanov, A., Voronkov, A.: Vampire 1.1 (system description). In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 376–380. Springer, Heidelberg (2001)
10. Wildenberg, A., Scharff, C.: OLIVER: an OnLine Inference and VERification system. In: 32 ASEE/IEEE Frontiers in Education Conference, IEEE (2002)