Behavioural Abstraction of Agent Models Addressing Mutual Interaction of Cognitive and Affective Processes

Alexei Sharpanskykh and Jan Treur

VU University Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
http://www.few.vu.nl/~{sharp,treur} {sharp, treur}@few.vu.nl

Abstract. In this paper the issue of relating a specification of the internal processes within an agent to a specification of the behaviour of the agent is addressed. A previously proposed approach for automated generation of behavioural specifications from an internal specification was limited to stratified specifications of internal processes. Therefore, it cannot be applied to mutually interacting cognitive and affective processes described by interacting loops. However, such processes are not rare in agent models addressing integration of cognitive and affective processes, agent learning and adaptation. In this paper a novel approach is proposed which addresses this issue. The proposed technique for loop abstraction is based on identifying dependencies of equilibrium states for interacting loops. The technique is illustrated by an example of an internal agent model with interdependent processes of believing, feeling, and trusting.

1 Introduction

Dynamics of an agent are usually modelled by an internal agent model specifying relations between mental states of the agent. Often such agent models are specified in an executable format following a noncyclic causal graph (e.g., [19], [23]). However, for more complex and adaptive types of agents, such agent models may have a format of dynamical systems including internal loops. In particular, when effects from cognitive on affective processes, and at the same time effects from affective on cognitive processes are taken into account, this results in agent models with internal loops. Such cyclic interactions are wellknown from the neurological and brain research areas. Examples of such cases are agents in which as-if body loops [6] are used to model the interaction between feelings and other mental states (e.g., [16]), or agents in which Hebbian learning ([12, 13]) is used to model the interaction between trust and emotional experiences (e.g., [14]). This shows that although the noncyclic graph assumption behind most existing agent models (as, for example in [19] or [23]) may be useful for the design of artificial software agents, it seriously limits applicability for modelling more realistic neurologically founded processes in natural or human-like agents.

To perform simulations with agents, for example in a multi-agent setting, it is often only the behaviour of the agents that matters, and the internal states can be kept out of the simulation model. The specification of the internal agent model in fact only acts as a format for an executable specification of the agent. Other work shows that automated transformations are possible (1) to obtain an executable internal model for a given behavioural specification (e.g., [20]), and (2) to obtain a behavioural specification from an executable internal model. The approach available for the second type of transformation (cf. [19]) has a severe limitation, as an executable internal model is assumed which has a noncyclic, stratified form. This limitation excludes the approach from being applied to agent models addressing more complex internal processes in which internal loops play a crucial role.

In the current paper a more generally applicable automated transformation is introduced from a internal agent model to a behavioural model, abstracting from the internal states.

Within this transformation, techniques for loop abstraction are applied by identifying how equilibrium states depend on inputs for these loops. It is also shown how interaction between loops is addressed. In particular for agent models, in which the interaction between cognitive and affective processes plays an important role the proposed approach is useful. Empirical work such as described in, for example, [9, 10, 11, 17, 22], reports such types of effects of emotions on beliefs. From the area of neuroscience informal theories and models have been proposed (e.g., [5, 6, 7, 22]), involving a causal relation from feeling to belief, which is in line, for example, with the Somatic Marker Hypothesis described in [2, 5], and may also be justified by a Hebbian learning principle (cf. [4, 13]).

These informal theories have been formalised in an abstracted computational form to obtain internal agent models in which also the mutual impact between affective factors and cognitive functioning is covered (e.g., [14, 16]). Such models usually use a valuation process of cognitive states by an affective loop triggered by such a state, and affecting this state. For example, in emergency situations where strong emotions of fear or panic occur, such effects are crucial to obtain accurate internal agent models.

The transformation is illustrated for two agent models that include interaction between cognitive and affective processes. Both applications address interaction between cognitive and affective processes, which has received much attention in Cognitive Science in recent years [8, 9]. A single loop case is illustrated for an existing agent model for emotion-affected beliefs, described in [16]. In addition, a novel agent model with interdependent processes of believing, feeling, and trusting is introduced in this paper illustrating a case with two interacting loops.

The paper is organised as follows. First, in Section 2 the modelling approach is briefly introduced. Section 3 presents the transformation procedure. The applications of the procedure are described in Section 4. Finally, Section 5 is a discussion.

2 Specifying Internal Agent Models

As in [19], both behavioural specifications and internal agent models are specified using the reified temporal predicate language *RTPL*, a many-sorted temporal predicate logic language that allows specification and reasoning about the dynamics of a system. To express state properties ontologies are used. *An ontology* is a signature specified by a tuple $<S_1,..., S_n,..., C, f, P, arity>$, where S_i is a sort for $i=1,.., n$, C is a finite set of constant symbols, f is a finite set of function symbols, P is a finite set of predicate symbols, $arity$ is a mapping of function or predicate symbols to a natural number. An interaction ontology InteractOnt is used to describe the (externally observable) behaviour of an agent. It is the union of input (for observations and incoming communications) and output (for actions and outgoing communications) ontologies: $InteractOnt = InputOnt \cup OutputOnt$. For example, observed(a, t) means that an agent has an observation of state property a at time point t, communicated(a1, a2, m, v, t) means that message m with confidence v is communicated from agent a1 to agent a2 at time point t, and performing_action(b) represents action b. The internal ontology InternalOnt is used to describe the agent's internal cognitive state properties (e.g., beliefs, trust states). Within the state ontology also numbers are included with the usual relations and functions.

In *RTPL* state properties as represented by formulae within the state language are used as terms (denoting objects). To this end the state language is imported in *RTPL* as follows: For every sort S from the state language the following sorts are introduced in *RTPL*: the sort S^{VARS}, which contains all variable names of sort S, the sort S^{GTERMS}, which contains names of all ground terms constructed using sort S; sorts S^{GTERMS} and S^{VARS} are subsorts of sort S^{TERMS}. Sort STATPROP contains names for all state formulae.

The set of function symbols of *RTPL* includes $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$: STATPROP x STATPROP $\rightarrow$ STATPROP; not: STATPROP $\rightarrow$ STATPROP, and $\forall$, $\exists$: S^{VARS} x STATPROP $\rightarrow$ STATPROP, of which the counterparts in the state language are Boolean propositional connectives and quantifiers. Except $\forall$,$\exists$ they are used in infix notation for better readability. To represent dynamics of a system sort TIME (a set of time points) and the ordering relation > : TIME x TIME are introduced in *RTPL*. To indicate that some state property holds at some time point the relation at: STATPROP x TIME is introduced. The terms of *RTPL* are constructed by induction in a standard way from variables, constants and function symbols typed with all before-mentioned sorts. The set of *atomic RTPL-formulae* is defined as:

(1) If t is a term of sort TIME, and p is a term of the sort STATPROP, then at(p, t) is an atomic *RTPL* formula.

(2) If τ_1, τ_2 are terms of any *RTPL* sort, then $\tau_1 = \tau_2$ is an *RTPL*-atom.

(3) If t_1, t_2 are terms of sort TIME, then $t_1 > t_2$ is an *RTPL*-atom.

The set of well-formed *RTPL* formulae is defined inductively in a standard way using Boolean connectives and quantifiers over variables of *RTPL* sorts. The language *RTPL* has the semantics of many-sorted predicate logic. More details can be found in [19].

Agent models are specified within *RTPL* in the following format: at(a, t) $\Rightarrow$ at(b, t+d) where d is the time delay of the effect of state property a on state property b, which for dynamical systems is often indicated by Δt. These state properties may involve variables, for example for real numbers. A simple example is

at(has_value(temp, V), t) $\Rightarrow$ at(has_value(temp, V-Vd), t+d)

which describes a process of cooling down to 0 degrees. This format subsumes both causal modelling languages (e.g., GARP [9]) and dynamical system modelling languages based on difference or differential equations (e.g., [17]), as well as hybrid languages combining the two, such as LEADSTO [3].

3 Abstraction of an Internal Agent Model: Eliminating Loops

In this section in a number of steps the transformation procedure is described. First the general transformation procedure as adopted from [19] is described. Next the contributed loop elimination procedure is addressed, starting by discussing the assumptions underlying this procedure and its setup, and further showing in more detail how both single loops and interaction between loops can be handled.

The general transformation procedure

The format at(a, t) $\Rightarrow$ at(b, t+d) is equivalent to at(a, t-d) $\Rightarrow$ at(b, t), where t is a variable of sort TIME. When a number of such specifications are available for one atom at(b, t), by taking the disjunction of the antecedents one specification in past to present format can be obtained

$\bigvee_i$ at(a_i, t-d_i) $\Rightarrow$ at(b, t)

When in addition a form of closed world assumption is assumed, also the format $\bigvee_i$ at(a_i, t-d_i) $\Leftrightarrow$ at(b, t) is obtained, which specifies to equivalence of the state formula b at t with a past formula. This type of format, called *pp-format* is used in the abstraction procedure introduced in [19].

The rough idea behind the overall procedure is as follows. Suppose a pp-specification B $\Leftrightarrow$ at(p, t) is available. Moreover, suppose that in B only two atoms of the form at(p1, t1) and at(p2, t2) occur, whereas as part of the agent model also specifications B1 $\Leftrightarrow$ at(p1, t1) and B2 $\Leftrightarrow$ at(p2, t2) are available. Then, within B the atoms can be replaced (by substitution) by the formula B1 and B2. This results in B[B1/at(p1, t1), B2/at(p2, t2)] $\Leftrightarrow$ at(p, t) which again is a pp-

specification. Here for any formula C the expression C[x/y] denotes the formula C transformed by substituting x for y. Such a substitution corresponds to an abstraction step. For the general case the procedure includes a sequence of abstraction steps; the last step produces a behavioural specification that corresponds to the given agent model.

Assumptions underlying the loop elimination approach

As indicated in [19] this abstraction transformation can be effective, however has a severe limitation that no loops in the given agent model specification are allowed. This limitation is addressed in the current paper.

The method for loop elimination introduced here is based on the following assumptions:

1. Internal dynamics develop an order of magnitude *faster* than the dynamics of the world external to the agent.
2. Loops are *internal* in the sense that they do not involve the agent's output states.
3. Different loops have *limited mutual interaction*; in particular, loops may contain internal loops; loops may interact in couples; interacting couples of loops may interact with each other by forming noncyclic interaction chains.
4. For static input information any internal loop reaches an *equilibrium state* for this input information.
5. It can be *specified* how the value for this equilibrium state of a given loop depends on the input values for the loop.
6. Within the agent model the loop can be *replaced* by the equilibrium specification of 4.

The idea is that when these assumptions are fulfilled, for each received input, before new input information arrives, the agent computes its internal equilibrium states, and based on that determines its behaviour. For example, these assumptions are fulfilled for agent models using repeated evaluation loops over options to determine decisions, or for agents integrating affective and cognitive processes by (recursive) as-if body loops considered in Section 4.1.

Loop elimination setup

To address the loop elimination process, the following representation of a loop is assumed

$$at(has\_value(u, V_1) \wedge has\_value(p, V_2), t) \Rightarrow at(has\_value(p, V_2 + f(V_1, V_2)d), t+d) \quad (1)$$

Here u is the name of an input variable, p of the loop variable, t is a variable of sort TIME, and $f(V_1, V_2)$ is a function combining the input value with the current value for p.

Property (1) can be compared to a recursive Prolog rule. Since the variable of the consequent of this property is used also in its antecedent, the property is necessarily being executed infinitely many times.

Note that an equilibrium state for a given input value V_1 in (1) is a value V_2 for p such that $f(V_1, V_2) = 0$. A specification of how V_2 depends on V_1 is a function g such that $f(V_1, g(V_1)) = 0$. Note that the latter expression is an implicit function definition, and under mild conditions (e.g., $\partial f(V_1, V_2)/\partial V_2 \neq 0$, or strict monotonicity of the function $V_2 \rightarrow f(V_1, V_2)$) the Implicit Function Theorem within calculus guarantees the existence (mathematically) of such a function g. However, knowing such an existence in the mathematical sense is not sufficient to obtain a procedure to calculate the value of g for any given input value V_1. When such a specification of g is obtained, the loop representation shown above can be transformed into:

$$at(has\_value(u, V1) \Rightarrow at(has\_value(p, g(V1)), t+D),$$

where D is chosen as a timing parameter for the process of approximating the equilibrium value up to some accuracy level.

In order to obtain a procedure to compute *g* based on a given function *f*, two options are available. The first option is, for a given input V_1 by numerical approximation of the solution V_2 of the equation $f(V_1, V_2) = 0$. This method can always be applied and is not

difficult to implement using very efficient standard procedures in numerical analysis, taking only a few steps to come to high precision. The second option, elaborated further below is by symbolically solving the equation $f(V_1, V_2) = 0$ depending on V_1 in order to obtain an explicit algebraic expression for the function g. This option can be used successfully when the symbolic expression for the function f is not too complex; however, it is still possible to have it nonlinear.

In various agent models involving such loops a threshold function is used to keep the combined values within a certain interval, for example $[0, 1]$. A threshold function can be defined, for example, in three ways:

(1) as a piecewise constant step-function, jumping from 0 to 1 at some threshold value

(2) by a logistic function with format $1/(1+exp(-\sigma(V_1+ V_2-\tau))$, or

(3) by a function $\beta(1-(1- V_1)(1- V_2)) + (1-\beta) V_1 V_2$.

The first option provides a discontinuous function, which is not desirable for analysis. The third format is used here, since it provides a continuous function, can be used for explicit symbolic (algebraic) manipulation, and is effective as a way of keeping the values between bounds. In this case:

$$f(V_1, V_2) = \beta (1-(1- V_1)(1- V_2)) + (1-\beta)V_1 V_2 - V_2$$

Note that $f(V_1, V_2)$ can be written as a linear function of $\mathbf{V_2}$ with coefficients in V_1 as follows:

$$\begin{aligned}
&f(V_1, V_2) = \beta(1-(1- V_1)(1- \mathbf{V_2})) + (1-\beta) V_1 \mathbf{V_2} - \mathbf{V_2} = \\
&\beta(V_1 + (1 - V_1) \mathbf{V_2}) + (1-\beta) V_1 \mathbf{V_2} - \mathbf{V_2} = \\
&\beta V_1 + \beta (1 - V_1) \mathbf{V_2} + ((1-\beta) V_1 - 1) \mathbf{V_2} = \\
&[\beta (1 - V_1) + (1-\beta) V_1 - 1] \mathbf{V_2} + \beta V_1 = \\
&- [1 - \beta + \beta V_1 - V_1 + \beta V_1] \mathbf{V_2} + \beta V_1 = \\
&- [(1- \beta)(1- V_1) + \beta V_1] \mathbf{V_2} + \beta V_1
\end{aligned}$$

From this form it follows that

$$\partial f(V_1, \mathbf{V_2}) / \partial \mathbf{V_2} = \partial -[[(1- \beta)(1- V_1) + \beta V_1] \mathbf{V_2} + \beta V_1] / \partial \mathbf{V_2} = - [(1- \beta)(1- V_1) + \beta V_1] \leq 0$$

This is only 0 for extreme cases: $\beta = 0$ and $V_1 = 1$ or $\beta = 1$ and $V_1 = 0$. So, for the general case $V_2 \rightarrow f(V_1, V_2)$ is strictly monotonically decreasing, which shows that it fulfills the conditions of the Implicit Function Theorem, thus guaranteeing the existence of a function g as desired.

Obtaining the equilibrium specification: single loop case

Using the above expression, the equation $f(V_1, V_2) = 0$ can be easily solved symbolically:

$$\begin{aligned}
&f(V_1, V_2) = - [(1- \beta)(1- V_1) + \beta V_1] \mathbf{V_2} + \beta V_1 = 0 \\
&\mathbf{V_2} = \beta V_1 / [(1- \beta)(1- V_1) + \beta V_1]
\end{aligned}$$

This provides an explicit symbolic definition of the function g:

$$g(V_1) = \mathbf{V_2} = \beta V_1 / [(1- \beta)(1- V_1) + \beta V_1]$$

For each β with $0<\beta<1$ this g is a strictly monotonically increasing function with $g(0) = 0$ and $g(1) = 1$. A few cases for specific values of the parameter β are as follows:

$\beta = 0$	$g(V_1) = 0$	constant
$\beta = 0.5$	$g(V_1) = V_1$	
$\beta = 1$	$g(V_1) = 1$	constant

Obtaining the equilibrium specification: interacting loops case

Interaction between two loops occurs when the outcome of one loop is used as (part of) input in another loop; it may occur in two forms: monodirectional or bidirectional. In the monodirectional case the previously described method can be used in a straightforward manner one-by-one for each of the loops, first for the loop providing input for the other loop.

The bidirectional case requires more elaboration. First it is assumed that the input from the other loop is combined with the externally provided input as follows: $v_1 = \lambda_1(u_1)p_2 + \mu_1(u_1)$ and $v_2 = \lambda_2(u_2)p_1 + \mu_2(u_2)$ where u_i denotes the external input (what was indicated above by V_2) for a loop i , p_i the state of the loop (what was indicated above by V_2), and λ_i and μ_i are functions of the external input u_i. Special cases are:

(1) $\lambda_1(u_1) = w_1$ and $\mu_1(u_1) = w_2 u_1$, in which case they are combined according to a weighted sum,
(2) $\lambda_1(u_1) = u_1$ and $\mu_1(u_1) = 0$, in which case p_2 acts as a modifier of the external input u_1, for example an estimated degree of reliability of the incoming information, or
(3) $\lambda_1(u_1) = - [(1- \beta)(1-u_1) + \beta u_1]$ and $\mu_1(u_1) = \beta u_1$ which provides the combination function used in *f(V1, V2)* above.

To solve the two coupled equations for this case a simplified notation is used: $v_1 = \lambda_1 p_2 + \mu_1$ and $v_2 = \lambda_2 p_1 + \mu_2$.

$$[(1- \beta_1)(1-(\lambda_1 \mathbf{p_2} + \mu_1)) + \beta_1(\lambda_1 \mathbf{p_2} + \mu_1)]\ \mathbf{p_1} = \beta_1(\lambda_1 \mathbf{p_2} + \mu_1)$$
$$[(1- \beta_2)(1-(\lambda_2 \mathbf{p_1} + \mu_2)) + \beta_2(\lambda_2 \mathbf{p_1} + \mu_2)]\ \mathbf{p_2} = \beta_2(\lambda_2 \mathbf{p_1} + \mu_2)$$

The first equation can be rewritten as follows:

$$[-(1- \beta_1)\lambda_1 \mathbf{p_2} + (1- \beta_1)(1- \mu_1) + \beta_1\lambda_1 \mathbf{p_2} + \beta_1\mu_1]\ \mathbf{p_1} = \beta_1(\lambda_1 \mathbf{p_2} + \mu_1)$$
$$[(2\beta_1-1)\lambda_1 \mathbf{p_2} + (1- \beta_1)(1- \mu_1) + \beta_1\mu_1]\ \mathbf{p_1} = \beta_1(\lambda_1 \mathbf{p_2} + \mu_1)$$
$$(2\beta_1-1)\lambda_1\, \mathbf{p_1 p_2} + [(1- \beta_1)(1- \mu_1) + \beta_1\mu_1]\ \mathbf{p_1} = \beta_1(\lambda_1 \mathbf{p_2} + \mu_1)$$

Similarly the second equation becomes:

$$(2\beta_2-1)\lambda_2\, \mathbf{p_1 p_2} + [(1- \beta_2)(1- \mu_2) + \beta_2\mu_2]\ \mathbf{p_2} = \beta_2(\lambda_2 \mathbf{p_1} + \mu_2)$$

Multiplying the first equation by $(2\beta_2-1)\lambda_2$ and the second by $(2\beta_1-1)\lambda_1$ and subtracting them from each other provides one equation

$$(2\beta_1-1)\lambda_1 [(1- \beta_2)(1- \mu_2) + \beta_2\mu_2]\ \mathbf{p_2} - (2\beta_2-1)\lambda_2 [(1- \beta_1)(1- \mu_1) + \beta_1\mu_1]\ \mathbf{p_1} = (2\beta_1-1)\lambda_1 \beta_2(\lambda_2 \mathbf{p_1} + \mu_2) - (2\beta_2-1)\lambda_2\, \beta_1(\lambda_1 \mathbf{p_2} + \mu_1)$$

This can be rewritten into a form that provides an explicit expression of $\mathbf{p_2}$ in terms of $\mathbf{p_1}$:

$$\mathbf{p_2} = [(2\beta_2-1)\lambda_2 [(1- \beta_1)(1- \mu_1) + \beta_1\mu_1 + (2\beta_1-1)\lambda_1\, \beta_2\lambda_2]\ \mathbf{p_1} + (2\beta_1-1)\lambda_1\, \beta_2\, \mu_2 - (2\beta_2-1)\lambda_2\, \beta_1\mu_1)] / [(2\beta_1-1)\lambda_1 [(1- \beta_2)(1- \mu_2) + \beta_2\mu_2 + (2\beta_2-1)\lambda_2\, \beta_1\lambda_1]]$$

Filling the expression for $\mathbf{p_2}$ in the original second equation provides one equation in $\mathbf{p_1}$:

$$[(1- \beta_2)(1-(\lambda_2 \mathbf{p_1} + \mu_2)) + \beta_2(\lambda_2 \mathbf{p_1} + \mu_2)]$$
$$[(2\beta_2-1)\lambda_2 [(1- \beta_1)(1- \mu_1) + \beta_1\mu_1 + (2\beta_1-1)\lambda_1\, \beta_2\lambda_2]\ \mathbf{p_1} +$$
$$(2\beta_1-1)\lambda_1\, \beta_2\, \mu_2 - (2\beta_2-1)\lambda_2\, \beta_1\mu_1)] / [(2\beta_1-1)\lambda_1 [(1- \beta_2)(1- \mu_2) + \beta_2\mu_2 + (2\beta_2-1)\lambda_2$$
$$\beta_1\lambda_1]] = \beta_2(\lambda_2 \mathbf{p_1} + \mu_2)$$

This provides a quadratic equation in $\mathbf{p_1}$ with as coefficients functions of $\beta_1, \beta_2, \lambda_1, \lambda_2, \mu_1, \mu_2$. By solving this equation an explicit symbolic expression is obtained for $\mathbf{p_1}$, and also for $\mathbf{p_2}$.

4 Feeling, Trusting and Believing

In this section two applications of the proposed procedure are described. First, the single loop case is illustrated for an agent model involving emotion-affected beliefs. Then, a novel agent model with interdependent processes of believing, feeling, and trusting is presented illustrating a case with two interacting loops. The models described in this section have been developed by exploiting patterns described in neurological theories in an abstracted form at the cognitive level. In such a way findings and principles from both Cognitive Science and Neuroscience become available for modelling.

4.1 A single loop case for emotion-affected beliefs

Beliefs of an agent are time-labelled internal representations created based on communication and observation results received by the agent. In [16] beliefs are specified using the function belief(p:STATPROP, v:VALUE), here p is the content of the belief and v is the degree of confidence of the agent from the interval [0, 1] that the belief content is true: v=1 indicates the agent's complete assurance of the belief; v=0 indicates the agent's complete lack of confidence in the belief content. Note that if both confidence values for some property representing the agent's belief content and for its negation equal 0, then the agent has the maximal lack of knowledge about the belief content.

According to the literature [8, 9], beliefs are only rarely emotionally unbiased. Previously, a model for emotion-affected beliefs was proposed in [16] (see also Fig. 1a) based on a *body loop* for a cognitive state described by Damasio [6, 7]:

input → cognitive state → preparation for the induced bodily response → induced bodily response → sensing the bodily response → sensory representation of the bodily response → induced feeling the emotion

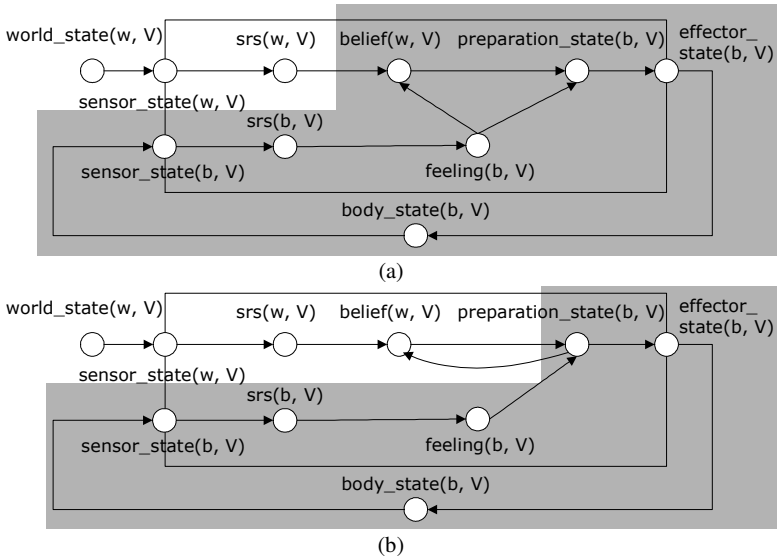

Figure 1. (a) The model for emotion-based beliefs taken from [16]; (b) An isomorphic model for the model in (a). Arrows represent causal relations and circles represent states; the grey area in both (a) and (b) contains the states involved in the body loop.

As a variation, an *as-if body loop* uses a direct causal relation preparation for the induced bodily response → sensory representation of the induced bodily response as a shortcut in the causal chain. The body loop and as-if body loop are extended to a recursive body loop or as-if

body loop by assuming that the preparation of the bodily response is (also) affected by the state of feeling the emotion. An as-if body loop for a cognitive state w is formalized in *RTPL* as follows:

at(input(w, V1) ∧ feeling(b, V2) ∧ cog_state(w, V3), t-Δt) ⇒
at(cog_state (w, V3 + γ(g(β1, V1,V2) - V3)Δt), t)

at(cog_state(w, V) ∧ body_state_for(b, w), t-Δt) ⇒ at(preparation_state(b, V), t)

at(preparation_state(B, V), t-Δt) ⇒ at(srs(B, V), t)
at(srs(B, V), t-Δt) ⇒ at(feeling(B, V), t)

Here g(β1, V1,V2) is a threshold function and γ determines the speed of change.

From the neurological perspective of Hebbian learning the existence of a connection from feeling to belief may be considered plausible, as neurons involved in the belief and in the associated feeling are often activated simultaneously [6, 22].

The model from [16] contains a composite body loop, which comprises two simple loops. To eliminate the composite loop using the mechanisms from Section 3, first an isomorphic model has been identified as shown in Fig. 1b. In this model a redefined simple body loop has a reciprocal relation with the belief state. Using the procedure from Section 3, two coupled equations are obtained for this model:

$$\boldsymbol{p_1} = \beta_1(1 - (1-\boldsymbol{p_2})(1-V)) + (1-\beta_1)\,\boldsymbol{p_2}V$$

$$[(1-\beta_2)(1-\boldsymbol{p_1}) + \beta_2\,\boldsymbol{p_1}]\,\boldsymbol{p_2} = \beta_2\,\boldsymbol{p_1}$$

Here $\boldsymbol{p_1}$ represents the confidence of the belief state, $\boldsymbol{p_2}$ is the variable for the preparation state and V is the input provided by the world. From this system a quadratic equation in $\boldsymbol{p_1}$ is obtained:

$$(1-2\beta_2)\,\boldsymbol{p_1}^2 + (\beta_2(1+\beta_1+V) + \beta_1 V - 1)\,\boldsymbol{p_1} + \beta_1 V(1-\beta_2) = 0$$

The solution to this equation agrees with the simulation results for particular values of parameters and the input reported in [16]. For example, for the case $\beta_1=0.8$, $\beta_2=0.4$, $V=0.8$, it is calculated that $\boldsymbol{p_1}=0.9255$ and $\boldsymbol{p_2}=0.8923$, which is also the case in the simulation.

4.2 Interacting loops for belief, feeling and trust

Previously, several models combining beliefs and trust were proposed [1, 21]. The authors are not aware of any computational model that combines cognitive processes of believing and trusting with affective processes (feelings and emotions). In the following a first attempt for a model for believing, feeling, and trusting is described. In this model two types of beliefs are distinguished: a factual belief of an agent that some information was observed in the environment or communicated by some source, and a belief representing the agent's own valuation of some property.

An agent creates beliefs not only about world states, but also about the world dynamics, specified by the function dyn_prop(o, f), where f is the name of a dynamic property describing the dynamics of the world object o which may be composite. In the absence of recent experience the agent may reason about the present world state using such beliefs and old experience stored in factual beliefs. To enable such reasoning, the auxiliary predicate belief_project(s:AGENT, ag:AGENT, w:STATPROP, V:VALUE) is introduced, which specifies a temporal projection of agent ag of the most recent factual belief about w based on the information received from source s. Here V is the confidence value obtained by projection; it is updated at each time point based on the agent's beliefs about the world dynamics with the highest confidence:

at(belief(communicated(s, ag, w, v, t1), q) ∧ belief(dyn_prop(w, f), v2) ∧ name_for(f, expr(x, y, z)) ∧
∀f1:STATPROP [f1≠f ∧ belief(dyn_prop(w, f1), v3, t1) → v3 < v2], t-Δt)
⇒ at(belief_project(s, ag, w, expr[x/v, y/t1, z/t]), t)

Here expr [x/v] denotes the substitution of x by v in expr(x, y, z).

It is assumed that the emotional influence on the factual beliefs is insignificant, belief projections are influenced by emotions indirectly through beliefs about the world dynamics, and all beliefs of the second type are influenced by emotions directly via an as-if body loop (see Figure 2). A belief prospect is provided to this loop as input mediated by the agent's trust in the information source of the belief prospect. In the model *trust* is an (cognitive and affective) attitude of an agent towards an information source that determines the extent to which information received by the agent from the source influences agent's beliefs. It is often argued that trust should be distinguished per information type [8]. In the model trust in a source w.r.t. an information type is represented by the preparation state to accept information of this type from the source. This preparation state accumulates all experience with the source. The amount of trust is a number from the range [0, 1]. Formally, the trust-mediated input to the as-if body loop for a belief about w is specified by:

$v = \eta pu$

Here η is the strength of the communication through the channel from the source ($\eta = 1$ if the source provided information about w, $\eta = 0$ if no information about w was received from the source); for an agent's observations the source is the environment, u is the confidence value for the belief prospect for w based on the information received from the source, p is the amount of trust of the agent to the source.

According to the formula, the higher the agent's trust in a source, the greater the source's influence on the input value. A high confidence value provided by a trustworthy source brings the input value further away from the minimal knowledge state ($v_1 = 0$).

In the case when more than one source provides information of a type w to the agent, the overall confidence value of the agent's belief representing its valuation of w is calculated by aggregating the agent's emotional beliefs about w for each source:

$b = \sum_{i=1..n} \eta_i b_i / \sum_{i=1..n} \eta_i$, where $\sum_{i=1..n} \eta_i > 0$.

Here n is the number of information sources, b_i is the confidence value of the emotional belief created based on information from the ith source, η_i is the strength of the communication channel from the ith source.

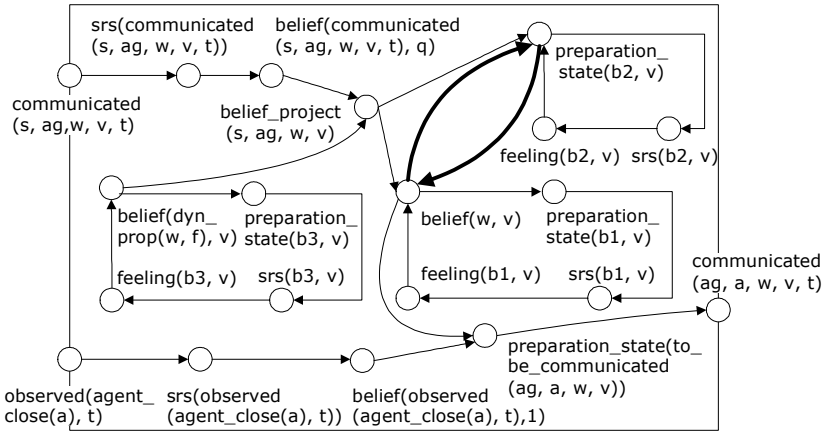

Figure 2. A schematic representation of the model for believing, feeling, and trusting for an information source s; the bold arrows represent interaction between two loops.

The trust in a source builds up over time based on the agent's experience with the source: when the agent has a positive (negative) experience with the source, the agent's trust in the source increases (decreases). The experience is evaluated as positive (or negative) when the information provided by the source is confirmed by (or disagrees with) the agent's beliefs, i.e., $\alpha|\ v1 - v2\ | < 0.5$ (or $\alpha|\ v1 - v2\ | > 0.5$), where v1 is the confidence value of the belief prospect based on the information about w received from the source, and v2 is the confidence value of the agent's belief about w; parameter α is an individual characteristic that determines the agent's tolerance for the difference between v1 and v2.

Thus, trust and beliefs are interdependent: on the one hand, the trust in a source builds up based on information received from the source evaluated using the agent's beliefs; on the other hand, the trust in a source determines the degree of influence of information from the source on the agent's beliefs. Furthermore, both trust and beliefs are influenced by emotions. Similarly to beliefs, the emotional influence on trust is modelled by an as-if body loop. The input for this loop is provided by the evaluation of experiences with the source.

The parameters of the model allow specifying diverse individual characteristics, similar to the Big Five traits: γ's in as-if body loops reflect the agent's flexibility to adopt new experiences; α reflects the agent's openness, as reported in [8], positive emotions, such as happiness, increase the agent's openness, whereas negative emotions, such as anger, have the opposite effect.

Based on the valuation of beliefs the agent decides how to act. In the model shown in Figure 2, if the agent has a high confidence (> 0.8) in a property, and observes that another agent is close, then it communicates this property to that agent. Formally:

$at(belief(observed(agent\_close(a), t-\Delta t), 1) \wedge belief(w, v) \wedge v > 0.8, t-\Delta t) \Rightarrow$
$at(preparation\_state(to\_be\_communicated(ag, a, w, v)), t)$

$at(preparation\_state(to\_be\_communicated(ag, a, w, v)), t-\Delta t)$
$\Rightarrow at(communicated(ag, a, p, w, t)), t)$

In the following it is demonstrated how the procedure from Section 3 is applied to eliminate the loops from the model from Figure 2. The loop for the belief about the world dynamics is eliminated as shown in Section 4.1. To eliminate two interacting loops from Figure 2, following the procedure, two coupled equations are obtained:

$$[(1-\beta_1)(1-\boldsymbol{p_2}u)+ \beta_1 \boldsymbol{p_2}u]\, \boldsymbol{p_1} = \beta_1 \boldsymbol{p_2} u$$

$$[(1-\beta_2)(1- \alpha|\, u - \boldsymbol{p_1}|) + \beta_2\, \alpha|\, u - \boldsymbol{p_1}|]\, \boldsymbol{p_2} = \beta_2\, \alpha|\, u - \boldsymbol{p_1}|$$

Here $\boldsymbol{p_1}$ represents the confidence of the agent's belief about w, $\boldsymbol{p_2}$ is the degree of agent's trust in the source for w; u is the confidence value for the belief prospect based on the information about w provided from the source (i.e., experience). The parameters β_1 and β_2 account for temporal discounting of old experiences in calculation of confidence values of beliefs and trust values. Furthermore, β_1 and β_2 reflect the agent's positive versus negative bias.

From this system for the case $u \geq p1$ a quadratic equation in $\boldsymbol{p_1}$ is obtained:

$$(h_2h_3 - (\beta_1-h_3)\beta_2\, \alpha)\, \boldsymbol{p_1}^2 + (h_1h_3 + (\beta_1-h_3)\beta_2\alpha u + \beta_1\beta_2\alpha u)\boldsymbol{p_1} - \alpha\beta_1\beta_2u^2 = 0, \qquad (2)$$

where

$h_1 = (1-\beta_2)(1-\alpha u)$, $h_2 = \alpha(1-\beta_2)$, $h_3 = 1-\beta_1$.

The case $u < \boldsymbol{p_1}$ is treated similarly. In cases with more than one source, each couple of loops for each source is eliminated as described above, and the obtained expressions for emotion-affected beliefs is used to calculate the overall confidence value of the agent's belief by aggregation.

Now, after all loops have been eliminated from the model, an executable behavioural specification containing a direct relation between the input and output states of the model can be automatically generated using the procedure from [19]:

$$at(observed(agent\_close(a), t\text{-}Dt) \wedge communicated(s, ag, w, v, t1) \wedge t\text{-}Dt \geq t1 \wedge f(expr(v, t1, t) > 0.8, t\text{-}Dt) \Rightarrow at(communicated(ag, a, w, v, t), t)$$

Here $f(expr(v, t1, t))$ is the solution to the equation (2) with $u=expr(v, t1, t)$ and $expr$ is the function used to calculate the belief projection; $Dt \gg \Delta t$ since the cognitive dynamics develop much faster than the externally observable dynamics.

5 Discussion

Existing models for an agent's internal functioning often have been designed from an artificial (software) agent perspective, without taking into account underlying neurological principles. In particular, they usually are based on a noncyclic causal graph assumption for the mental states involved. From the literature in the neurological and brain research area it is known that realistic processes often have a highly cyclic character. For example, affective processes may be triggered by cognitive processes, but in turn affect the very same cognitive processes. To obtain more realistic and neurologically founded agent models such mutual interactions cannot be ignored. To obtain such agent models, as for example argued in [18], techniques from the dynamical (complex) systems area in principle are a useful option, as opposed to the logical methods usually advocated (e.g., [23]). In general, the complexity of such dynamical systems may provide some difficulties, for example, for simulation and analysis of models with larger numbers of agents. However, for a substantial class of applications of such models their complexity can be analysed by identifying a number of loops that during processing lead to equilibria, and transforming the model into one in which these loops are replaced by the equilibria they reach. Previously, hybrid modelling techniques have been developed that combine aspects of dynamical systems and logical modelling (cf [15]). However, the representational and computational complexities of such techniques are high.

This paper contributes such a transformation procedure to relate a specification of an agent's internal processes to its behavioural specification, in particular for more complex and neurologically founded agent models. Thus, the scope of application of an existing transformation is substantially extended. In particular, due to this contribution also agent models have become within reach with internal processing and adaptation involved in valuation of cognitive states based on the emotional responses they trigger. As such processes theoretically involve infinitely often processed internal loops, such agent models inherently suffer from a lack of formal analysis possibilities. Elimination of loops is a complex problem, which cannot be addressed by a minor modification of existing approaches. A qualitatively new procedure is required, such as the procedure for loop elimination proposed in this paper based on identifying dependencies for equilibrium states for loops. It has been shown in the paper that when an approximation perspective is adopted loops can be eliminated by replacing them by direct functional association specifications that only require limited time for their processing. The developed procedure is novel; it interacts with the existing work (e.g., the approach from [19]) only at the level of interfaces, i.e., input and output variables. Noncyclic specifications obtained using the proposed procedure can be handled by more common analysis methods. By loop elimination the resulting agent models also become suitable for other analysis methods, for example model checking. The application of the developed procedure has been demonstrated for two neurologically founded agent models that address interaction between cognitive and affective processes.

References

[1] K. S. Barber and J. Kim. Belief Revision Process Based on Trust: Agents Evaluating Reputation of Information Sources, Trust in Cyber-societies, LNAI 2246, pp. 73-82, Springer (2001)

[2] A. Bechara and A. Damasio. The Somatic Marker Hypothesis: a neural theory of economic decision. Games and Economic Behavior, vol. 52, pp. 336-372 (2004)

[3] T. Bosse, C.M. Jonker, L. van der Meij and J. Treur. A Language and Environment for Analysis of Dynamics by Simulation. Int. J. of AI Tools, vol. 16, 435-464 (2007)

[4] G.Q. Bi and M.M. Poo. Synaptic Modifications by Correlated Activity: Hebb's Postulate Revisited. Ann Rev Neurosci, vol. 24, pp. 139-166. (2001)

[5] A. Damasio. Descartes' Error: Emotion, Reason and the Human Brain, Papermac, London (1994)

[6] A. Damasio. The Feeling of What Happens. Body and Emotion in the Making of Consciousness. New York: Harcourt Brace (1999)

[7] A. Damasio. Looking for Spinoza. Vintage books, London (2004)

[8] J.R. Dunn and M.E. Schweitzer. Feeling and Believing: The Influence of Emotion on Trust. Journal of Personality and Social Psychology. Vol 88(5), pp. 736-748 (2005)

[9] E. Eich, J.F. Kihlstrom, G.H. Bower, J.P. Forgas, and P.M. Niedenthal. Cognition and Emotion. New York: Oxford University Press (2000)

[10] J.P. Forgas, S.M. Laham and P.T. Vargas. Mood effects on eyewitness memory: Affective influences on susceptibility to misinformation. Journal of Experimental Social Psychology, vol. 41, pp. 574–588 (2005)

[11] J.P. Forgas, L. Goldenberg and C. Unkelbach. Can bad weather improve your memory? An unobtrusive field study of natural mood effects on real-life memory. Journal of Experimental Social Psychology, vol. 45, pp. 254–257 (2009)

[12] W. Gerstner and W.M. Kistler. Mathematical formulations of Hebbian learning. Biol. Cybern., vol. 87, pp. 404–415 (2002)

[13] D. Hebb. The Organisation of Behavior. Wiley, New York (1949)

[14] M. Hoogendoorn, S.W. Jaffry and J. Treur. Modelling Trust Dynamics from a Neurological Perspective. In: Proceedings of the Second International Conference on Cognitive Neurodynamics, ICCN'09. Springer Verlag (2009).

[15] P. Lincoln and A. Tiwari. Symbolic systems biology: Hybrid modeling and analysis of biological networks, Hybrid Systems: Computation and Control, HSCC 2004, Lecture Notes in Computer Science 2993, 660-672.

[16] Z.A. Memon and J. Treur. Modelling the Reciprocal Interaction between Believing and Feeling from a Neurological Perspective. In: N. Zhong et al. (eds.), Proceedings of the First International Conference on Brain Informatics, BI'09. Lecture Notes in Artificial Intelligence, vol. 5819. Springer Verlag, pp. 13-24 (2009)

[17] P.M. Niedenthal. Embodying Emotion. Science, vol. 316, pp. 1002-1005 (2007)

[18] R.F. Port and T. van Gelder. (eds.) Mind as Motion: Explorations in the Dynamics of Cognition. MIT Press, Cambridge (1995)

[19] A. Sharpanskykh and J. Treur. Relating Cognitive Process Models to Behavioural Models of Agents. In: Jain, L., Gini, M., Faltings, B.B., Terano, T., Zhang, C., Cercone, N., Cao, L. (eds.), Proceedings of the 8th IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT'08. IEEE Computer Society Press, pp. 330-335 (2008)

[20] A. Sharpanskykh and J. Treur. Verifying Interlevel Relations within Multi-Agent Systems. In: Brewka, G., Coradeschi, S., Perini, A., and Traverso, P. (eds.), Proc. of the 17th European Conference on Artificial Intelligence, ECAI'06, IOS Press, pp. 290-294 (2006). Extended version in International Journal of Agent-Oriented Software Engineering, vol. 4, 2010, to appear

[21] Y. Wang and M.P. Singh. Formal Trust Model for Multiagent Systems. In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07). pp. 1551 - 1556 (2007)

[22] P. Winkielman, P.M. Niedenthal, and L.M. Oberman. Embodied Perspective on Emotion-Cognition Interactions. In: Pineda, J.A. (ed.), Mirror Neuron Systems: the Role of Mirroring Processes in Social Cognition. Springer Science, pp. 235-257 (2009)

[23] M. Wooldridge, An Introduction to Multi-Agent Systems. John Wiley and Sons Limited: Chichester, 2002