Available at www.ComputerScienceWeb.com POWERED BY SCIENCE DIRECT•

ELSEVIER

Cognitive Systems Research 4 (2003) 119–136

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

# Modelling the dynamics of reasoning processes: Reasoning by assumption

Action editor: Vasant Honavar

Catholijn M. Jonker[a], Jan Treur[a,b,*]

[a]Vrije Universiteit Amsterdam, Department of Artificial Intelligence, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
[b]Utrecht University, Department of Philosophy, Heidelberglaan 8, 3584 CS Utrecht, The Netherlands

Received 16 July 2002; accepted 10 November 2002

**Abstract**

To model the dynamics of cognitive processes, often the dynamical systems theory (DST) is advocated. However, for higher cognitive processes such as reasoning and certain forms of natural language processing the techniques adopted within DST are not very adequate. This paper shows how an analysis of the dynamics of reasoning processes can be made using techniques different from those of DST. The approach makes use of temporal traces consisting of sequences of reasoning states over time to describe reasoning processes. It is shown for the example reasoning pattern 'reasoning by assumption', how relevant dynamic properties can be identified and expressed using a temporal trace language. Example traces have been acquired in two ways. Firstly, empirical traces have been generated based on think-aloud protocols of a subject solving a number of cases of a reasoning puzzle. Secondly, a simulation model has been developed and executed for the same cases of the reasoning puzzle. For all these traces, the dynamic properties can and have been analysed automatically, using a software environment that has been developed. Thus the feasibility of the approach was shown.
© 2003 Elsevier Science B.V. All rights reserved.

*Keywords:* Reasoning processes; Reasoning by assumption; Dynamic properties

## 1. Introduction

Within Cognitive Science in recent years the dynamical perspective on cognitive phenomena has been emphasized and received much attention. In most literature focusing on the dynamics of cognition, the dynamical systems theory (DST) is taken as a point of departure (e.g. Kelso, 1995; Port & van Gelder, 1995). This theory assumes that, in contrast to the use of symbolic representations, modelling dynamics of cognitive phenomena can be done more effectively by using representations based on real numbers and mathematical techniques from calculus; it offers mathematical simulation and analysis techniques from the area of difference and differential equations. The many convincing examples that have

*Corresponding author. Tel.: +31-20-444-7763; fax: +31-20-444-7653.
*E-mail addresses:* jonker@cs.vu.nl (C.M. Jonker), http://www.cs.vu.nl/~jonker (C.M. Jonker), treur@cs.vu.nl (J. Treur), http://www.cs.vu.nl/~treur (J. Treur).

been used to illustrate the usefulness of this perspective often address lower level cognitive processes such as sensory or motor processing. Indeed one of the advantages of the dynamical systems theory is that it is able to model the temporal aspects of events taking place on a continuous time scale, such as, for example, recognition time, response time, and time involved in motor patterns and locomotion.

Also some examples of higher level cognitive processes have been addressed using DST; for example, the dynamic models for decision making developed in Busemeyer and Townsend (1993). Especially the continuous adaptive aspects of the decision making are covered nicely in this approach. Areas for which the quantitative approach based on DST is assumed to have less to offer are the dynamics of higher level processes that are considered to have mainly a qualitative character, such as certain capabilities of language processing and reasoning. This evaluation is based on three assumptions: (1) if dynamics of cognitive processes is to be modelled, then DST is the appropriate approach, and (2) DST is based on the combination of two principles or commitments, the first one of which is a methodological or philosophical commitment, and the other one is a commitment to a certain package of techniques to be used: (a) modelling cognitive phenomena requires modelling their dynamics, and (b) modelling dynamics of cognitive phenomena requires mathematical techniques based on difference and differential equations, and (3) phenomena where qualitative aspects are considered dominant cannot be adequately modelled using difference or differential equations.

In this paper the position is taken that, in contrast to assumption (1) above, due to its commitment to quantitative representations and techniques, DST is not always the most adequate possibility to model dynamics of cognitive phenomena. In the last two decades, within the areas of Computer Science and Artificial Intelligence, alternative techniques have been developed to model the dynamics of phenomena using qualitative means. Examples are process algebra, dynamic and temporal logic, event, situation and fluent calculus (e.g. van Eck et al., 2001; Hölldobler & Thielscher, 1990; Kowalski & Sergot, 1986; McCarthy & Hayes, 1969). Just as difference or differential equations, these alternative techniques allow to express temporal relations, i.e. relations between a state of a process at one point in time, and states at other points in time. In contrast, the form in which these temporal relations are expressed can cover symbolic and non-quantitative aspects as well. To illustrate the usefulness of such an approach for higher level cognitive phenomena, the dynamics of a practical reasoning pattern is addressed: reasoning by assumption. For this reasoning pattern both analysis of human reasoning protocols and agent-based simulation of reasoning patterns have been performed.

The language used to express dynamic properties is formal but not based on calculus. It allows for precise specification of these dynamic properties, covering both qualitative and quantitative aspects of states and temporal relations. Moreover, software tools can and actually have been developed to: (1) support specification of dynamic properties, and (2) automatically check specified dynamic properties against example traces to find out whether they hold. This provides a useful supporting software environment to evaluate empirical data on the dynamics of cognitive processes. In the paper it is shown how dynamic properties of think-aloud protocols of reasoning patterns can be checked automatically using this software environment.

In Section 2 the dynamic perspective on reasoning is discussed in some more detail, and focused on the pattern 'reasoning by assumption'. Section 3 addresses some more details of the language used. Section 4 presents a number of the dynamic properties that have been identified for patterns of reasoning by assumption. Section 5 discusses empirical validation. Here it is shown how existing think-aloud protocols involving reasoning by assumption can be formalised to reasoning traces. For these reasoning traces a number of the dynamic properties have been (automatically) checked. In Section 6 a similar analysis of the reasoning traces generated by a simulation model is presented. In Section 7 the results are compared and discussed.

## 2. A model for the dynamics of reasoning

In history, formalisation of the cognitive capability to perform reasoning has been addressed from different areas and angles: Philosophy, Logic, Cognitive Science, Artificial Intelligence. Within Philosophy and Logic much emphasis has been put on the results (conclusions) of a reasoning process, abstracting from the process by which such a result is found: when is a statement a valid conclusion, given a certain set of premises. Within Artificial Intelligence, much emphasis has been put on effective inference procedures to automate reasoning processes. The dynamics of such inference procedures usually is described in a procedural, algorithmic manner; dynamics are not described and analysed in a conceptual, declarative manner. Within Cognitive Science, reasoning is often addressed from within one of the two dominant streams: the syntactic approach (based on inference rules applied to syntactic expressions, as common in the logic-based approach (e.g. Braine & O'Brien, 1998; Rips, 1994)), or the semantic approach (based on construction of mental models) (e.g. Johnson-Laird, 1983; Johnson-Laird & Byrne, 1991; Yang & Johnson-Laird, 1999; Yang & Bringsjord, 2001; Schroyens, Schaeken, & d'Ydewalle, 2001). Especially this second approach provides a wider scope than the scope usually taken within logic. Formalisation and formal analysis of the dynamics within these approaches has not been developed in depth yet.

To understand a specific reasoning process, especially for practical reasoning in humans, the dynamics are important. In particular, for reasoning processes in natural contexts, which are usually not restricted to simple deduction, also dynamic aspects play an important role and have to be taken into account, such as dynamically posing goals for the reasoning, or making (additional) assumptions during the reasoning, thus using a dynamic set of premises within the reasoning process. Decisions made during the process, for example, on which reasoning goal to pursue, or which assumptions to make, are an inherent part of such a reasoning process. Such reasoning processes or their outcomes cannot be understood, justified or explained to others without taking into account these dynamic aspects.

The approach to the semantical formalisation of the dynamics of reasoning presented in Section 2 is based on the concepts reasoning state, transitions between reasoning states, and reasoning traces: traces of reasoning states. To specify dynamic properties of a reasoning process, in Section 3 a language is introduced in which it is possible to express properties of reasoning traces.

### 2.1. Reasoning state

A reasoning state formalises an intermediate state of a reasoning process. It may include information on different aspects of the reasoning process, such as content information or control information. Within a syntactical inference approach, a reasoning state includes the set of statements derived (or truth values of these statements) at a certain point in time. Within a semantical approach based on mental models, a reasoning state may include a particular mental model constructed at some point in time, or a set of mental models representing the considered possibilities. However, also additional (meta-)information can be included in a reasoning state, such as control information indicating what is the focus or goal of the reasoning, or information on which statements have been assumed during the reasoning. Moreover, to be able to cover interaction between reasoning and the external world, also part of the state of the external world is included in a reasoning state. This can be used, for example, to model the presentation of a reasoning puzzle to a subject, or to model the subject's observations in the world. The set of all reasoning states is denoted by $\mathsf{RS}$.

### 2.2. Transition of reasoning states

A transition of reasoning states, i.e. an element $\langle \mathsf{S}, \mathsf{S}' \rangle$ of $\mathsf{RS} \times \mathsf{RS}$, defines a step from one reasoning state to another reasoning state; this formalises one reasoning step. A *reasoning transition relation* is a set of these transitions, or a relation on $\mathsf{RS} \times \mathsf{RS}$. Such a relation can be used to specify the allowed transitions within a specific type of reasoning. Within a syntactical approach, inference rules such as modus ponens typically define transitions between reasoning states. For example, if two statements

$$p, p \rightarrow q$$

are included in a reasoning state, then by a modus ponens transition, a reasoning state can be created where, in addition, also

$$q$$

is included. Within a semantical approach a construction step of a mental model, after a previous mental model, defines a transition between reasoning states. For example, if knowledge 'if p then q' is available, represented in a mental state

[p], q

and in addition not-q is presented, then a transition may occur to a reasoning state consisting of a set of mental models

p, q; ~p, ~q; ~p, q

which represents the set of possibilities considered; a next transition may be selection of the possibility that fits not-q, leading to the reasoning state

~p, ~q

### 2.3. Reasoning trace

Reasoning dynamics or reasoning behaviour is the result of successive transitions from one reasoning state to another. By applying transitions in succession, a time-indexed sequence of reasoning states $(\gamma_t)t \in T$ is constructed, where T is the time frame used (e.g. the natural numbers). A reasoning trace, created in this way, is a sequence of reasoning states over time, i.e. an element of $RS^T$. Traces are sequences of reasoning states such that each pair of successive reasoning states in this trace forms an allowed transition, as has been defined under transitions. A trace formalises one specific line of reasoning. A set of reasoning traces is a declarative description of the semantics of the behaviour of a reasoning process; each reasoning trace can be seen as one of the alternatives for the behaviour.

### 2.4. Reasoning by assumption

The specific reasoning pattern used in this paper to illustrate the approach is 'reasoning by assumption'. This type of reasoning often occurs in practical reasoning; for example, in

- diagnostic reasoning based on causal knowledge;
- everyday reasoning;
- reasoning based on natural deduction.

An example of diagnostic reasoning by assumption in the context of a car that won't start is:

> *Suppose the battery is empty, then the lights won't work. But if I try, the lights turn out to work. Therefore the battery is not empty.*

Note that on the basis of the assumption that the battery is empty, and causal knowledge that without a functioning battery the lights will not burn, a prediction is made on an observable world fact, namely that the lights will not burn. After this an observation is initiated which has a result (lights do burn) that contradicts the prediction. Based on this outcome the assumption is evaluated and, as a result, rejected.

An example of an everyday process of reasoning by assumption is:

> *Suppose I do not take my umbrella with me. Then, if it starts raining at* 5 *pm, I will get wet, which I don't want. Therefore I better take my umbrella with me.*

Again, based on the assumption some prediction is made, this time using probabilistic knowledge that it may rain at 5 pm. The prediction is in conflict with the desire not to get wet. The assumption is evaluated and rejected.

Examples of reasoning by assumption in natural deduction are as follows.

*Reductio ad absurdum or method of indirect proof*
After assuming A, I have derived a contradiction. Therefore I can derive not A.

*Implication introduction*
After assuming A, I have derived B. Therefore I can derive that A implies B.

*Reasoning by cases*
After assuming A, I have derived C. Also, after assuming B, I derived C. Therefore I can derive C from A or B.

Notice that as a common pattern in all of the examples presented, it seems that first a reasoning state is entered in which some fact is *assumed*. Next (possibly after some intermediate steps) a reasoning state is entered where *consequences* of this assumption have been *predicted*. Moreover, in some cases *observations* can be performed obtaining additional information about the world to be included in a next reasoning state. Finally, a reasoning state is entered in which an *evaluation* has taken place, for example, resulting in rejection of the assumption; possibly in the next state the assumption actually is retracted, and further conclusions are added.

This first analysis already shows some peculiarities of this type of reasoning. Within a reasoning state not (only) content information is included, but within the reasoning a major role is played by different types of (meta-)information on the status of other information; this meta-information goes beyond direct content information. For example, the following types of meta-information can be included in a *reasoning state*:

- which assumption has been made;
- which predictions have been made based on an assumption;
- which information is observation information;
- which evaluation has been made.

The examples also show that the reasoning transitions that take place are of different types; for example:

- from a reasoning state without an assumption to a reasoning state with an assumption;
- from a reasoning state with an assumption to a reasoning state with a prediction;
- from a reasoning state with a prediction to a reasoning state with an observation result;
- from a reasoning state with an assumption, a prediction, and an observation result (or other comparison information) to a reasoning state with an evaluation of the assumption, e.g. that it is rejected.

Reasoning traces in the examples suggest a number of such reasoning states and transitions.

To explore the usefulness of the presented model for reasoning dynamics, in Section 5 a simple version of a reasoning puzzle is used: the wise persons puzzle. This puzzle as considered here requires two wise persons (A and B) and two hats. Each wise person is wearing a hat, of which the colour is unknown. Both wise persons know that:

- hats can be white or black;
- there is at least one white hat;
- they can observe the colour of each other's hat;
- if, after reasoning, a person knows the colour of its own hat (s)he will tell the other that colour;
- if, after reasoning, a person does not know the colour of its own hat (s)he will tell so to the other;
- communications are limited to comments regarding the colour of the person's own hat;
- they both reason fully logically.

If, for example, both persons have a white hat and wise person A is asked whether he knows the colour of his hat, then A must answer that he does not know. On the basis of this knowledge, wise person B can then reason that his/her own hat is white. A solution of this reasoning puzzle is obtained if wise person B dynamically introduces and rejects assumptions about the colour of his/her own hat. For example, in the case that B sees that A has a white hat, and B hears that A says (s)he does not know the colour, B can have the following reasoning trace:

0. observation results: A's hat is white; A says (s)he does not know the colour;
1. assumption that B's own hat is black;
2. prediction that A knows (s)he has white;
3. evaluation that the prediction contradicts the observation result; the assumption is to be rejected;
4. no assumption anymore that B's own hat is black;
5. assumption that B's own hat is white.

## 3. A temporal trace language to express dynamic properties

To specify properties on the dynamics of a reasoning process, the temporal trace language TTL used in Herlea, Jonker, Treur and Wijngaards (1999) and Jonker and Treur (1998) is adopted. This is a language in the family of languages to which also situation calculus (McCarthy & Hayes, 1969), event calculus (Kowalski & Sergot, 1986), and fluent calculus (Hölldobler & Thielscher, 1990) belong.

An *ontology* is a specification (in order-sorted logic) of a vocabulary, i.e. a signature. For the example reasoning pattern 'reasoning by assumption' the state ontology includes binary relations such as

assumed,
predicted,
rejected,
observation_result,
holds_in_world

on sorts INFO_ELEMENT × SIGN. The sort INFO_ELEMENT includes specific domain statements such as

hat_colour(white, self),
conclusion(my_hat_is_white, other),
conclusion(don't_know_my_colour, other).

The sort SIGN consists of the elements pos and neg. Using this state ontology, for example the following state properties can be expressed:

holds_in_world(hat_colour(white, self), pos)
assumed(hat_colour(white, self), neg)
prediction_for(conclusion (my_hat_is_white, other), pos, hat_colour(white, self), pos)
rejected(hat_colour(white, self), pos)

A (*reasoning*) *state* for ontology Ont is characterised by the properties expressed in Ont which are true. This is formalised by an assignment of truth values {true, false} to the set of ground atoms At(Ont). A part of the description of an example reasoning state S is the following:

| | |
|---|---|
| holds_in_world(hat_colour(white, self), pos) | :true |
| assumed(hat_colour(white, self), neg) | :true |
| prediction_for(conclusion(my_hat_is_white, other), pos, hat_colour(white, self), neg) | :true |
| observation_result(conclusion (don't_know_my_colour, other), pos) | :true |
| rejected(hat_colour(white, self), neg) | :false |

An alternative, but equivalent notation for such a reasoning state leaves out all value assignments true, and indicates the value assignments false by a ~ symbol in front of the property (notation):

holds_in_world(hat_colour(white, self), pos)
assumed(hat_colour(white, self), neg)
prediction_for(conclusion(my_hat_is_white, other), pos, hat_colour(white, self), pos)
observation_result(conclusion (don't_know_my_colour, other), pos)
~rejected(hat_colour(white, self), neg)

The *set of all possible states* for ontology Ont is denoted by STATES(Ont). By RS the sort of all reasoning states of the agent is denoted. As indicated earlier, world states are considered substates of reasoning states, and iws is a unary predicate on RS that defines that the world state within a reasoning state belongs to the set of *intended world states*. In the wise person example this specification iws(S) is defined byg

iws(S):
not [S |== holds_in_world(hat_colour(white, self), neg) ∧
S |== olds_in_world(hat_colour(white, other), neg)]

which expresses that the situation with two black hats is to be excluded. So, for example, the world state

holds_in_world(hat_colour(white, self), pos)
holds_in_world(hat_colour(white, other), neg)

is one of the intended world states, whereas

holds_in_world(hat_colour(white, self), neg)
holds_in_world(hat_colour(white, other), neg)

indicates a not intended world state (the forbidden black–black situation).

The standard satisfaction relation |== between states and state properties is used: S |== p means that state property p holds in state S. For example, in the reasoning state S above it holds

S |== assumed(hat_colour(white, self), neg).

To describe dynamics, explicit reference is made to time in a formal manner. A fixed *time frame* T is assumed which is linearly ordered. Depending on the application, it may be dense (e.g. the real numbers), or discrete (e.g. the set of integers or natural numbers or a finite initial segment of the natural numbers), or any other form, as long as it has a linear ordering. A *trace* $\gamma$ over an ontology Ont and time frame T (e.g. the natural numbers) is a mapping

$$\gamma: \text{T} \rightarrow \text{STATES(Ont)},$$

i.e. a time-indexed sequence of reasoning states

$$\gamma_t (t \in \text{T})$$

in STATES(Ont). The set of all traces over ontology Ont is denoted by TRACES(Ont), i.e. TRACES(Ont) = $\text{STATES(Ont)}^{\text{T}}$. The set TRACES(Ont) is also denoted by $\Gamma$. Note that to be able to cover observation in the external world as well, the (current) world state is part of each reasoning state in each trace.

States of a trace can be related to state properties via the formally defined satisfaction relation $|==$ between states and state formulae. Comparable to the approach in situation calculus, the sorted predicate logic *temporal trace language* TTL is built on atoms referring to traces, time and state properties, such as state($\gamma$, t) $|==$ p. This expression denotes that state property p is true in the state of trace $\gamma$ at time point t. Here $|==$ is a predicate symbol in the language (in infix notation), comparable to the Holds-predicate in situation calculus. Temporal formulae are built using the usual logical connectives and quantification (for example, over traces, time and state properties). The set TFOR(Ont) is the set of all *temporal formulae* that only make use of ontology Ont. We allow additional language elements as abbreviations of formulae of the temporal trace language. An example of such a dynamic property is

$$\begin{aligned}
&\forall\gamma: \Gamma\ \forall t: \text{T}\ \forall A: \text{INFO\_ELEMENT}\ \forall S: \text{SIGN} \\
&\quad [\text{state}(\gamma, t) |== \text{rejected}(A, S) \\
&\quad \Rightarrow [\forall t': \text{T} \geq t: \text{T} \\
&\qquad \text{state}(\gamma, t') |== \text{rejected}(A, S)]
\end{aligned}$$

This persistence property expresses that in any reasoning trace $\gamma$ at any point in time t, if an assumption A has been rejected, then A remains rejected within $\gamma$ for all $t' \geq t$. For more examples of dynamic properties, see Section 4.

The fact that this language is formal allows for precise specification of properties. Moreover, editors can and actually have been developed to support specification of properties. Specified properties can be checked automatically against example traces to find out whether they hold.

For the domain of the wise persons puzzle some world facts can be assumed; others cannot be assumed. Furthermore, there are some relations between domain predicates, like the knowledge that if one agent wears a black hat then the other agent must be wearing a white hat. In order to validate the behaviour of some reasoning agent (human or otherwise) all this information needs to be available. Therefore, two additional predicates are introduced: pa, and is_relevant_for. The unary predicate pa defines the set of possible assumptions that can be made for the specific application. The predicate is_relevant_for can be used, for example, to express that hat_colour(white, other) with truth value true is relevant for hat_colour(white, self) with truth value false. To describe the dynamics of the reasoning process this information is used to express that a certain (observable) prediction is relevant for a certain assumption. In effect, the predicate defines the set of all relevant predictions that regard the observable part of the world state. In the case of the wise person puzzle, the relational facts that together form these sets are defined as follows:

```
pa(hat_colour(white, self), pos)
pa(hat_colour(white, self), neg)
is_relevant_for(conclusion (dont_know_my_colour, other), pos,
  hat_colour(white, self), pos)
is_relevant_for(conclusion (my_hat_is_white, other), pos,
  hat_colour(white, self), neg)
is_relevant_for(hat_colour(white, other), pos,
  hat_colour(white, self), neg)
```

## 4. Characterising dynamic properties

In this section a number of the most relevant of the dynamic properties that have been identified as relevant for patterns of reasoning by assumption are presented in both an informal and formal way. Notice that specifying these dynamic properties does not automatically include at forehand a claim that they hold or should hold for all reasoning traces. The only claim made is that they are relevant or interesting to be considered for a specific reasoning trace in the sense whether or not they are true. Moreover, the properties, although they have been formulated universally quantified for all traces, are to be considered for instances of traces separately.

## 4.1. Global dynamic properties

Global properties address the overall reasoning behaviour of the agent, not the step by step reasoning process of the agent.

### 4.1.1. GP1 termination of the reasoning

This property ensures that the reasoning will not go indefinitely. Formally:

$\forall \gamma: \Gamma\ \exists t: T\ \forall t': T\ t' \geq t \Rightarrow \text{state}(\gamma, t) = \text{state}(\gamma, t')$

In the current formulation, property GP1 demands that the whole agent shows no more reasoning activity. It is possible formulate GP1 in a more precise manner by limiting the inactivity to those parts of the agent involved in the assumption reasoning process.

Based on this property the following abbreviation is defined for use in other properties:

$\text{termination}(\gamma, t) \equiv$

$\forall t': T\ t' \geq t \Rightarrow \text{state}(\gamma, t) = \text{state}(\gamma, t')$

### 4.1.2. GP2 correctness of rejection

Everything that has been rejected does not hold in the world situation

$\forall \gamma: \Gamma\ \forall t: T\ \text{state}(\gamma, t) \models\!= \text{rejected}(A, S)$
$\quad \Rightarrow \text{state}(\gamma, t) \models\!= \text{not holds\_in\_world}(A, S)$

### 4.1.3. GP3 completeness of rejection

After termination, all assumptions that have not been rejected hold in the world situation

$\forall \gamma: \Gamma\ \forall t: T\ \forall A: \text{INFO\_ELEMENT}\ \forall S: \text{SIGN}$
$\quad \text{termination}(\gamma, t)$
$\quad \wedge\ \text{state}(\gamma, t) \models\!= \text{assumed}(A, S)$
$\quad \wedge\ \text{state}(\gamma, t) \not\models\!= \text{rejected}(A, S)$
$\qquad \Rightarrow \text{state}(\gamma, t) \models\!= \text{holds\_in\_world}(A, S)$

### 4.1.4. GP4 guaranteed outcome

This property expresses that a terminated reasoning process with a world state in the set of intended world states has as an outcome at least one evaluated assumption that was not rejected.

$\forall \gamma: \Gamma\ \forall t: T$
$\quad [\text{termination}(\gamma, t) \wedge \text{iws}(\text{state}(\gamma, t))]$
$\quad \Rightarrow [\exists A: \text{INFO\_ELEMENT}, \exists S: \text{SIGN}$
$\qquad \text{state}(\gamma, t) \models\!= \text{assumed}(A, S)$
$\qquad \wedge\ \text{state}(\gamma, t) \not\models\!= \text{rejected}(A, S)]$

### 4.1.5. GP5 persistence

Two types of persistence properties can be defined: unconditional or conditional. The first, unconditional type expresses that once a state property holds in a reasoning state, this property will hold for all future reasoning states. In this unconditional form relevant persistence properties can be expressed for state properties based on

| | |
|---|---|
| holds_in_world | (static world assumption) |
| observation_result | (persistent observations) |
| rejected | (once rejected remains rejected) |

Formally, these unconditional persistence properties are expressed as follows.

#### 4.1.5.1. Unconditional persistence properties

$\forall \gamma: \Gamma\ \forall t: T\ \forall A: \text{INFO\_ELEMENT}\ \forall S: \text{SIGN}$
$\quad [\text{state}(\gamma, t) \models\!= \text{holds\_in\_world}(A, S)$
$\quad \Rightarrow [\forall t': T \geq t: T$
$\qquad \text{state}(\gamma, t') \models\!= \text{holds\_in\_world}(A, S)]$

$\forall \gamma: \Gamma\ \forall t: T\ \forall A: \text{INFO\_ELEMENT}\ \forall S: \text{SIGN}$
$\quad [\text{state}(\gamma, t) \models\!= \text{rejected}(A, S)$
$\quad \Rightarrow [\forall t': T \geq t: T$
$\qquad \text{state}(\gamma, t') \models\!= \text{rejected}(A, S)]$

$\forall \gamma: \Gamma\ \forall t: T\ \forall A: \text{INFO\_ELEMENT}\ \forall S: \text{SIGN}$
$\quad \text{state}(\gamma, t) \models\!= \text{observation\_result}(A, S)$
$\quad \Rightarrow [\forall t': T \geq t: T$
$\qquad \text{state}(\gamma, t') \models\!= \text{observation\_result}(A, S)]$

Conditional persistence properties can be specified for assumptions (persistent as long as they are not rejected), and possibly for predictions (persistent as long as the related assumption is not rejected). Formally, these properties are expressed as follows.

#### *4.1.5.2. Conditional persistence properties*

$\forall\gamma: \Gamma\ \forall t, t', t'': T\ \forall A$: INFO_ELEMENT $\forall S$: SIGN
  $t \leq t'' \wedge$ state($\gamma$, t) $\models\!=$ assumed(A, S)
  $\wedge\ \forall t'\ [t \leq t' \leq t'' \Rightarrow$ state($\gamma$, t) $\not\models$ rejected(A, S)]
  $\Rightarrow$ state($\gamma$, t″) $\models\!=$ assumed(A, S)
$\forall\gamma: \Gamma\ \forall t, t', t'': T$
$\forall$A1, A2: INFO_ELEMENT $\forall$S1, S2: SIGN
  $t \leq t'' \wedge$ state($\gamma$, t) $\models\!=$ prediction_for(A1, S1, A2, S2)
  $\wedge\ \forall t'\ [t \leq t' \leq t'' \Rightarrow$ state($\gamma$, t) $\not\models$ rejected(A, S)]
  $\Rightarrow$state($\gamma$, t″) $\models\!=$ prediction_for(A1, S1, A2, S2)

### *4.1.6. GP6 nonintended situations*

If a world situation is nonintended (e.g. the situation with the two black hats), then property GP4 will not give any guarantee. However, it may be possible that the reasoning trace fulfils the property that in such a case all assumptions have been considered and rejected, i.e.

| | |
|---|---|
| If | the reasoning has terminated |
| and | the world situation is not an intended world situation, |
| then | all possible assumptions have rejected. |

Formally:

$\forall\gamma: \Gamma\ \forall t: T$
  termination($\gamma$, t) $\wedge$ not iws(state($\gamma$, t))
  $\Rightarrow$ [$\forall$A: INFO_ELEMENT, $\forall$S: SIGN
    pa(A, S) $\Rightarrow$ state($\gamma$, t) $\models\!=$ rejected(A, S)]

## *4.2. Local dynamic properties*

Global properties describe the overall reasoning behaviour of the agent, in this case applied to solve the wise persons puzzle, but they are not detailed enough to track the dynamics of the reasoning process, say step by step. Local properties each describe a part of the reasoning process.

### *4.2.1. LP1 observation result correctness*

The first property expresses that observations that are obtained from the world, indeed hold in the world.

$\forall\gamma: \Gamma\ \forall t: T\ \forall$A: INFO_ELEMENT $\forall$S: SIGN
  state($\gamma$, t) $\models\!=$ observation_result(A, S)
  $\Rightarrow$ state($\gamma$, t) $\models\!=$ holds_in_world(A, S)

### *4.2.2. LP2 assumption effectiveness*

This property expresses that the reasoning process will go on to generate new assumptions as long as not all of them have been rejected. This guarantees progress in the reasoning process; if no assumption has been found that is not rejected, keep trying new assumptions for as long as possible.

| | |
|---|---|
| If | the world situation is an intended world situation, |
| then | as long as there are assumptions that have not been rejected |
| and | as long as all assumptions that have been made have been rejected, |

the agent will keep generating new assumptions. Formally:

$\forall\gamma: \Gamma\ \forall t: T$
  iws(state($\gamma$, t))
  $\Rightarrow$ [[$\exists$A: INFO_ELEMENT, $\exists$S: SIGN
    pa(A, S) $\wedge$ state($\gamma$, t) $\not\models$ rejected(A, S)]
    $\wedge$ [$\forall$A: INFO_ELEMENT $\forall$S: SIGN $\forall$t1: T
    [t1$\leq$t $\wedge$ state($\gamma$, t1) $\models\!=$ assumed(A, S)]
    $\Rightarrow$ [$\exists$t2: T t1$\leq$t2$\leq$t
      $\wedge$ state($\gamma$, t2) $\models\!=$ rejected(A, S)]]
    $\Rightarrow$ [$\exists$t′: T$\geq$t: T $\exists$A: INFO_ELEMENT $\exists$S: SIGN
      state($\gamma$, t′) $\models\!=$ assumed(A, S)
      $\wedge$ state($\gamma$, t′) $\not\models$ rejected(A, S)]]

### *4.2.3. LP5 prediction effectiveness*

For each assumption the agent makes all relevant predictions about the observable part of the world state

$\forall\gamma: \Gamma\ \forall t: T\ \forall$A: INFO_ELEMENT $\forall$S1: SIGN
  state($\gamma$, t) $\models\!=$ assumed(A, S1)
  $\Rightarrow$ $\forall$B: INFO_ELEMENT $\forall$S2: SIGN
    [is_relevant_for(B, S2, A, S1)
    $\Rightarrow\exists$ t′: T$\geq$t: T
      state($\gamma$, t′) $\models\!=$ prediction_for(B, S2, A, S1)]

Property LP5 represents the agents knowledgeability to predict the consequences of its assumptions.

### *4.2.4. LP6 observation effectiveness*

For each prediction (that regards the observable part of the world state), the agent makes the appropriate observation.

$\forall\gamma$: $\Gamma$ $\forall$t: T $\forall$A, B: INFO_ELEMENT $\forall$S1, S2: SIGN
  [state($\gamma$, t) |== prediction_for(A, S1, B, S2)
  $\Rightarrow$ [$\exists$t′: T $\exists$S3: SIGN
    state($\gamma$, t′) |== observation_result(A, S3)]

Property LP6 ensures that the agent gathers enough information to evaluate its assumptions. It is assumed that only observable predictions are made. If this assumption does not hold, additional conditions with assumed and is_relevant_for are needed in LP6.

### 4.2.5. LP7 evaluation effectiveness

Each assumption for which there is a prediction that does not match the corresponding observation result is rejected by the agent.

$\forall\gamma$: $\Gamma$ $\forall$t: T $\forall$A, B: INFO_ELEMENT $\forall$S1, S2, S3: SIGN
  [state($\gamma$, t) |== assumed(A, S1)
  $\wedge$ state($\gamma$, t) |== prediction_for(B, S2, A, S1)]
  $\wedge$ state($\gamma$, t) |== observation_result(B, S3)
  $\wedge$ S2$\neq$S3]
  $\Rightarrow$ [$\exists$t′: T$\geq$t: T
    state($\gamma$, t′) |== rejected(A, S1)]

Properties LP5, LP6, and LP7 together ensure that the agent puts enough effort into the reasoning process; it has the knowledge (in the form of predictions) and it gathers enough information to evaluate its assumptions.

### 4.2.6. LP8 rejection grounding

Each assumption that is rejected has been considered; no rejections take place without an underlying assumption.

$\forall\gamma$: $\Gamma$ $\forall$t: T $\forall$A: INFO_ELEMENT $\forall$S: SIGN
  state($\gamma$, t) |== rejected(A, S)
  $\Rightarrow$ [$\exists$t′: T<t: T state($\gamma$, t′) |== assumed(A, S)]

### 4.2.7. LP9 no assumption repetition

Only assumptions are made that have not been made earlier.

$\forall\gamma$: $\Gamma$ $\forall$t1, t2, t3: T $\forall$A: INFO_ELEMENT $\forall$S: SIGN
    state($\gamma$, t1) |== assumed(A, S) $\wedge$
    state($\gamma$, t2) |$\neq$ assumed(A, S) $\wedge$ t1$\leq$t2$\leq$t3
  $\Rightarrow$ state($\gamma$, t3) |$\neq$ assumed(A, S)

### 4.2.8. LP10 rejection effectiveness

If an assumption has been made and it does not hold in the world state then that assumption will be rejected.

$\forall\gamma$: $\Gamma$ $\forall$t: T $\forall$A: INFO_ELEMENT
  [state($\gamma$, t) |== assumed(A, S)
  $\wedge$ state($\gamma$, t) |== not holds_in_world(A, S)
  $\Rightarrow$ [$\exists$t′: T>t: T
    state($\gamma$, t′) |== rejected(A: INFO_ELEMENT, S)]

Note that the assumptions themselves usually are not observable in the world.

### 4.2.9. LP11 rejection correctness

If an assumption has been made and it does hold in the world state then that assumption will not be rejected.

$\forall\gamma$: $\Gamma$ $\forall$t: T $\forall$A: INFO_ELEMENT
  [state($\gamma$, t) |== assumed(A, S)
  $\wedge$ state($\gamma$, t) |== holds_in_world(A, S)
$\Rightarrow$ [$\forall$t′: T$\geq$t: T
  state($\gamma$, t′) |$\neq$ rejected(A: INFO_ELEMENT, S)]

### 4.2.10. LP12 assumption uniqueness

At all times, the agent only makes one assumption at a time

$\forall\gamma$: $\Gamma$ $\forall$t: T $\forall$A, B: INFO_ELEMENT $\forall$S1, S2: SIGN
  [state($\gamma$, t) |== assumed(A, S1)
  $\wedge$ state($\gamma$, t) |== assumed(B, S2)]
    $\Rightarrow$ A=B $\wedge$ S1=S2

## 5. Human reasoning traces for reasoning by assumption

It seems plausible, but it is not clear at forehand whether the dynamic properties identified above are satisfied by traces of reasoning patterns exhibited by human reasoners. To verify the psychological validity of our dynamics approach and the dynamic properties identified, protocols have been analysed of a subject (a Ph.D. student) that was asked to solve the wise persons puzzle (see Section 2) and to think aloud while doing so. The human reasoning protocols for this analysis of empirical work were taken from van Langevelde and Treur (1992). The think-aloud protocol acquired from this experiment has been coded in terms of our language and analysed to see to what extent the subject's reasoning pattern satisfies the specified dynamic properties. The subject was given the following description:

*This puzzle is about two wise men, A and B, each of which is wearing a hat. Each hat is either black or white but at least one of the hats is white. Each wise man can only observe the colour of the other wise man's hat. Both wise men are able to reason logically and they know this from each other.*

The subject was asked to solve four variants of the puzzle. For each variant he was given the colour of A's hat and whether A knows this colour. The cases of world situations were presented as indicated in Table 1, in which the left column indicates observability. For each variant of the puzzle the subject was asked to reason as if he was B and to determine the colour of B's hat given the colour of A's hat and A's knowledge about his own hat. The subject was given instructions to think aloud and the protocol was recorded on a recording device.

The transcripts are presented in the traces in the numbered lines. For the analysis of the protocols each fragment is encoded in terms of our language. This results in the encodings of the protocol fragments as is presented in the traces below in the special font.

The traces are labelled 'HTxy' where HT denotes Human Trace and xy denotes the hat colours of the other and the agent (the human in this case) itself. The numbers in the right column refer to lines in the protocol. In the traces A is presented by 'other' and B is 'self'. A statement of the form observation_result(X, pos) expresses that it has been observed that X holds. Similarly, observation_result(X, neg) expresses that it has been observed that X does not hold. Only true statements are presented in the traces. In each table in the top cell of the right column, the world observations are shown. For the observation results no lines in the protocol were available, the human reasoning started after that.

HTbw: trace of protocol fragment BW

| No. | Atom/protocol with protocol lines |
|---|---|
| 1 | observation_result(hat_colour(white, other), neg)<br>/A is wearing a black hat<br>observation_result(conclusion(dont_know_my_colour, other), pos)<br>/A does not know that he is wearing a black hat |
| 2 | assumed(hat_colour(white, self), pos)<br>/19. If A is wearing a black hat<br>20. and B sees this<br>21. then B knows that his hat has to be white<br>22. because there must be at least one white<br>23. and then that is the answer |

In HTbw the human directly makes the right assumption and then checks it against the puzzle information and the claims of the other agent. No rejections are needed, so the assumption can be kept.

HTww: trace of protocol fragment WW

| No. | Atom/protocol with protocol lines |
|---|---|
| 1 | observation_result(hat_colour(white, other), pos) |

Table 1
The four cases for the reasoning puzzle

| | BW | WB |
|---|---|---|
| Observable | not hat_colour(white, other);<br>conclusion(don't_know_my_colour, other); | hat_colour(white, other);<br>conclusion(my_hat_is_white, other); |
| Not observable | hat_colour(white, self); | not hat_colour(white, self); |
| | WW | BB |
| Observable | hat_colour(white, other);<br>not conclusion(my_hat_is_white, other);<br>conclusion(dont_know_my_colour, other); | not hat_colour(white, other);<br>not conclusion(dont_know_ my_colour, other);<br>conclusion(my_hat_is_white, other); |
| Not observable | hat_colour(white, self); | not hat_colour(white, self); |

| No. | Atom |
|---|---|
| | /A is wearing a white hat |
| | observation_result(conclusion(dont_know_my_colour, other), pos) |
| | /A does not know that he is wearing a white hat |
| 2 | assumed(hat_colour(white, self), neg) |
| | /1. A sees either a white hat or a black hat of B |
| | 2. If he sees a black hat of B |
| 3 | prediction_for(conclusion(my_hat_is_white, other), pos, hat_colour(white, self), neg) |
| | /3. then he knows that he wears a white one |
| | 4. and then he also knows what colour he wears |
| | 5. that is the white one |
| 4 | rejected(hat_colour(white, self), neg) |
| | /6. so in that case he doesn't answer 'I don't know' |
| 5 | assumed(hat_colour(white, self), pos) |
| | /7. so A must see a white hat |
| 6 | prediction_for(conclusion(dont_know_my_colour, other), pos, hat_colour(white, self), pos) |
| | /8. then he doesn't know |
| | 9. since there can be two white hats involved |
| | 10. it can be also the case that A wears a black hat |
| | 11. and B a white hat |
| | 12. so A doesn't know what hat he is wearing |
| 7 | assumed(hat_colour(white, self), pos) |
| | /13. and that means that A, as I mentioned before, must have seen a white hat |
| | 14. so B can conclude, after A's answer that he is wearing a white hat |

The human in HTww first makes the wrong assumption. When he realises that the assumption cannot hold, he makes a new assumption. The prediction that he makes in row 6 enables him to evaluate his assumption in row 7.

HTwb: trace of protocol fragment WB

| No. | Atom |
|---|---|
| 1 | observation_result(hat_colour(white, other), pos) |
| | /A is wearing a white hat |
| | observation_result(conclusion(my_hat_is_white, other), pos) |
| | /A knows that he is wearing a white hat |
| 2 | assumed(hat_colour(white, self), pos) |
| | /2. If A knows the colour of the hat he is wearing |
| | 3. then he must have seen a black one |
| | 4. because if B wears a white one |
| 3 | prediction_for(conclusion(dont_know_my_colour, other), pos, hat_colour(white, self), pos) |
| | /5. then there can be another white one involved |
| | 6. or there can be a black one involved |
| 4 | rejected(hat_colour(white, self), pos) |
| | /7. so you can exclude this possibility |
| 5 | assumed(hat_colour(white, self), neg) |
| | /8. we may assume that B is wearing a black hat |
| 6 | prediction_for(conclusion(my_hat_is_white, other), pos, hat_colour(white, self), neg) |
| | /9. and that A concludes from this 'I am wearing a white hat' |

In HTwb the human needs to revise his assumptions again, and carefully checks his assumptions.

HTbb: trace of protocol fragment BB

| No. | Atom |
|---|---|
| 1 | observation_result(hat_colour(white, other), neg) |
| | /A is wearing a black hat |
| | observation_result(conclusion(my_hat_is_black, other), pos) |
| | /A knows that he is wearing a black hat |
| | If A, A says, 'I know the colour of my hat |
| | 4. and it is black' |
| 2 | assumed(hat_colour(white, self), neg) |
| | /5. if A sees a black hat |
| 3 | prediction_for(conclusion(don't_know_my_colour, other), pos, hat_colour(white, self), neg) |
| | /6. then he doesn't know which hat he is wearing |
| 4 | prediction_for(conclusion(my_hat_is_white, other), pos, hat_colour(white, self), neg) |
| | /7. yes, he does know |
| | 8. then he is wearing the white one |
| 5 | assumed(hat_colour(white, self), pos) |
| | /9. if B is wearing a white hat |
| 6 | prediction_for(conclusion(dont_know_my_colour, other), pos, hat_colour(white, self), pos) |
| | /10. then A can wear either a white hat or a black hat |
| 7 | rejected(hat_colour(white, self), pos) |
| | /11. so, in my opinion, A can't claim he is wearing a black hat |

In HTbb after making the first assumption the human is slightly confused because the other concludes having a black hat. After error in prediction, the human makes the correct prediction for his first assumption and then rejects this first assumption. He then makes his second assumption, and makes a prediction based on that assumption. Again this prediction is refuted which leads the subject to state that A can't claim he has a black hat; he has discovered that this world state is not consistent with the rules of the puzzle; it is not an intended world state.

## 6. Simulated reasoning traces

In this section a software model for the reasoning method used in this paper is described briefly, before the traces created with that software agent are presented.

### 6.1. Simulation model

Reasoning by assumption entails reasoning with and about assumptions. Reasoning about assumptions can be considered as a form of *meta-reasoning*. The agent reasons about a set of assumptions when deciding for them to be assumed for a while (reasoning *about* assumptions). After making assumptions, the agent derives which facts are logically implied by this set of assumptions (reasoning *with* assumptions). The derived facts may be evaluated; based on this evaluation some of the assumptions may be rejected and/or a new set of assumptions may be chosen (reasoning *about* assumptions). As an example, if an assumption has been chosen, and the facts derived from this assumption contradict information obtained from a different source (e.g. by observation), then the assumption is rejected and the converse assumed.

A generic reasoning model[1] behind this pattern of reasoning has been designed in the component-based design method DESIRE for agent systems; cf. Brazier, Jonker and Treur (1998). This formally specified design has been automatically translated into a software program capable of simulating the reasoning process. The reasoning model consists of four basic (primitive) components: External World, Observation Result Prediction, Assumption Determination, and Assumption Evaluation (see Fig. 1). The component External World contains the world state and is used to execute observations. The component Observation Result Prediction reasons *with* assumptions; e.g. given the assumption

assumed(hat_colour(white, self), neg),

within this component the rule

'if not hat_colour(white, self) then
conclusion(my_hat_is_white, other)'

can be used to predict that

conclusion(my_hat_is_white, other).

The two components Assumption Determination and Assumption Evaluation reason *about* assumptions (they perform the meta-reasoning). Information is exchanged between the components where necessary. Within DESIRE, the functionality of the different components has been specified by knowledge bases in the following manner.

#### 6.1.1. Assumption determination

The component Assumption Determination performs meta-reasoning to derive which assumption to make. It uses observation results of the colour of the other agent's hat, but not of what the other agent knows about his own colour. This knowledge base expresses a form of heuristic knowledge able to generate assumptions for the different situations. It is taken into account whether or not an assumption

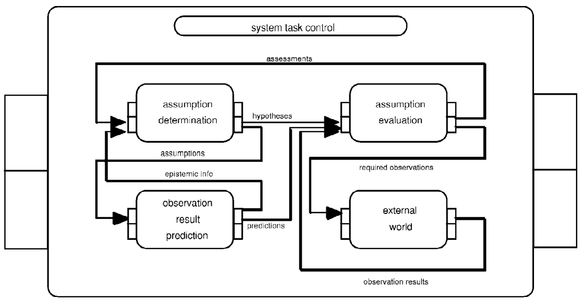

Fig. 1. Architecture of the simulation model.

[1]A complete specification of the model (with clickable components) can be found at www.cs.vu.nl/~wai/GTM/assumption/assumption_fixed_tc_2WP_07.

already has been considered before (i.e. was an assumption earlier), to avoid repetition.

```
if    observation_result(hat_colour(white, other), pos)
  and not has_been_considered(hat_colour(white, self), neg)
then  possible_assumption(hat_colour(white, self), neg);

if    observation_result(hat_colour(white, other), neg)
  and not has_been_considered(hat_colour(white, self), pos)
then  possible_assumption(hat_colour(white, self), pos);

if    observation_result(hat_colour(white, other), pos)
  and not has_been_considered(hat_colour(white, self), pos)
  and rejected(hat_colour(white, self), neg)
then  possible_assumption(hat_colour(white, self), pos);

if    not has_been_considered(hat_colour(white, self), neg)
  and rejected(hat_colour(white, self), pos)
then  possible_assumption(hat_colour(white, self), neg).
```

### *6.1.2. Observation result prediction*

The component Observation Result Prediction takes as assumption and derives from this assumption what should be expected as observations in the world. Notice that the other agent is plainly considered as part of the world. No epistemic considerations are made about the other agent; for a different, more complex model where this actually has been done, see Brazier and Treur (1999). Notice that the first rule (assuming own hat colour black) specifies that both a prediction is made about the (visible) hat colour of the other agent and about what the other agent will tell. In the other case (assuming own hat colour white) only the latter prediction is possible.

```
if    assumed(hat_colour(white, self), neg)
then  predicted_for(hat_colour(white, other), pos,
        hat_colour(white, self), neg)
  and predicted_for(conclusion(white, other), pos,
        hat_colour(white, self), neg);

if    assumed(hat_colour(white, self), pos)
then  predicted_for(conclusion(dont_know, other), pos,
        hat_colour(white, self), pos).
```

### *6.1.3. Assumption evaluation*

The component Assumption Evaluation compares predictions and observations, and, where these are conflicting, rejects the underlying assumption (see second rule below). A second functionality is to determine which observations have to be made, namely, those for which predictions exist; this is specified in the first rule.

```
if    predicted_for(OBS: INFO_ELEMENT, S1: SIGN,
        HYP: INFO_ELEMENT, S2: SIGN)
then  to_be_observed(OBS: INFO_ELEMENT);

if    assumed(HYP: INFO_ELEMENT, S: SIGN)
  and predicted_for(OBS: INFO_ELEMENT, S1: SIGN,
        HYP: INFO_ELEMENT, S: SIGN)
  and observation_result(OBS: INFO_ELEMENT, S2: SIGN)
  and S1≠S2
then  rejected(HYP: INFO_ELEMENT, S: SIGN)
  and has_been_considered(HYP: INFO_ELEMENT, S: SIGN).
```

## *6.2. Simulated traces*

For the software agent described in the previous section all world states have been tested and traces of the agent logged. The software agent reasons according to a preset strategy. It tries to opposite assumptions first. If the other wears black, the software agent will first assume that it itself wears white. If the other wears white, it will assume black for its own hat first.

STbw: trace of Seq in BW

| No. | Atom |
|---|---|
| 1 | observation_result(hat_colour(white, other), neg) |
| 2 | assumed(hat_colour(white, self), pos) |
| 3 | prediction_for(conclusion(dont_know_my_colour, other), pos, hat_colour(white, self), pos) |
| 4 | to_be_observed(conclusion(dont_know_my_colour, other)) |
| 5 | observation_result(conclusion(dont_know_my_colour, other), pos) |

In STbw, the first assumption works just fine, its evaluation is positive: no rejection is generated. Therefore no new assumption needs to be made. However, the simulation model could have been made in a manner that justification is deepened by trying the opposite assumption as well, and evaluating that this opposite assumption has to be rejected.

STwb: trace of Seq in WB

| No. | Atom |
|---|---|
| 1 | observation_result(hat_colour(white, other), pos) |
| 2 | assumed(hat_colour(white, self), neg) |
| 3 | prediction_for(hat_colour(white, other), pos, hat_colour(white, self), neg)) |
| | prediction_for(conclusion(my_hat_is_white, other), pos, hat_colour(white, self), neg)) |
| 4 | to_be_observed(hat_colour(white, other)) |
| | to_be_observed(conclusion(my_hat_is_white, other)) |
| 5 | observation_result(conclusion(my_hat_is_white, other), pos) |

In STwb, again the first assumption works just fine, it is evaluated positively: no rejection generated. Therefore no new assumption needs to be made; as in the case above, the simulation model does not look for further justification.

STww: trace of Seq in WW

| No. | Atom |
|---|---|
| 1 | observation_result(hat_colour(white, other), pos) |
| 2 | assumed(hat_colour(white, self), neg) |
| 3 | predicted(hat_colour(white, other), pos) |
| | predicted(conclusion(my_hat_is_white, other), pos) |
| 4 | to_be_observed(conclusion(my_hat_is_white, other)) |
| | to_be_observed(hat_colour(white, other)) |
| 5 | observation_result(conclusion(my_hat_is_white, other), neg) |
| 6 | rejected(hat_colour(white, self), neg) |
| 7 | assumed(hat_colour(white, self), pos) |
| 8 | predicted(conclusion(dont_know_my_colour, other), pos) |
| 9 | to_be_observed(conclusion(dont_know_my_colour, other)) |
| 10 | observation_result(conclusion(dont_know_my_colour, other), pos) |

In this case a bit more work is done. In STww, after evaluation the first assumption has to be rejected. Therefore a new assumption is made and evaluated positively: no rejection.

STbb: trace of Seq in BB

| No. | Atom |
|---|---|
| 1 | observation_result(hat_colour(white, other), neg) |
| 2 | assumed(hat_colour(white, self), pos) |
| 3 | predicted(conclusion(dont_know_my_colour, other), pos) |
| 4 | to_be_observed(conclusion(dont_know_my_colour, other)) |
| 5 | observation_result(conclusion(dont_know_my_colour, other), neg) |
| 6 | rejected(hat_colour(white, self), pos) |
| 7 | assumed(hat_colour(white, self), neg) |
| 8 | predicted(hat_colour(white, other), pos) |
| | predicted(conclusion(my_hat_is_white, other), pos) |
| 9 | to_be_observed(conclusion(my_hat_is_white, other)) |
| | to_be_observed(hat_colour(white, other)) |
| 10 | observation_result(conclusion(my_hat_is_white, other), pos) |
| 11 | rejected(hat_colour(white, self), neg) |

In STbb, the agent diligently tries both assumptions and rejects both of them. The simulation model has not been modelled to detect the impossibility of this situation and just stops reasoning when there were no more assumptions that it could make.

## 7. Validation of dynamic properties

All properties introduced in Section 4 were validated against the human and software traces of Sections 5 and 6. First, in Section 7.1 the checking program is briefly described. Next, in Section 7.2 some of the results of the validation process are discussed.

### *7.1. The checking program*

A Prolog program of about 500 lines has been developed that takes a dynamic property and a set of (empirical or simulated) traces as input, and checks whether the dynamic property holds for the traces. As an example, the specified observation result correctness is represented in this Prolog programme as a nested term structure:

```
forall(T, A, S,
implies(holds(state(C, T), observation_result(A, S), true),
holds(state(C, T), holds_in_world(A, S), true))
```

Traces are represented by sets of Prolog facts of the form

```
holds(state(m1, t(2)), a, true).
```

where m1 is the trace name, t(2) time point 2, and a is a state formula in the ontology of the component's input. It is indicated that state formula a is true in the component's input state at time point t2. The Prolog program for temporal formula checking uses Prolog rules such as

```
sat(and(F,G)):- sat(F), sat(G).
sat(not(and(F,G))):- sat(or(not(F), not(G))).
sat(or(F,G)):- sat(F).
sat(or(F,G)):- sat(G).
sat(not(or(F,G))):- sat(and(not(F), not(G))).
```

that reduce the satisfaction of the temporal formula finally to the satisfaction of atomic state formulae at certain time points, which can be read from the trace representation.

### *7.2. Outcomes of validation*

The outcome of part of this validation process is presented in Table 2.

Studying the results of the validation process gives more insight in the reasoning process of the human subject and the software simulation. Before discussing the results it might be important to know that the human was presented with all possible observations before he started his reasoning. In contrast, the software agent had to initiate all observations explicitly somewhere during its reasoning.

In general, all reasoning traces, human and simulated traces, satisfied almost all of the properties presented in Section 4. An example of an exception is in the human trace HTbb where the subject makes an error. The line of reasoning that could have followed this error according to property LP7 is not realised. Instead this line of reasoning is blocked: immediately the incorrect prediction is retracted (breaking its persistence) and replaced by the correct one. Due to this error, and its immediate correction, property GP5 and LP7 are not satisfied in this trace.

Table 2
Outcome of part of the validation

| Prop | HT wb | HT ww | HT bb | ST bw | ST wb | ST ww | ST bb |
|---|---|---|---|---|---|---|---|
| GP1 | Y | Y | Y | Y | Y | Y | Y |
| GP4 | Y | Y | – | Y | Y | Y | – |
| LP2 | Y | Y | – | Y | Y | Y | – |
| LP5 | N | N | N | Y | Y | Y | Y |
| LP6 | Y | Y | Y | Y | Y | Y | Y |
| LP7 | Y | Y | ? | Y | Y | Y | Y |

There are a few interesting differences between the human traces and the simulated traces.

- LP5: in the protocols the human reasoner only predicts the conclusions of the other party, not the hat colour of the other party. It might be that the prediction about hat colours is so obvious to the human reasoner that predicting that seems superfluous. This argument has not been checked. The software agent, if possible, also predicted a hat colour for the other agent.
- LP7: in HTbb the human does not explicitly reject the first assumption. His last conclusion suggests that he actually did reject both the first and second assumption. The software agent explicitly rejects both assumptions.

Another difference in their reaction to the impossible world situation (both A and B wear black) can be found. The human agent seems already a bit confused half way through its reasoning and makes an error. At the end he flatly concludes that A could not have said what A has said. The software agent does not get confused, but, on the other hand, was not equipped to reflect on the impossibility of the situation. It only rejected all the assumptions it could make, and then stopped reasoning.

## 8. Discussion

The dynamics of practical reasoning processes within an agent often depends on decisions about which conclusions to try to derive (the goals of the reasoning), or which premises to use (the assumptions made). An agent usually makes these types of decisions during the reasoning process. The dynamical systems theory (DST), put forward in Kelso (1995) and Port and Gelder (1995) is based on difference and differential equations, the use of which depends on the possibility to find quantitative relations over time.

For a qualitative reasoning process, this constraint makes it impossible to use these techniques. Nevertheless, it is relevant to analyse the dynamics of qualitative reasoning processes as well. This paper shows how an analysis of these dynamics can be made using techniques different from those of DST. The approach put forward makes use of traces consisting of sequences of reasoning states over time to describe reasoning processes. It is shown for the example reasoning pattern 'reasoning by assumption', how relevant dynamic properties can be identified and expressed using a temporal trace language. Example traces have been acquired in two ways. Firstly, empirical traces have been generated based on think-aloud protocols of a subject solving a reasoning puzzle.

Secondly, a simulation model has been developed and executed for a number of cases. For all these traces, the dynamic properties can and have been checked automatically, using a software environment. Thus the feasibility of the approach was shown.

Earlier work addresses the dynamics of defeasible reasoning processes based on formalisms from nonmonotonic logic; e.g. Reiter (1980) and Marek and Truszczynski (1993). In Engelfriet and Treur (1995, 1998) and Engelfriet, Marek, Treur, and Trusczczinski (2001) formalisations of the dynamics of default reasoning were contributed; for more papers in this direction see also Meyer and Treur (2001). This work fully concentrates on the internal interaction and dynamics of states during a nonmonotonic reasoning process; in contrast to the current paper, interaction with the external world is not addressed.

A pattern of reasoning similar to the pattern of reasoning by assumption occurs in the Modus Tollens case of conditional reasoning, i.e. concluding not-p from 'if p then q' and not-q. Also in that case, different alternatives are explored for p, and a falsification takes place. For a more extensive description from the viewpoint of conditional reasoning, see Schroyens et al. (2001) and Rips (1994). In these approaches the meta-level aspects are left more implicit than in our approach. It would be an interesting further step to investigate in more depth the relationships.

Future research will further address the analysis of the dynamics of other types of practical reasoning, both from the syntactical and semantical stream, or their combination; (e.g. Johnson-Laird, 1983; Johnson-Laird & Byrne, 1991; Yang & Johnson-Laird, 1999; Yang & Bringsjord, 2001; Braine and O'Brien, 1998; Rips, 1994).

Within the Artificial Intelligence literature a number of belief revision techniques have been contributed; e.g. Doyle (1979), De Kleer (1986) and Dechter and Dechter (1996), which have not been exploited to model human belief revision. Within Cognitive Science, recently an increased interest is shown in human belief revision; e.g. Byrne and Walsh (2002) and Dieussaert, Schaeken and d'Ydewalle (2002). An extension of the work reported in the current paper could address formal modelling and analysis of human belief revision in the context of reasoning by assumption in more detail.

Another area in which the formal modelling and analysis approach can be applied is human reasoning processes based on multiple representations (e.g. arithmetic, geometric). In Jonker and Treur (2002) some first steps have been made.

## References

Braine, M. D. S., & O'Brien, D. P. (Eds.), (1998). *Mental logic*. London: Lawrence Erlbaum.

Brazier, F. M. T., Jonker, C. M., & Treur, J. (1998). Principles of component-based design of intelligent agents. In Cuena, J. (Ed.), *Proceedings of the 15th IFIP World Computer Congress, WCC'98, conference on information technology and knowledge systems, IT&KNOWS'98*, pp. 347–360, Extended version in *Data and Knowledge Engineering, 41* (2002) 1–28.

Brazier, F. M. T., & Treur, J. (1999). Compositional modelling of reflective agents. *International Journal of Human–Computer Studies*, *50*, 407–431.

Busemeyer, J., & Townsend, J. T. (1993). Decision field theory: a dynamic-cognitive approach to decision making in an uncertain environment. *Psychological Review*, *100*, 432–459.

Byrne, R. M. J., & Walsh, C. R. (2002). Contradictions and counterfactuals: generating belief revision in conditional inference. In Gray, W. D., & Schunn, C. D. (Eds.), *Proceedings of the 24th annual conference of the cognitive science society, CogSci 2002*. Mahwah, NJ: Lawrence Erlbaum, pp. 160–165.

Dechter, R., & Dechter, A. (1996). Structure-driven algorithms for truth maintenance. *Artificial Intelligence*, *82*, 1–20.

Dieussaert, K., Schaeken, W., & d'Ydewalle, G. (2002). The quality of test context and contra evidence as a moderating factor in the belief revision process. In Gray, W. D., & Schunn, C. D. (Eds.), *Proceedings of the 24th annual conference of the cognitive science society, CogSci 2002*. Mahwah, NJ: Lawrence Erlbaum, pp. 280–285.

Doyle, J. (1979). A truth maintenance system. *Artificial Intelligence*, *12*, 231–272.

De Kleer, J. (1986). An assumption-based TMS. *Artificial Intelligence*, *28*, 127–162.

van Eck, P. A. T., Engelfriet, J., Fensel, D., van Harmelen, F., Venema, Y., & Willems, M. (2001). A survey of languages for specifying dynamics: a knowledge engineering perspective. *IEEE Transactions on Knowledge and Data Engineering*, *13*, 462–496.

Engelfriet, J., Marek, V. W., Treur, J., & Truszczinski, M. (2001). Default logic and specification of nonmonotonic reasoning. *Journal of Experimental and Theoretical AI*, *13*, 99–112.

Engelfriet, J., & Treur, J. (1995). Temporal theories of reasoning. *Journal of Applied Non-Classical Logics*, *5*, 239–261.

Engelfriet, J., & Treur, J. (1998). An interpretation of default logic in minimal temporal epistemic logic. *Journal of Logic, Language and Information*, *7*, 369–388.

Herlea, D. E., Jonker, C. M., Treur, J., & Wijngaards, N. J. E. (1999). Specification of behavioural requirements within compositional multi-agent system design. In Garijo, F. J., & Boman, M. (Eds.), *Multi-agent system engineering, proceedings of the 9th European workshop on modelling autonomous agents in a multi-agent world, MAAMAW'99*, *Lecture notes in AI*, vol. 1647. Springer Verlag, pp. 8–27.

Hölldobler, S., & Thielscher, M. (1990). A new deductive approach to planning. *New Generation Computing*, *8*, 225–244.

Johnson-Laird, P. N. (1983). *Mental models*. Cambridge: Cambridge University Press.

Johnson-Laird, P. N., & Byrne, R. M. J. (1991). *Deduction*. Hillsdale, NJ: Erlbaum.

Jonker, C. M., & Treur, J. (1998). Compositional verification of multi-agent systems: a formal analysis of pro-activeness and reactiveness. In de Roever, W. P., Langmaack, H., & Pnueli, A. (Eds.), *Proceedings of the international workshop on compositionality, COMPOS'97, Lecture Notes in Computer Science*, vol. 1536. Springer Verlag, pp. 350–380, Extended version in *International Journal of Cooperative Information Systems, 11* (2002) 51–92.

Jonker, C. M., & Treur, J. (2002). Analysis of the dynamics of reasoning using multiple representations. In Gray, W. D., & Schunn, C. D. (Eds.), *Proceedings of the 24th annual conference of the cognitive science society, CogSci 2002*. Mahwah, NJ: Lawrence Erlbaum, pp. 512–517.

Kelso, J. A. S. (1995). *Dynamic patterns: the self-organisation of brain and behaviour*. Cambridge, MA: MIT Press.

Kowalski, R., & Sergot, M. (1986). A logic-based calculus of events. *New Generation Computing*, *4*, 67–95.

van Langevelde, I. A., & Treur, J. (1992). Logical methods in protocol analysis. In Linster, M., & Gaines, B. (Eds.), *Proceedings of the European knowledge acquisition workshop, EKAW'91, GMD-Studien 211*, pp. 162–183.

Marek, V. W., & Truszczynski, M. (1993). *Nonmonotonic logics; context-dependent reasoning*. Berlin: Springer-Verlag.

McCarthy, J., & Hayes, P. (1969). Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, *4*, 463–502.

Meyer, J.-J. Ch. & Treur, J. (Vol. Eds.) (2001). Dynamics and management of reasoning processes. In Gabby, D. & Smets, Ph., Series Eds.), *Series in defeasible reasoning and uncertainty management systems*. Kluwer, Vol. 6.

Port, R. F., & van Gelder, T. (Eds.), (1995). *Mind as motion: explorations in the dynamics of cognition*. Cambridge, MA: MIT Press.

Reiter, R. (1980). A logic for default reasoning. *Artificial Intelligence*, *13*, 81–132.

Rips, L. J. (1994). *The psychology of proof: deductive reasoning in human thinking*. Cambridge, MA: MIT Press.

Schroyens, W. J., Schaeken, W., & d'Ydewalle, G. (2001). A meta-analytic review of conditional reasoning by model and/or rule: mental models theory revised. In *Psychological report no. 278*. University of Leuven, Laboratory of Experimental Psychology.

Yang, Y., & Johnson-Laird, P. N. (1999). A study of complex reasoning: the case GRE 'logical' problems. In Gernsbacher, M. A., & Derry, S. J. (Eds.), *Proceedings of the twenty first annual conference of the cognitive science society*, pp. 767–771.

Yang, Y., & Bringsjord, S. (2001). Mental metalogic: a new paradigm in psychology of reasoning. Extended abstract. In Chen, L., & Zhuo, Y. (Eds.), *Proceedings of the third international conference on cognitive science, ICCS 2001, Beijing*, pp. 199–204.