KNOWLEDGE ACQUISITION MODELING THROUGH DIALOGUE BETWEEN COGNITIVE AGENTS

Mehdi Yousfi-Monod, Violaine Prince

LIRMM, UMR 5506
161 rue Ada
34392 Montpellier Cedex 5 - France
{yousfi,prince}@lirmm.fr

May 26th, 2005
Synopsis

1. Purpose
2. Knowledge acquisition between agents
3. Our dialogical agents modellisation
4. Architecture and implementation
5. Conclusion, perspectives
Purpose

• defining a set of knowledge acquisition algorithms
Purpose

- defining a set of knowledge acquisition algorithms
- using only dialogue to acquire knowledge
Purpose

- defining a set of knowledge acquisition algorithms
- using only dialogue to acquire knowledge
- being in a teacher-student situation
Purpose

- defining a set of knowledge acquisition algorithms
- using only dialogue to acquire knowledge
- being in a teacher-student situation
- using of socratic dialogues
Purpose

- defining a set of knowledge acquisition algorithms
- using only dialogue to acquire knowledge
- being in a teacher-student situation
- using of socratic dialogues
- belief revision and reasoning
Purpose

- defining a set of knowledge acquisition algorithms
- using only dialogue to acquire knowledge
- being in a teacher-student situation
- using of socratic dialogues
- belief revision and reasoning
- using of the first-order predicate logic (with functions) for their knowledge bases (KB)
Purpose

- defining a set of knowledge acquisition algorithms
- using only dialogue to acquire knowledge
- being in a teacher-student situation
- using of socratic dialogues
- belief revision and reasoning
- using of the first-order predicate logic (with functions) for their knowledge bases (KB)
- allowing the student to question the teacher about unknown predicates or functions before revising its KB
Points 1/2

Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
Cognitive agent

Knowledge base

\[ \text{human}(x) \rightarrow \text{smart}(x) \]
\[ \text{mortal}(x) \rightarrow \text{live}(x) \]

\[ \text{lesson to teach} \]

\[ \text{human}(x) \rightarrow \text{mortal}(x) \]
\[ \text{human}(\text{John}) \]
\[ \text{human}(\text{Mary}) \]

\[ \text{human}(x) \rightarrow \text{not(bird}(x)) \]
\[ \text{animal}(x) \rightarrow \text{living}(x) \]
\[ \text{animal}(\text{titi}) \]
\[ \ldots \]
Modelling of our communicating agents

Cognitive agent

Knowledge base

- $human(x) \rightarrow smart(x)$
- $mortal(x) \rightarrow live(x)$

Lesson to teach

- $human(x) \rightarrow mortal(x)$
- $human(John)$
- $human(Mary)$

- $human(x) \rightarrow not(bird(x))$
- $animal(x) \rightarrow living(x)$
- $animal(titi)$
- ...

Model of itself

- depends on the task and the competences
- Role: Teacher or Student
- Goal: Teach a lesson or Learn a lesson
Modelling of our communicating agents

Cognitive agent

Knowledge base

\[ \text{human}(x) \rightarrow \text{smart}(x) \]
\[ \text{mortal}(x) \rightarrow \text{live}(x) \]

Model of itself

- depends on the task and the competences
- Role: Teacher or Student
- Goal: Teach a lesson or Learn a lesson

Model of the interlocutor

- Identity
- Characteristics
- Model of its KB

\[ \text{mortal}(x) \rightarrow \text{live}(x) \]
\[ \text{human}(Peter) \]
...
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  → A. Ravenscroft, R.M. Pilkington, 2000
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  → A. Ravenscroft, R.M. Pilkington, 2000
- Dialogue models
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  → A. Ravenscroft, R.M. Pilkington, 2000
- Dialogue models
  → P. Cohen, H. Levesque, 1992
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  → A. Ravenscroft, R.M. Pilkington, 2000
- Dialogue models
  → P. Cohen, H. Levesque, 1992
  → J.L. Austin, 1975
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  - → A. Ravenscroft, R.M. Pilkington, 2000
- Dialogue models
  - → P. Cohen, H. Levesque, 1992
  - → J.L. Austin, 1975
  - → M. E. Pollack, 1998
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  - A. Ravenscroft, R.M. Pilkington, 2000
- Dialogue models
  - P. Cohen, H. Levesque, 1992
  - J.L. Austin, 1975
  - M. E. Pollack, 1998
  - G. Sabah et al, 1998
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  - A. Ravenscroft, R.M. Pilkington, 2000
- Dialogue models
  - P. Cohen, H. Levesque, 1992
  - J.L. Austin, 1975
  - M. E. Pollack, 1998
  - G. Sabah et al, 1998
- Reasoning
Knowledge acquisition between agents

- Modelling of our communicating cognitive agents
- Learning methods
  - A. Ravenscroft, R.M. Pilkington, 2000
- Dialogue models
  - P. Cohen, H. Levesque, 1992
  - J.L. Austin, 1975
  - M. E. Pollack, 1998
  - G. Sabah et al, 1998
- Reasoning
  - J.R. Josephson, S.G. Josephson, 1994
Our approach

- KB specifications
Our approach

- KB specifications
  - bootstrap
Our approach

- KB specifications
  - bootstrap
  - Connextity
The connexity notion

A connex KB

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow r(y) \]
\[ q(a) \]
\[ r(b) \]

The corresponding graph

\[ t(z) \land p(x) \rightarrow q(y) \]
The connexity notion

A connex KB

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow r(y) \]
\[ q(a) \]
\[ r(b) \]

The corresponding graph
**The connexity notion**

A connex KB

\[
\begin{align*}
t(z) \land p(x) & \rightarrow q(y) \\
r(x) & \rightarrow q(y) \\
s(x) & \rightarrow r(y) \\
q(a) & \\
r(b) &
\end{align*}
\]

The corresponding graph
The connexity notion

A connex KB

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow r(y) \]
\[ q(a) \]
\[ r(b) \]

The corresponding graph
The connexity notion

A connex KB

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow r(y) \]
\[ q(a) \]
\[ r(b) \]

The corresponding graph
The connexity notion

A connex KB
\[ t(z) \land p(x) \to q(y) \]
\[ r(x) \to q(y) \]
\[ s(x) \to r(y) \]
\[ q(a) \]
\[ r(b) \]

The corresponding graph

A non connex KB
\[ t(z) \land p(x) \to q(y) \]
\[ r(x) \to q(y) \]
\[ s(x) \to u(y) \]
\[ q(a) \]
\[ u(b) \]

The corresponding graph
The connexity notion

A connex KB

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow r(y) \]
\[ q(a) \]
\[ r(b) \]

The corresponding graph

A non connex KB

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow u(y) \]
\[ q(a) \]
\[ u(b) \]

The corresponding graph
**The connexity notion**

**A connex KB**

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow r(y) \]
\[ q(a) \]
\[ r(b) \]

**The corresponding graph**

**A non connex KB**

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow u(y) \]
\[ q(a) \]
\[ u(b) \]

**The corresponding graph**
Connexity

**The connexity notion**

**A connex KB**

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow r(y) \]
\[ q(a) \]
\[ r(b) \]

**The corresponding graph**

**A non connex KB**

\[ t(z) \land p(x) \rightarrow q(y) \]
\[ r(x) \rightarrow q(y) \]
\[ s(x) \rightarrow q(y) \]
\[ q(a) \]
\[ u(b) \]

**The corresponding graph**
### The connexity notion

#### A connex KB

\[
\begin{align*}
t(z) & \land p(x) \rightarrow q(y) \\
r(x) & \rightarrow q(y) \\
s(x) & \rightarrow r(y) \\
q(a) & \\
r(b) &
\end{align*}
\]

#### A non connex KB

\[
\begin{align*}
t(z) & \land p(x) \rightarrow q(y) \\
r(x) & \rightarrow q(y) \\
s(x) & \rightarrow u(y) \\
q(a) & \\
u(b) &
\end{align*}
\]

#### The corresponding graph

![Diagram of connex KB]

![Diagram of non connex KB]
Our approach

- KB specifications
- the functional roles as modeller
Our approach

- KB specifications
- the functional roles as modeller
  - give-knowledge
Our approach

- KB specifications
- the functional roles as modeller
  1. give-knowledge
     → give-knowledge(cat(x) → mortal(x)): ”Cats are mortal.”
Our approach

- KB specifications
- the functional roles as modeller
  1. give-knowledge
  2. askfor/give-information
Our approach

- KB specifications
- the functional roles as modeller
  1. give-knowledge
  2. askfor/give-information
     → askfor-information(cat(Folley)): "Is Folley a cat?"
Our approach

- KB specifications
- the functional roles as modeller
  1. give-knowledge
  2. askfor/give-information
     → askfor-information(cat(Folley)): ”Is Folley a cat?”
     → give-information(true): ”Yes.”
Our approach

- KB specifications
- the functional roles as modeller
  1. give-knowledge
  2. askfor/give-information
  3. give-explanation (predicate case)
Our approach

- KB specifications
- the functional roles as modeller
  1. give-knowledge
  2. askfor/give-information
  3. give-explanation (predicate case)
     → give-explanation\(\text{cat}(x) \leftrightarrow \text{animal}(x) \land \text{pet}(x))\):
     ”A cat is a pet animal.”
### Our approach

- KB specifications
- the functional roles as modeller
  1. give-knowledge
  2. askfor/give-information
  3. give-explanation (predicate case)
  4. say-(dis)satisfaction
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
    → Z. Manna, 1974
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
  - the curious students
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
  - the curious students
    - enlarge the base of one of their predicate
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
  - the curious students

![Increase KB connexity](images/increase_kb_connexity.png)

- enlarge the base of one of their predicate
  - learns $p(a)$, knows $q(a)$, then ask if $q(x) \rightarrow p(x)$
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
  - the curious students
    1. enlarge the base of one of their predicate
    2. increase its KB connexity
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
  - the curious students
    1. enlarge the base of one of their predicate
    2. increase its KB connexity
    3. learn new terms
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
  - the curious students
    1. enlarge the base of one of their predicate
    2. increase its KB connexity
    3. learn new terms
    4. understand implications
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
  - the fundamental axioms
  - FR interpretation axioms
  - the curious students
  1. enlarge the base of one of their predicate
  2. increase its KB connexity
  3. learn new terms
  4. understand implications
     \[ \text{knows } q(x) \rightarrow p(x), \text{ asks if } q(x) \rightarrow ? \rightarrow p(x) \]
### Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
- the knowledge derivation strategies
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
- the knowledge derivation strategies
  - lesson strategies
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
- the knowledge derivation strategies
  1. lesson strategies
  2. dialogue strategies
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
- the knowledge derivation strategies
  1. lesson strategies
  2. dialogue strategies
     - the socratic method for the teacher
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
- the knowledge derivation strategies
  1. lesson strategies
  2. dialogue strategies
     - the socratic method for the teacher
  3. local strategies
Our approach

- KB specifications
- the functional roles as modeller
- the tutored learning
- the knowledge derivation strategies
  1. lesson strategies
  2. dialogue strategies
     - the socratic method for the teacher
  3. local strategies
     - the conflict management for the student
<table>
<thead>
<tr>
<th>Points 1/3</th>
<th>Architecture, Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>General architecture</td>
</tr>
</tbody>
</table>
Scheme

Architecture

World

Strategies
- Dialogue: Explain a predicate
- Explain an implication

Local:
- Conflicts management

Lessons:
- Learning
- Teaching

Functional Roles

Teacher Agent
- Knowledge base
- Lesson to be taught
- Model of oneself
- Student’s Model

Student Agent
- Knowledge base
- Model of oneself
- Teacher’s Model
Architecture, Implementation

- General architecture
- Implementation
Diagram class

Strategies:
+DialogueStrategy1()
+DialogueStrategy2()
+...
+LessonStrategy1()
+LessonStrategy2()
+...
+LocalStrategy1()
+LocalStrategy2()
+...

FunctionalRoles:
+ask-for-knowledge(F)
+give-knowledge(F)
+ask-for-information(F)
+give-information(F)
+...
+say-satisfaction()
+say-dissatisfaction()

CognitiveAgent:
+name

World:
+bootstrap()
+sendMessage(agent,message)

KnowledgeBase:
+formulas
+addImplication(i)
+addFact(f)
+delImplication(i)
+delFact(f)
+hasImplication(i)
+hasFact(f)
+...

Teacher:
+Teaches

Student:
+knows

Teaches:

knows:

lives in 0..*

uses:

knows 1

uses 1..*

1
General architecture

Implementation

An example: a conflict management
Example: a conflict management

Teacher’s KB:

- $\text{living}_\text{being}(x) \rightarrow \text{reproduce}(x)$
- $\text{animal}(x) \rightarrow \text{living}_\text{being}(x)$
- $\text{human}(x) \rightarrow \text{mortal}(x)$
- $\text{human}(x) \rightarrow \text{living}_\text{being}(x)$
- $\text{human}(x) \rightarrow \text{reproduce}(x)$

Student’s KB:

- $\text{human}(x) \rightarrow \text{animal}(x)$
- $\text{animal}(x) \rightarrow \text{living}_\text{being}(x)$
- $\text{living}_\text{being}(x) \rightarrow \text{not}(\text{reproduce}(x))$
Example: a conflict management

Teacher's KB:

- living_being(x) → reproduce(x)
- animal(x) → living_being(x)
- human(x) → mortal(x)
- human(x) → living_being(x)
- human(x) → reproduce(x)

Student's KB:

- human(x) → animal(x)
- animal(x) → living_being(x)
- living_being(x) → not(reproduce(x))

Teacher - give-knowledge(human(x) → mortal(x))
## Example: a conflict management

**Teacher’s KB:**

- `living_being(x) → reproduce(x)`
- `animal(x) → living_being(x)`
- `human(x) → mortal(x)`
- `human(x) → living_being(x)`
- `human(x) → reproduce(x)`

**Student’s KB:**

- `human(x) → animal(x)`
- `animal(x) → living_being(x)`
- `living_being(x) → not(reproduce(x))`
- `human(x) → mortal(x)`

Teacher - `give-knowledge(human(x) → mortal(x))`

Student - `say-satisfaction()`
Example: a conflict management

Teacher’s KB:

\[
\begin{align*}
\text{living}_\text{being}(x) & \rightarrow \text{reproduce}(x) \\
\text{animal}(x) & \rightarrow \text{living}_\text{being}(x) \\
\text{human}(x) & \rightarrow \text{mortal}(x) \\
\text{human}(x) & \rightarrow \text{living}_\text{being}(x) \\
\text{human}(x) & \rightarrow \text{reproduce}(x)
\end{align*}
\]

Student’s KB:

\[
\begin{align*}
\text{human}(x) & \rightarrow \text{animal}(x) \\
\text{animal}(x) & \rightarrow \text{living}_\text{being}(x) \\
\text{living}_\text{being}(x) & \rightarrow \text{not(\text{reproduce}(x))} \\
\text{human}(x) & \rightarrow \text{mortal}(x)
\end{align*}
\]

Teacher - give-knowledge(\text{human}(x) \rightarrow \text{mortal}(x))
Student - say-satisfaction()
Teacher - give-knowledge(\text{human}(x) \rightarrow \text{living}_\text{being}(x))
Example: a conflict management

Teacher’s KB:

- \( \text{living} \_ \text{being}(x) \rightarrow \text{reproduce}(x) \)
- \( \text{animal}(x) \rightarrow \text{living} \_ \text{being}(x) \)
- \( \text{human}(x) \rightarrow \text{mortal}(x) \)
- \( \text{human}(x) \rightarrow \text{living} \_ \text{being}(x) \)
- \( \text{human}(x) \rightarrow \text{reproduce}(x) \)

Student’s KB:

- \( \text{human}(x) \rightarrow \text{animal}(x) \)
- \( \text{animal}(x) \rightarrow \text{living} \_ \text{being}(x) \)
- \( \text{living} \_ \text{being}(x) \rightarrow \neg \text{reproduce}(x) \)
- \( \text{human}(x) \rightarrow \text{mortal}(x) \)

Teacher - give-knowledge(\( \text{human}(x) \rightarrow \text{mortal}(x) \))
Student - say-satisfaction()
Teacher - give-knowledge(\( \text{human}(x) \rightarrow \text{living} \_ \text{being}(x) \))
Student - say-satisfaction()
Example: a conflict management

<table>
<thead>
<tr>
<th>Teacher’s KB:</th>
<th>Student’s KB:</th>
</tr>
</thead>
<tbody>
<tr>
<td>( living_being(x) \rightarrow reproduce(x) )</td>
<td>( human(x) \rightarrow animal(x) )</td>
</tr>
<tr>
<td>( animal(x) \rightarrow living_being(x) )</td>
<td>( animal(x) \rightarrow living_being(x) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow mortal(x) )</td>
<td>( living_being(x) \rightarrow not(reproduce(x)) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow living_being(x) )</td>
<td>( human(x) \rightarrow mortal(x) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow reproduce(x) )</td>
<td></td>
</tr>
</tbody>
</table>

Teacher - give-knowledge(\( human(x) \rightarrow mortal(x) \))
Student - say-satisfaction()
Teacher - give-knowledge(\( human(x) \rightarrow living\_being(x) \))
Student - say-satisfaction()
Teacher - give-knowledge(\( human(x) \rightarrow reproduce(x) \))
Example: a conflict management

Teacher's KB:

\[
\begin{align*}
\text{living\_being}(x) & \rightarrow \text{reproduce}(x) \\
\text{animal}(x) & \rightarrow \text{living\_being}(x) \\
\text{human}(x) & \rightarrow \text{mortal}(x) \\
\text{human}(x) & \rightarrow \text{living\_being}(x) \\
\text{human}(x) & \rightarrow \text{reproduce}(x)
\end{align*}
\]

Student's KB:

\[
\begin{align*}
\text{human}(x) & \rightarrow \text{animal}(x) \\
\text{animal}(x) & \rightarrow \text{living\_being}(x) \\
\text{living\_being}(x) & \rightarrow \text{not(\text{reproduce}(x))} \\
\text{human}(x) & \rightarrow \text{mortal}(x)
\end{align*}
\]

Teacher - give-knowledge(\text{human}(x) \rightarrow \text{mortal}(x))
Student - say-satisfaction()
Teacher - give-knowledge(\text{human}(x) \rightarrow \text{living\_being}(x))
Student - say-satisfaction()
Teacher - give-knowledge(\text{human}(x) \rightarrow \text{reproduce}(x))
   Student - ask-for-information(\text{human}(x) \rightarrow \text{animal}(x))
### Example: a conflict management

<table>
<thead>
<tr>
<th>Teacher's KB:</th>
<th>Student's KB:</th>
</tr>
</thead>
<tbody>
<tr>
<td>( living_being(x) \rightarrow reproduce(x) )</td>
<td>( human(x) \rightarrow animal(x) )</td>
</tr>
<tr>
<td>( animal(x) \rightarrow living_being(x) )</td>
<td>( animal(x) \rightarrow living_being(x) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow mortal(x) )</td>
<td>( living_being(x) \rightarrow not(reproduce(x)) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow living_being(x) )</td>
<td>( human(x) \rightarrow mortal(x) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow reproduce(x) )</td>
<td></td>
</tr>
</tbody>
</table>

Teacher - give-knowledge(\( human(x) \rightarrow mortal(x) \))
Student - say-satisfaction()
Teacher - give-knowledge(\( human(x) \rightarrow living\_being(x) \))
Student - say-satisfaction()
Teacher - give-knowledge(\( human(x) \rightarrow reproduce(x) \))
  Student - ask-for-information(\( human(x) \rightarrow animal(x) \))
  Teacher - give-information(\( Unknown \))
Example: a conflict management

Teacher's KB:

\[
\begin{align*}
\text{living\_being}(x) & \rightarrow \text{reproduce}(x) \\
\text{animal}(x) & \rightarrow \text{living\_being}(x) \\
\text{human}(x) & \rightarrow \text{mortal}(x) \\
\text{human}(x) & \rightarrow \text{living\_being}(x) \\
\text{human}(x) & \rightarrow \text{reproduce}(x)
\end{align*}
\]

Student's KB:

\[
\begin{align*}
\text{human}(x) & \rightarrow \text{animal}(x) \\
\text{animal}(x) & \rightarrow \text{living\_being}(x) \\
\text{living\_being}(x) & \rightarrow \text{not} (\text{reproduce}(x)) \\
\text{human}(x) & \rightarrow \text{mortal}(x)
\end{align*}
\]

Teacher - give-knowledge(\text{human}(x) \rightarrow \text{mortal}(x))
Student - say-satisfaction()
Teacher - give-knowledge(\text{human}(x) \rightarrow \text{living\_being}(x))
Student - say-satisfaction()
Teacher - give-knowledge(\text{human}(x) \rightarrow \text{reproduce}(x))
   Student - ask-for-information(\text{human}(x) \rightarrow \text{animal}(x))
   Teacher - give-information(\text{Unknown})
   Student - ask-for-information(\text{animal}(x) \rightarrow \text{living\_being}(x))

Example: a conflict management

Teacher's KB:

<table>
<thead>
<tr>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>living_being(x) \rightarrow reproduce(x)</td>
</tr>
<tr>
<td>animal(x) \rightarrow living_being(x)</td>
</tr>
<tr>
<td>human(x) \rightarrow mortal(x)</td>
</tr>
<tr>
<td>human(x) \rightarrow living_being(x)</td>
</tr>
<tr>
<td>human(x) \rightarrow reproduce(x)</td>
</tr>
</tbody>
</table>

Student's KB:

<table>
<thead>
<tr>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>human(x) \rightarrow animal(x)</td>
</tr>
<tr>
<td>animal(x) \rightarrow living_being(x)</td>
</tr>
<tr>
<td>living_being(x) \rightarrow not(reproduce(x))</td>
</tr>
<tr>
<td>human(x) \rightarrow mortal(x)</td>
</tr>
</tbody>
</table>

Teacher - give-knowledge(human(x) \rightarrow mortal(x))
Student - say-satisfaction()
Teacher - give-knowledge(human(x) \rightarrow living_being(x))
Student - say-satisfaction()
Teacher - give-knowledge(human(x) \rightarrow reproduce(x))
  Student - ask-for-information(human(x) \rightarrow animal(x))
  Teacher - give-information(Unknown)
  Student - ask-for-information(animal(x) \rightarrow living_being(x))
  Teacher - give-information(True)
**Example**

### a conflict management

<table>
<thead>
<tr>
<th>Teacher's KB:</th>
<th>Student's KB:</th>
</tr>
</thead>
<tbody>
<tr>
<td>( living_being(x) \rightarrow reproduce(x) )</td>
<td>( human(x) \rightarrow animal(x) )</td>
</tr>
<tr>
<td>( animal(x) \rightarrow living_being(x) )</td>
<td>( animal(x) \rightarrow living_being(x) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow mortal(x) )</td>
<td>( living_being(x) \rightarrow not(reproduce(x)) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow living_being(x) )</td>
<td>( human(x) \rightarrow mortal(x) )</td>
</tr>
<tr>
<td>( human(x) \rightarrow reproduce(x) )</td>
<td></td>
</tr>
</tbody>
</table>

Teacher - give-knowledge\( (human(x) \rightarrow mortal(x)) \)
Student - say-satisfaction()
Teacher - give-knowledge\( (human(x) \rightarrow living\_being(x)) \)
Student - say-satisfaction()
Teacher - give-knowledge\( (human(x) \rightarrow reproduce(x)) \)
  Student - ask-for-information\( (human(x) \rightarrow animal(x)) \)
  Teacher - give-information\( (Unknown) \)
  Student - ask-for-information\( (animal(x) \rightarrow living\_being(x)) \)
  Teacher - give-information\( (True) \)
  Student - ask-for-information\( (living\_being(x) \rightarrow not(reproduce(x))) \)
Example: a conflict management

Teacher's KB:

- `living_being(x) → reproduce(x)`
- `animal(x) → living_being(x)`
- `human(x) → mortal(x)`
- `human(x) → living_being(x)`
- `human(x) → reproduce(x)`

Student's KB:

- `human(x) → animal(x)`
- `animal(x) → living_being(x)`
- `living_being(x) → not(reproduce(x))`
- `human(x) → mortal(x)`

Teacher - give-knowledge(`human(x) → mortal(x)`)
Student - say-satisfaction()
Teacher - give-knowledge(`human(x) → living_being(x)`)
Student - say-satisfaction()
Teacher - give-knowledge(`human(x) → reproduce(x)`)
  - Student - ask-for-information(`human(x) → animal(x)`)
  - Teacher - give-information(`Unknown`)
  - Student - ask-for-information(`animal(x) → living_being(x)`)
  - Teacher - give-information(`True`)
  - Student - ask-for-information(`living_being(x) → not(reproduce(x))`)
  - Teacher - give-information(`False`)
### Example: a conflict management

<table>
<thead>
<tr>
<th>Teacher's KB:</th>
<th>Student's KB:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{living_being}(x) \rightarrow \text{reproduce}(x)$</td>
<td>$\text{human}(x) \rightarrow \text{animal}(x)$</td>
</tr>
<tr>
<td>$\text{animal}(x) \rightarrow \text{living_being}(x)$</td>
<td>$\text{animal}(x) \rightarrow \text{living_being}(x)$</td>
</tr>
<tr>
<td>$\text{human}(x) \rightarrow \text{mortal}(x)$</td>
<td>$\text{living_being}(x) \rightarrow \neg\text{reproduce}(x)$</td>
</tr>
<tr>
<td>$\text{human}(x) \rightarrow \text{living_being}(x)$</td>
<td>$\text{human}(x) \rightarrow \text{mortal}(x)$</td>
</tr>
<tr>
<td>$\text{human}(x) \rightarrow \text{reproduce}(x)$</td>
<td></td>
</tr>
</tbody>
</table>

Teacher - give-knowledge($\text{human}(x) \rightarrow \text{mortal}(x)$)
Student - say-satisfaction()
Teacher - give-knowledge($\text{human}(x) \rightarrow \text{living\_being}(x)$)
Student - say-satisfaction()
Teacher - give-knowledge($\text{human}(x) \rightarrow \text{reproduce}(x)$)
  Student - ask-for-information($\text{human}(x) \rightarrow \text{animal}(x)$)
  Teacher - give-information($\text{Unknown}$)
  Student - ask-for-information($\text{animal}(x) \rightarrow \text{living\_being}(x)$)
  Teacher - give-information($\text{True}$)
  Student - ask-for-information($\text{living\_being}(x) \rightarrow \neg\text{reproduce}(x)$)
  Teacher - give-information($\text{False}$)
Example: a conflict management

Teacher's KB:

\[ living\text{-}being(x) \rightarrow reproduce(x) \]
\[ animal(x) \rightarrow living\text{-}being(x) \]
\[ human(x) \rightarrow mortal(x) \]
\[ human(x) \rightarrow living\text{-}being(x) \]
\[ human(x) \rightarrow reproduce(x) \]

Student's KB:

\[ human(x) \rightarrow animal(x) \]
\[ animal(x) \rightarrow living\text{-}being(x) \]
\[ living\text{-}being(x) \rightarrow not(reproduce(x)) \]
\[ human(x) \rightarrow mortal(x) \]
\[ human(x) \rightarrow reproduce(x) \]

Teacher - give-knowledge(\(human(x) \rightarrow mortal(x)\))
Student - say-satisfaction()
Teacher - give-knowledge(\(human(x) \rightarrow living\text{-}being(x)\))
Student - say-satisfaction()
Teacher - give-knowledge(\(human(x) \rightarrow reproduce(x)\))
  Student - ask-for-information(\(human(x) \rightarrow animal(x)\))
  Teacher - give-information(\(Unknown\))
  Student - ask-for-information(\(animal(x) \rightarrow living\text{-}being(x)\))
  Teacher - give-information(\(True\))
  Student - ask-for-information(\(living\text{-}being(x) \rightarrow not(reproduce(x))\))
  Teacher - give-information(\(False\))
## Example: a conflict management

<table>
<thead>
<tr>
<th>Teacher's KB:</th>
</tr>
</thead>
<tbody>
<tr>
<td>living_being(x) → reproduce(x)</td>
</tr>
<tr>
<td>animal(x) → living_being(x)</td>
</tr>
<tr>
<td>human(x) → mortal(x)</td>
</tr>
<tr>
<td>human(x) → living_being(x)</td>
</tr>
<tr>
<td>human(x) → reproduce(x)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student's KB:</th>
</tr>
</thead>
<tbody>
<tr>
<td>human(x) → animal(x)</td>
</tr>
<tr>
<td>animal(x) → living_being(x)</td>
</tr>
<tr>
<td>living_being(x) → not(reproduce(x))</td>
</tr>
<tr>
<td>human(x) → mortal(x)</td>
</tr>
<tr>
<td>human(x) → reproduce(x)</td>
</tr>
</tbody>
</table>

**Teacher - give-knowledge(\textit{human(x) → mortal(x)})**

**Student - say-satisfaction()**

**Teacher - give-knowledge(\textit{human(x) → living\_being(x)})**

**Student - say-satisfaction()**

**Teacher - give-knowledge(\textit{human(x) → reproduce(x)})**

  **Student - ask-for-information(\textit{human(x) → animal(x)})**
  **Teacher - give-information(\textit{Unknown})**
  **Student - ask-for-information(\textit{animal(x) → living\_being(x)})**
  **Teacher - give-information(\textit{True})**
  **Student - ask-for-information(\textit{living\_being(x) → not(reproduce(x)))**
  **Teacher - give-information(\textit{False})**

**Student - say-satisfaction()**
Conclusions

- Dialogue: efficient learning method
- Introduced notion: the Connexity of a KB
- The Functional Roles theory as modeller
- The potential of curious students
- Use of strategies to derive knowledge

Perspectives

- enrich the formulas type
- define new strategies
- use our learning method in conjunction with other methods