CAAF: A Cognitive Affective Agent Programming Framework

F. Kaptein, J. Broekens, K. V. Hindriks, and M. Neerincx

Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands,
F.C.A.Kaptein@tudelft.nl

Abstract. Cognitive agent programming frameworks facilitate the development of intelligent virtual agents. By adding a computational model of emotion to such a framework, one can program agents capable of using and reasoning over emotions. Computational models of emotion are generally based on cognitive appraisal theory; however, these theories introduce a large set of appraisal processes, which are not specified in enough detail for unambiguous implementation in cognitive agent programming frameworks. We present CAAF (Cognitive Affective Agent programming Framework), a framework based on the belief-desire theory of emotions (BDTE), that enables the computation of emotions for cognitive agents (i.e., making them cognitive *affective* agents). In this paper we bridge the remaining gap between BDTE and cognitive agent programming frameworks. We conclude that CAAF models consistent, domain independent emotions for cognitive agent programming.

Keywords: models of emotionally communicative behavior · theoretical foundations and formal models · dimensons of intelligence, cognition and behavior

1 Introduction

Interaction with intelligent virtual agents is facilitated by providing such agents with affective abilities. For example, affective abilities in intelligent agents have been applied to facilitate *entertainment* [17, 23], to make an agent more *likable* for the user [3], to get *empathic* reactions from the user [7], and to create the so-called *the illusion of life* [2, 18], where characters are modelled to appear more life-like.

Cognitive agents can be programmed in frameworks like, e.g., GOAL [11], Jadex [16], or Jason [4]. A cognitive agent is an autonomous agent that perceives its environment through sensors and acts upon that environment with actuators [24]. It does so based on its *beliefs*, *desires* and *intentions*. Cognitive agents have a *mental state* and a *reasoning cycle* (see Figure 1). The mental state consists of *beliefs* and *desires*. Beliefs are the agent's representation of its environment. The agent can believe it is walking down the street, or that it is raining outside. Desires are things the agent *wants* to be true. For example, the agent can want to have an umbrella. The *intention* to get an umbrella reflects the agent's commitment to achieve that desire. After sensing *percepts* from the environment, the agent updates its mental state. Based on its beliefs, desires, and intentions, the agent reasons about its next action. The environment can change by itself, in response to an action of the agent, or actions from other agents that are situated in the same environment; thus, the agent may not always be *certain* of the exact *state of affairs* in its environment.

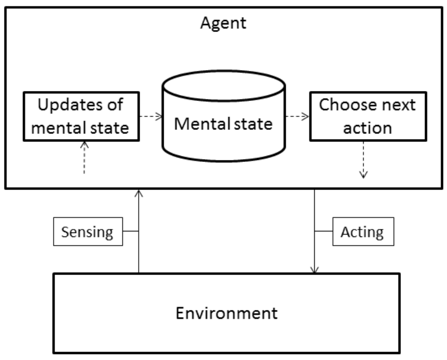

Fig. 1. The reasoning cycle of a cognitive agent.

By adding a computational model of emotion to cognitive agent programming frameworks, one can program intelligent agents capable of using and reasoning over emotions. Computational models of emotion are usually based on cognitive appraisal theories [13]. Cognitive appraisal theory proposes that emotions are consequences of cognitive evaluations (*appraisals*), relating the event to an individual's desires. For example, one is happy because one believes something to be true, and desires this to be true.

However, cognitive appraisal theories [12, 15, 25] typically introduce a large set of appraisal processes, which are not specified in enough detail for unambiguous implementation in cognitive agent programming frameworks. Psychological theories are developed to explain emotions for humans. These theories are thus not obligated to provide worked out computational specifications for the appraisals.

Here we address this problem by integrating a computational model of the belief-desire theory of emotions (BDTE) [19, 20] with a BDI (belief-desire-intention)-based, cognitive agent programming framework. We present CAAF, a Cognitive Affective Agent programming Framework. Emotions are computed based on BDTE for two reasons: 1) because it is conceptually close to the BDI agent framework; and 2) it does not introduce a large set of appraisals that are difficult to describe in a computational manner.

The two main contributions of this work are: 1) We define semantics for the programming constructs of cognitive agents, formalizing how an agent updates its *mental state*, and how emotions are computed. 2) We show when the agent should minimally (re)appraise, by proving that, under some circumstances, the computation of emotions stays consistent when reducing the frequency with which the agent's emotions are recomputed, thereby increasing the efficiency of the computation.

2 Motivation & Related Work

In this article we focus on computational models of emotion based on cognitive appraisal theory. A computational model of emotion describes the eliciting conditions for emotions, often including corresponding intensity. A popular appraisal theory among computer scientists, is the OCC-model [1, 15, 27]. The appraisal theory by Lazarus [12], and the sequential check theory (SCT) by Scherer [25, 26] have also found some attention among computer scientists. For example, the computational model EMA [10, 14] is mainly based on the appraisal theory by Lazarus [12], where the link between appraisal and coping is emphasized. EMA models how emotions develop and influence each other. For example, sadness can turn into anger at the responsible source. In [5] a formal notation for the declarative semantics of the structure of appraisal is proposed. Using this, a computational model of emotion is developed based on SCT.

The OCC model is the most implemented cognitive appraisal theory. Computational models based on the OCC model include AR [9], EM [18], FLAME[8], FearNot! [7], FAtiMA [6], and GAMYGDALA [17]. In AR [9] agents judge events based on their pleasantness, and whether they are confirmed, unconfirmed, or disconfirmed. For example, sadness is achieved when an agent confirms an unpleasant event. In EM [18] the aim is to build 'believable agents', agents that appear emotional and engage in social interactions. The EM architecture facilitates artists to model emotional agents in their applications. In FLAME the desirability of an event is modelled with fuzzy sets. For example, they define a fuzzy set 'undesirable event'. Individual events are then partly a member of this set, the amount of membership is adaptively learned over time. FearNot! is an application that helps children to cope with bullying. The agents use planning and expected utility to derive proper emotional responses. Currently the emotional responses in FearNot are triggered with a more enhanced model FAtiMA. FAtiMA divides the appraisal into different modules, all responsible for a separate part of the computation. This enables implementing such modules independently. GAMYGDALA is an emotion engine that can be added to games by annotating events with their influence on the beliefs and desires of different characters.

An underlying problem with many appraisal theories is that cognitive agent programming frameworks lack the required knowledge representations to compute most appraisal processes. For example, a computational model of emotion that aims to describe the OCC-model in total [15], including emotion intensities, needs to model 12 different appraisals. For many of these appraisals it is unclear *how* they should be implemented, e.g., *deservingness*, *sense of reality*, or *proximity*. Other appraisals, e.g., praiseworthiness, require complex constructs

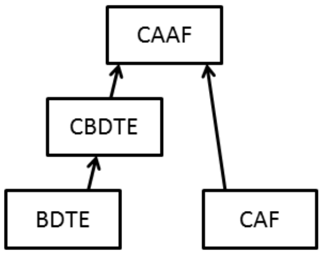

Fig. 2. CAAF is build upon CBDTE [20] and CAFs (Cognitive Agent programming Frameworks). With CAAF, we close the gap between CBDTE and CAFs, and provide a fully worked out, computational account of BDTE.

like norms and values to be represented by the agent. SCT [25, 26] additionally introduces multiple layers in the appraisal process. An event is first analysed in a reactive, bodily responsive, type of way, and later analysed with increasingly nuanced cognitive processes. The computational model of emotion, EMA [14], is mainly based on the appraisal theory by Lazarus [12]. EMA [14] aims to simplify the appraisal processes, introduced by the underlying appraisal theories, and models them from a knowledge representation consisting of beliefs, desires, intentions, and (decision-theoretic) plans. This is conceptually closer to cognitive agent programming frameworks; however, though these frameworks are suited for programming decision-theoretic plans, they do not always do so. This would thus put constraints on the agent programming frameworks for which we want to compute emotions.

The appraisals and knowledge representation proposed by the belief-desire theory of emotion (BDTE) [19, 20] are more compatible with cognitive agent programming frameworks. In BDTE, emotions are derived only from beliefs and desires. In its minimal form BDTE requires only two appraisals. This makes BDTE more suitable as a basis for simulated emotions for such frameworks.

In this paper we integrate a computational model of BDTE with a cognitive agent programming framework (CAF), hence developing CAAF. In [20], Reisenzein extended BDTE to a computational form (CBDTE). CBDTE has been referred to as a computational model of emotion [13]; however, Reisenzein acknowledges that the motivation behind developing CBDTE was not to develop a worked-out computational model, but rather to clarify aspects of BDTE [20]. Here, we build upon CBDTE, and close the gap between CAFs and CBDTE (see figure 2). Thus, this paper presents a *full* computational account of BDTE, and formalizes how a cognitive agent should (efficiently) compute emotions.

3 A Model of Emotion for Cognitive Agent Programming Frameworks

In this Section we present CAAF. We present the formal semantics needed to integrate BDTE with cognitive agent programming. Further, based on this formal system we show in Section 4 that emotions can be computed in an efficient way using the model presented here.

3.1 Semantics for a Basic Knowledge Representation & BDTE

The mental state of an agent requires a *knowledge representation.* The agent needs to *represent* states of affairs, to *store* these representations, and to *change* the stored representations.

Representing the states of affairs is achieved with a *language.* This language needs to define a syntax of *well-formed formulae.* We write $\varphi \in \mathcal{L}$ to denote that φ is a formula of language $\mathcal{L}$. Here, a formula is a single proposition that contains information about a *state of affairs*, i.e., it is a sentence that *expresses whether a state of affairs is true (or not).* We do not define how *logical connectives* work in this language, i.e., symbols that connect propositions such that the sense of the compound proposition depends only on the original sentences (for example, φ_1 *and* φ_2). The contribution of this paper is to define semantics for the programming constructs of cognitive agents, formalizing how an agent updates its *mental state*, and how emotions are computed.

Storing states of affairs is done with a *set.* The belief, desire and emotion base are represented in the semantics as a set of formulae, mapped to a value $[0,1]$. These bases are a subset of some language $\mathcal{L}$, but contain further information as well. A belief base has the form: $\Sigma : \langle C : \mathcal{L} \rightarrow [0,1] \rangle$, where C is mapping of a formula φ to (exactly one) certainty value between $[0,1]$. We denote $b\{\varphi \rightarrow c\} \in \Sigma$ for 'the agent believes φ with certainty c'. Furthermore, we add the constraint that if C contains the mappings $b\{\varphi \rightarrow c\}$ and $b\{\neg\varphi \rightarrow c'\}$, then $c = 1 - c'$. A desire base has the form $\Gamma : \langle U : \mathcal{L} \rightarrow [0,1] \rangle$, where U is mapping that maps formula φ to a utility value between $[0,1]$. We denote $d\{\varphi \rightarrow c\} \in \Gamma$ for 'the agent desires φ with utility (strength of desire) u'. Finally an emotion base has the form $\Upsilon : \langle I : \mathcal{L} \times \Theta \rightarrow [0,1] \rangle$, where $\theta \in \Theta$ is an emotion label (happy, unhappy, hope, fear, surprise, relieve, or disappointment), and I maps formula $\varphi \in \mathcal{L}$ and label $\theta \in \Theta$ to an intensity value between $[0,1]$. We denote $e\{\varphi \times \theta \rightarrow i\} \in \Upsilon$ for 'the agent has emotion θ (concerning formula φ) with intensity i'. Note that traditional boolean propositional logic (where formulae are either true or false, rather than mapped to a value between $[0,1]$) would be sufficient for programming cognitive (BDI-based) agents [11]. However, for the computation of many emotions in BDTE we need values between $[0,1]$. For example, an agent that applies for a new job cannot feel hope (according to BDTE) when it only knows if it got the job afterwards. It should reason over the certainty of getting this job. For example, after having a good job interview. Also note that the emotions in Υ contain a formula, rather than just a label and intensity. With this we model the apparent directedness of emotions, in line with BDTE [20]. One is happy *about* some formula, e.g., $\varphi =$ 'I will get a new job'.

Changing the knowledge representation is denoted with a combine operator $\oplus$. Given some set S and some set T containing a number of formulae, $S \oplus T$ denotes an update of S with T. $\oplus$ is a simple set join, with elements in set T taking priority over elements in set S, to allow updating of c, u and i in S. For all formulae $\varphi \in S$ and $\varphi \in T$, the mapping $\varphi \to n$ in the resulting set is taken from the set T. Thus, $\oplus$ is not symmetric, i.e., $S \oplus T \neq T \oplus S$.

Definition 1. (Combine $\oplus$)
Given some sets S, and T, which contain a number of elements $e = \{\varphi \to n\}$, where φ is a formula $\varphi \in \mathcal{L}$, and n a value $n \in [0, 1]$. $S \oplus T$ is defined as follows:

$$e \in S \oplus T \quad \textit{iff} \quad e \in T, \ \textit{or} \ (e \in S \ \textit{and} \ e \notin T)$$

A knowledge representation is a pair $\langle \mathcal{L}, \oplus \rangle$, where $\mathcal{L}$ is a language to represent states of affairs, and $\oplus$ defines how a set of formulae is updated with another set of formula. Using our definition of a knowledge representation, we can now formally define what a *mental state* of an agent is. We call this initial definition a 'Simple Mental State' because we will expand it later in the paper.

Definition 2. (Simple Mental State)
A mental state is a pair $\langle \Sigma, \Gamma \rangle$ where Σ is called a belief base, and Γ is a desire base.

The aim of the work presented here is to add *emotional reasoning* to these agent programming frameworks. The belief-desire theory of emotion (BDTE) [19, 20] provides a method for computing emotional responses based solely on ones beliefs and desires. For BDTE we need only the beliefs and desires, before and after an agent's update of its mental state. We could imagine that a computation of an agent program is a sequence of mental states $m_0, m_1, m_2, \ldots$. BDTE then enables the computation of an agent's emotions in a mental state m_i by using the belief- and desire base corresponding to mental states m_{i-1} and m_i. Based on BDTE we can define the inner workings of this function [20].

Definition 3 describes BDTE in a computational manner. This is based on CBDTE [20]. In function $R(\Sigma, \Sigma', \Gamma, \Gamma') \to \Upsilon$ (R for Reisenzein's appraisal [20]), we denote Σ as the belief base of mental state m_{i-1}, Γ as the desire base of mental state m_{i-1}, Σ' as the belief base in mental state m_i, and Γ' as the desire base of mental state m_i. The function $R(\Sigma, \Sigma', \Gamma, \Gamma')$ computes all new emotions resulting from changes in the mental state.

Definition 3. (BDTE R)
Given function $R(\Sigma, \Sigma', \Gamma, \Gamma') \to \Upsilon$. Let S be the set containing all φ such that $b\{\varphi \to c\} \in \Sigma$, $b\{\varphi \to c'\} \in \Sigma'$, $d\{\varphi \to u\} \in \Gamma$, and $d\{\varphi \to u'\} \in \Gamma'$, with $c \neq c'$, or $u \neq u'$. $S = \{\varphi_1, \ldots, \varphi_n\}$. If we iterate through S with $i = 1..n$, add the following emotions as follows: $\Upsilon = E_1 \oplus E_2 \oplus \ldots \oplus E_n$, such that:

$$\begin{array}{lll}
e\{\varphi_i \times \textit{happy} \to u\} \in E_i & \textit{iff} & c' = 1 \ \& \ u > 0 \\
e\{\varphi_i \times \textit{unhappy} \to u\} \in E_i & \textit{iff} & c' = 0 \ \& \ u > 0 \\
e\{\varphi_i \times \textit{hope} \to c' \times u\} \in E_i & \textit{iff} & 0 < c' < 1 \ \& \ u > 0 \\
e\{\varphi_i \times \textit{fear} \to (1 - c') \times u\} \in E_i & \textit{iff} & 0 < c' < 1 \ \& \ u > 0 \\
e\{\varphi_i \times \textit{surprise} \to 1 - c\} \in E_i & \textit{iff} & c' = 1 \\
e\{\varphi_i \times \textit{surprise} \to c\} \in E_i & \textit{iff} & c' = 0 \\
e\{\varphi_i \times \textit{relief} \to 1 - c\} \in E_i & \textit{iff} & c' = 1 \ \& \ u > 0 \\
e\{\varphi_i \times \textit{disappointment} \to c\} \in E_i & \textit{iff} & c' = 0 \ \& \ u > 0
\end{array}$$

For example, let $\varphi_1 =$ '*I got a new job*', $b\{\varphi_1 \to 1\} \in \Sigma'$ (i.e., the agent beliefs to have gotten a new job), and $d\{\varphi_1 \to 0.9\} \in \Gamma$ (i.e., the agent strongly desires to have gotten a new job), then Definition 3 prescribes $e\{\varphi_1 \times \mathit{happy} \to 0.9\} \in \Upsilon$ (i.e., the agent is very happy that it got a new job).

With these definitions we already have a framework to implement emotions, which basically works as proposed in previous work [20]. We might imagine that the computation of an agent program results in a sequence of mental states $m_0, m_1, m_2, \ldots$. Computing emotions can then be done by computing Υ over two consecutive mental states. However, this approach does not take into account that emotion intensities decay over time, how to deal with multiple appraisals of the same emotion label (θ), or the fact that you might want to store emotions for reasoning purposes. Furthermore, computation based on BDTE gives a large set containing multiple emotions for every formula φ the agent has in its mental state, meaning we need a method to abstract useful information from it.

3.2 Closing the Semantic Gap between BDTE and BDI

In this Section we expand the model such that BDTE can be used for agent programming in an efficient way, including decay, repeated appraisals, and querying the emotions. We start with expanding the mental state of an agent with an emotion base. With this we can store the current emotional state of an agent, and query this when needed.

Definition 4. (Mental State)
A mental state is a triple $\langle \Sigma, \Gamma, \Upsilon \rangle$ where Σ is called a belief base, Γ is a desire base, and Υ is an emotion base.

With an emotion base storing the emotional responses we can now define a function that gradually decays the intensities of the stored emotions. Function $d(\Upsilon, \Delta t)$ is responsible for decaying the emotional state Υ over time Δt. For the consistency of our model (see Section 4) we define Δt to be zero within one reasoning cycle of an agent. Between reasoning cycles, Δt is a function over the actual system time passed between the start of the previous and current reasoning cycle. Function decay is a mapping $d : \Upsilon \to \Upsilon'$, that decreases the intensity $i \in [0, 1]$ for all elements $e\{\varphi \times \theta \to i\} \in \Upsilon$.

Definition 5. (Decay Function d)
Let $e\{\varphi \times \theta \to i\} \in \Upsilon$. d is a function $d(\Upsilon, \Delta t) \to \Upsilon'$ defined as:

$$e\{\varphi \times \theta \to f(\theta, i, \Delta t)\} \in d(\Upsilon, \Delta t) \quad \textit{iff} \quad e\{\varphi \times \theta \to i\} \in \Upsilon$$

Where $f(\theta, i, \Delta t)$ is a function that decreases the intensity i, and for all emotions $e \in \Upsilon$ the emotion also exists in Υ' with a decayed intensity. The function can be initialized differently for every emotion label $\theta \in \Theta$. An example of exponential decay for happy would be: $f(\mathit{happy}, i, \Delta t) = i - i \times \Delta t$.

We adopt the view in [18] that decay may need different instantiations for different emotions, depending on the corresponding emotion label $\theta \in \Theta$. For example, hope and fear may decay slower than surprise. In our model an agent programmer can adjust the default decay function, for every emotion label independently.

The above defined functions come together in (i.e., are sub-functions of) function **EM**. This function is a mapping: $\mathbf{EM}(\Sigma \times \Sigma \times \Gamma \times \Gamma \times \Upsilon) \rightarrow \Upsilon$.

Definition 6. (Emotion Base Transformer **EM**)
Let Σ, Γ, and Υ be a belief base, desire base, and emotion base in some mental state m. Further, let Σ', and $dbase'$ be the belief base and desire base after some update on this mental state. Function $\boldsymbol{EM}(\Sigma \times \Sigma' \times \Gamma \times \Gamma' \times \Upsilon) \rightarrow \Upsilon'$ computes the emotion base in this updated mental state as follows:

$$\Upsilon' = d(\Upsilon, \Delta t) \oplus R(\Sigma, \Sigma', \Gamma, \Gamma')$$

This function is called when the belief base or desire base of an agent change. This happens through *updates*. There is a set of build-in updates that act on the mental state bases of the agent. Updates change the belief and desire bases of the agent. Whilst performing these updates, the agent will automatically add emotions to its emotion base Υ.

Definition 7. (Mental State Transformer $\mathcal{M}$)
Let $\varphi \in \mathcal{L}$, and $n \in [0,1]$. The mental state transformer function $\mathcal{M}(update, m) \rightarrow m'$ *is a mapping from built-in updates (update = [insert, adopt, drop]) and mental states $m = \langle \Sigma, \Gamma, \Upsilon \rangle$ to mental states as follows:*

$$\begin{array}{ll} \mathcal{M}(\boldsymbol{insert}(\varphi, n), m) & = \langle \Sigma \oplus \{\varphi \rightarrow n\}, \Gamma, \Upsilon' \rangle \\ \mathcal{M}(\boldsymbol{adopt}(\varphi, n), m) & = \langle \Sigma, \Gamma \oplus \{\varphi \rightarrow n\}, \Upsilon' \rangle \\ \mathcal{M}(\boldsymbol{drop}(\varphi), m) & = \langle \Sigma, \Gamma \oplus \{\varphi \rightarrow 0\}, \Upsilon' \rangle \end{array}$$

with $\Upsilon' = \boldsymbol{EM}(\Sigma, \Sigma', \Gamma, \Gamma', \Upsilon)$, where Σ' is the belief base, and Γ' is the desire base in the resulting mental state m'.

Mental state bases are defined as sets, thus, if a previous mapping $\{\varphi \rightarrow n\}$ exists in the mental state, then the updates defined above overwrite the previous mapping. In BDTE the claim is made that emotions are subconscious meta-representations of ones beliefs and desires [20]. In the definition above, we model this with function **EM**, which automatically updates the emotions when updating the beliefs, and desires in the mental state.

Definition 8. (Transition rule)
Let m be a mental state, and $\mathtt{u}$ be an update ([insert, adopt, drop]) performed in mental state m. The transition relation $\xrightarrow{\mathtt{u}}$ is the smallest relation induced by the following transition rule.

$$\frac{\mathcal{M}(\mathtt{u}, m) \textit{ is defined}}{m \xrightarrow{\mathtt{u}} \mathcal{M}(\mathtt{u}, m)}$$

The execution of an agent as explicated above, results in a *computation.* A computation in this context is a list of mental states and corresponding updates, performed by the agent. The new mental state is derived from the transition rule in Definition 8. The agent chooses its next update from the set of possible updates in the current state, this set is filled through the rules defined by the programmer. The computation starts in the initial mental state of the agent.

Definition 9. (Mental Computation)
A mental computation is a sequence of mental states $m_0, \mathtt{u}_0, m_1, \mathtt{u}_1, m_2, \mathtt{u}_2, \ldots$ *such that for each* i *we have that* $m_i \xrightarrow{\mathtt{u}_i} m_{i+1}$ *can be derived using the transition rule of Definition 8.*

The emotion update function **EM** is triggered as part of the Mental State Transformer (Definition 7). It is a part of the mapping from $m_i \xrightarrow{\mathtt{u}_i} m_{i+1}$. Emotions are thus computed after every mental state change of an agent.

Figure 1 showed the reasoning cycle of an agent. The mental computation, defined in Definition 9, operates solely in the 'updates of mental state' box. This means that in the model presented here, an agent senses its environment and starts updating its mental state based on these observations. With these mental state updates, we now defined how emotions are automatically changed accordingly. After updating its mental state, the agent can choose a new action to perform in the environment, which in turn changes the environment. The agent then again senses the changes in the environment, and the cycle starts anew.

3.3 Querying the Emotion Base

Querying the emotion base of an agent is useful. For example, if one wants to know if the agent is happy then one should inspect the emotion base for formulae about which the agent is happy. However, a computation based on BDTE gives a large set containing multiple emotions for every formula φ the agent has in its mental state. We therefore need a function that abstracts over these formulae.

To model this, we define an overall *affective state*, which summarizes the agent's emotions. We compute this affective state with function A. This function computes abstractions from the emotion base that enable a programmer to, for example, query the overall happiness of an agent. It summarizes the emotions in some emotion base Υ. It does so by taking all formulae in the emotion base Υ, for all emotion labels $\theta \in \Theta$, and computing a single intensity from these emotions in Υ concerning the emotion label θ.

Besides the computational argumentation there is also a psychological argumentation to define the affective state. In [21] Reisenzein argues that emotions have a hedonic tone, different than that of beliefs and desires. It *feels* a certain way to have an emotion, which is essentially different from how a belief or desire feels. In his own words: "To account for the hedonic tone of emotions in BDTE, one must assume that 'emotional' belief-desire configurations cause a separate mental state that carries the hedonic tone. [21]" By means of an affective state we model this hedonic tone of emotions.

Definition 10. (Affective State Ω)
Ω is a function, that computes a generalized affective state which summarizes the emotions $e\{\varphi \times \theta \to i\} \in \Upsilon$ for some emotion label $\theta \in \Theta$.

$$\Omega(\theta, \Upsilon) = \log_2(\textstyle\sum_{e\{\varphi\times\theta\to i\}\in\Upsilon} 2^{i\times 10})/10$$

In our model we have implemented $\Omega(\theta, \Upsilon)$ with a logarithmic function (Log_2 $(\sum 2^{i\times 10})/10$), where we sum over all emotions $e\{\varphi \times \theta \to i\} \in \Upsilon$ corresponding to label θ. Other possible functions might be normal combine: $i' = I/(I + 1)$, with I the summation of all intensities concerning θ), or a simple MAX function (taking the highest intensity emotion corresponding to θ.

From these functions the logarithmic is computationally speaking slightly less efficient; however, the function forces the resulting intensity to be as least as large as the highest value, but takes other values into account. For example, happiness about three different propositions: φ_1 = 'Getting a new job', φ_2 = 'Buying a new car', and φ_3 = 'Going out for dinner', with corresponding intensities: $[0.7, 0.6, 0.3]$, will compute to an overall happiness of 0.76 with logarithmic combine, to 0.62 with normal combine, and to 0.7 with the MAX function.

We do not claim that this is the only correct way to compute the overall affective state, but rather that an agent programmer *requires* a summary to efficiently query the emotion base, and that the here proposed approach will thus help the programmer.

4 Proof of Consistency when Minimizing the (Re)Appraisal of Emotions

In Section 3 we defined the (re)computation of an agent's emotions to occur after every mental state update. However, this is not a computationally optimal approach. In this Section we show how one can optimize this by showing when an agent should minimally (re)compute its emotions (i.e., when the agent should *(re)appraise*).

There are three conditions that should trigger a reappraisal: 1, An agent should reappraise before querying its emotion base, if it has updated its mental state since the last reappraisal, since otherwise it would query an outdated emotional state. 2, An agent should reappraise before a mental state update if the last reappraisal was in a previous reasoning cycle, otherwise the emotions are not correctly decayed. 3, An agent should reappraise when it performs a mental state update on a formula that had already been updated after the last reappraisal, otherwise the previous update will be lost. Since 1 and 2 directly follow from the formal semantics, we need only to show that 3 is true. We do so by proving that if we assume that updates refer to different formulae, appraisal can be postponed to the last update. From this one can infer point 3.

Theorem 1. *Consistency For Delayed Appraisal*
Let $\mathtt{u}_1, \mathtt{u}_2, .., \mathtt{u}_n$ be different mental state updates, with $\varphi_1, \varphi_2, \ldots, \varphi_n$ the formulae these updates refer to respectively. Furthermore, let $\mathtt{u}'_1, \mathtt{u}'_2, .., \mathtt{u}'_n$ be the same

mental state updates; however, for these mental state updates we define the Mental State Transformer (Definition 7) to delay updating the emotion base until $\mathtt{u}'_n$. *Furthermore let* $\varphi_1 \neq \varphi_2 \neq \ldots \neq \varphi_n$. *Consider the following two possible reasoning cycles:*

$$\begin{aligned} rc_1: &\quad m_0 \xrightarrow{\mathtt{u}_1} m_1 \xrightarrow{\mathtt{u}_2} \ldots \xrightarrow{\mathtt{u}_n} m_n \\ rc_2: &\quad m_0 \xrightarrow{\mathtt{u}'_1} m'_1 \xrightarrow{\mathtt{u}'_2} \ldots \xrightarrow{\mathtt{u}'_n} m'_n \end{aligned}$$

where rc_2 *delays updating the emotion base until update* $\mathtt{u}'_n$. *Under the constraint that* $\varphi_1 \neq \varphi_2 \neq \ldots \neq \varphi_n$, *we can derive that* $m_n = m'_n$.

To show the truth of this claim, let the knowledge bases corresponding to mental state m_i be denoted with, $m_i = \langle \Sigma_i, \Gamma_i, \Upsilon_i \rangle$. Since Σ and Γ are updated normally we need only to show that $\Upsilon_n = \Upsilon'_n$. To this end, we first need to define a property of the definitions. We defined Δt in function d (decay) to be zero within one reasoning cycle. Furthermore, $d(\Upsilon, 0) = \Upsilon$. Due to this, we can ignore decay when comparing reasoning cycles rc_1 and rc_2. If we denote E_i to be the set of emotions resulting from function R in transition $m_{i-1} \xrightarrow{\mathtt{u}_i} m_i$, then we can write:

$$\begin{aligned} \Upsilon_1 &= d(\Upsilon_0, 0) \oplus E_1 \\ &= \Upsilon_0 \oplus E_1 \\ \Upsilon_2 &= d(\Upsilon_0 \oplus E_1, 0) \oplus E_2 \\ &= \Upsilon_0 \oplus E_1 \oplus E_2 \\ \Upsilon_n &= \Upsilon_0 \oplus E_1 \oplus E_2 \oplus \ldots \oplus E_n. \end{aligned}$$

The emotion base resulting from reasoning cycle 2 can be found with the same definitions. Since the update of the emotion base is delayed, the emotion base $\Upsilon'_{n-1} = \Upsilon_0$. Furthermore, the computation of new emotions (Definition 3) will consider all updated formulae:

$$\begin{aligned} \Upsilon'_n &= d(\Upsilon_0, 0) \oplus \{E_1 \oplus E_2 \oplus \ldots \oplus E_n\} \\ &= \Upsilon_0 \oplus E_1 \oplus E_2 \oplus \ldots \oplus E_n. \end{aligned}$$

If $\varphi_1 \neq \varphi_2 \neq \ldots \neq \varphi_n$, then the emotions in sets $E_1, \ldots, E_n$ do not overwrite each other when added to the emotion bases. Therefore, we can conclude that $\Upsilon_n = \Upsilon'_n$. Together we can now also conclude $m_n = m'_n$.

5 Discussion

In this section we discuss some drawbacks of using BDTE as psychological background. BDTE models a limited range of emotions compared to other theories (BDTE models 7 emotions, while, for example, OCC models over 20 different emotions). Should an agent programmer want to use the emotions in the agent's decision making, then a smaller set of emotions might be more conceivable; however, there can also be domains in which the set of emotions modelled by BDTE is too limited. For example, when a programmer needs the agent to properly reason over empathic emotions like gratitude and remorse, then BDTE is inadequate in its current form.

Future work could thus complement this framework by modelling social emotions. In [22], Reisenzein discusses possible extensions of BDTE to take social emotions into account. For example, he proposes introducing *altruistic desires*. For example, pity is then explained as a form of displeasure following from the frustration of an *altruistic desire* (desiring something good for someone else). However, this does not provide explanations for all social emotions (e.g., anger). When adding social emotions, one might need to complement the presented framework with additional concepts such as norms.

6 Conclusion

In this paper we presented CAAF (a Cognitive Affective Agent programming Framework), a framework where emotions are computed automatically when agents update their mental states. We presented semantics showing the programming constructs of these agents in a domain-independent manner. With these constructs, a programmer can build an agent program with cognitive agents that automatically compute emotions during runs. We chose BDTE to compute new emotions because it is conceptually close to the BDI architecture and therefore allowed us to embed emotions without introducing many additional concepts in the mental states of the agents.

Our semantics facilitate incremental work. For example, if it is desirable to change the affective state (Definition 10) with a global mood, then one could change the function that computes the affective state (function A), without being forced to adjust the entire framework. One might also want to enable programmers to adjust the emotion base without changing the belief base. Definition 7 defined functions to update the agent's mental state. We could simply complement this definition to contain function *Appraise*, capable of inserting emotions in the emotion base ($\varUpsilon$), similar to the update *insert* for the belief base ($\varSigma$). This fits well in the modular approach suggested by Marsella et. al. [13], where models can implement parts of a complete cycle of emotional reasoning. For example, one could add a module capable of using emotions to guide the agent's decision making (e.g., what action to perform in the environment, or when to decrease the utility of a desire as a type of coping behaviour). The framework presented in this paper thus provides a modular, domain-independent, and consistent implementation for the computation of emotions for cognitive agent programming frameworks, thus facilitating the development of intelligent virtual agents with affective abilities.

Acknowledgements

This research is done for the PAL (a Personal Assistant for a healthy Lifestyle)-project. PAL is funded by Horizon2020 grant nr. 643783-RIA.

References

[1] C. Adam, A. Herzig, and D. Longin. A logical formalization of the occ theory of emotions. *Synthese*, 168(2):201–248, 2009.

[2] J. Bates et al. The role of emotion in believable agents. *Communications of the ACM*, 37(7):122–125, 1994.

[3] R. Beale and C. Creed. Affective interaction: How emotional agents affect users. *International Journal of Human-Computer Studies*, 67(9):755–776, 2009.

[4] R. H. Bordini, J. F. Hübner, and M. Wooldridge. *Programming multi-agent systems in AgentSpeak using Jason*, volume 8. John Wiley & Sons, 2007.

[5] J. Broekens, D. Degroot, and W. A. Kosters. Formal models of appraisal: Theory, specification, and computational model. *Cognitive Systems Research*, 9(3):173–197, 2008.

[6] J. Dias, S. Mascarenhas, and A. Paiva. Fatima modular: Towards an agent architecture with a generic appraisal framework. In *Emotion Modeling*, pages 44–56. Springer, 2014.

[7] J. Dias and A. Paiva. Feeling and reasoning: A computational model for emotional characters. In *Progress in artificial intelligence*, pages 127–140. Springer, 2005.

[8] M. S. El-Nasr, J. Yen, and T. R. Ioerger. Flamefuzzy logic adaptive model of emotions. *Autonomous Agents and Multi-agent systems*, 3(3):219–257, 2000.

[9] C. D. Elliott. The affective reasoner: A process model of emotions in a multi-agent system. 1992.

[10] J. Gratch and S. Marsella. A domain-independent framework for modeling emotion. *Cognitive Systems Research*, 5(4):269–306, 2004.

[11] K. V. Hindriks. Programming rational agents in goal. In *Multi-Agent Programming:*, pages 119–157. Springer, 2009.

[12] R. S. Lazarus. *Emotion and adaptation.* Oxford University Press, 1991.

[13] S. Marsella, J. Gratch, and P. Petta. Computational models of emotion. *A Blueprint for Affective Computing-A sourcebook and manual*, pages 21–46, 2010.

[14] S. C. Marsella and J. Gratch. Ema: A process model of appraisal dynamics. *Cognitive Systems Research*, 10(1):70–90, 2009.

[15] A. Ortony, G. L. Clore, and A. Collins. *The cognitive structure of emotions.* Cambridge university press, 1990.

[16] A. Pokahr, L. Braubach, and W. Lamersdorf. Jadex: A bdi reasoning engine. In *Multi-agent programming*, pages 149–174. Springer, 2005.

[17] A. Popescu, J. Broekens, and M. van Someren. Gamygdala: an emotion engine for games. *Affective Computing, IEEE Transactions on*, 5(1):32–44, 2014.

[18] W. S. Reilly. Believable social and emotional agents. Technical report, DTIC Document, 1996.

[19] R. Reisenzein. Appraisal processes conceptualized from a schema-theoretic perspective: Contributions to a process analysis of emotions. 2001.

[20] R. Reisenzein. Emotions as metarepresentational states of mind: Naturalizing the belief–desire theory of emotion. *Cognitive Systems Research*, 10(1):6–20, 2009.

[21] R. Reisenzein. What is an emotion in the belief-desire theory of emotion? 2012.

[22] R. Reisenzein. Social emotions from the perspective of the computational belief-desire theory of emotion. In *The Cognitive Foundations of Group Attitudes and Social Interaction*, pages 153–176. Springer, 2015.

[23] P. Rizzo. Why should agents be emotional for entertaining users? a critical analysis. In *Affective interactions*, pages 166–181. Springer, 2000.

[24] S. Russell, P. Norvig, and A. Intelligence. A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, 25:27, 1995.

[25] K. R. Scherer. Appraisal theory. *Handbook of cognition and emotion*, pages 637–663, 1999.

[26] K. R. Scherer. Appraisal considered as a process of multilevel sequential checking. *Appraisal processes in emotion: Theory, methods, research*, 92:120, 2001.

[27] B. R. Steunebrink, M. Dastani, and J.-J. C. Meyer. The occ model revisited. In *Proc. of the 4th Workshop on Emotion and Computing*, 2009.