Verifiable Machine Ethics in Changing Contexts

Louise A. Dennis[1], **Martin Mose Benzen**[2], **Felix Lindner**[3], **Michael Fisher**[1]

[1]Department of Computer Science, University of Manchester,
[2] Management Engineering, Technical University of Denmark,
[3] Institute of Artificial Intelligence, Ulm University
{louise.dennis, michael.fisher}@manchester.ac.uk, mmbe@dtu.dk, felix.lindner@uni-ulm.de

Abstract

Many systems proposed for the implementation of ethical reasoning involve an encoding of user values as a set of rules or a model. We consider the question of how changes of context affect these encodings. We propose the use of a reasoning cycle, in which information about the ethical reasoner's context is imported in a logical form, and we propose that context-specific aspects of an ethical encoding be prefaced by a guard formula. This guard formula should evaluate to true when the reasoner is in the appropriate context and the relevant parts of the reasoner's rule set or model should be updated accordingly. This architecture allows techniques for the model-checking of agent-based autonomous systems to be used to verify that all contexts respect key stakeholder values. We implement this framework using the hybrid ethical reasoning agents system (HERA) and the model-checking agent programming languages (MCAPL) framework.

1 Introduction

Moor (2006) defines an *explicitly ethical* agent as one which reasons using some explicit representation of ethical concepts. Several such systems have been produced (for examples see (Tolmeijer et al. 2020; Nallur 2020)). Explicit representations of ethical concepts require some encoding of values and/or ethics in order to perform their reasoning. We will refer to this as the *ethical encoding*. This encoding may be, for example, utilities for outcomes or obligations and prohibitions. We are concerned with how this information can be transferred from one context to another.

A practical reality is that current reasoning systems are limited in the extent to which they can fully analyse context. This makes it difficult to provide an ethical encoding that does not refer explicitly to contexts. As a simple example, people in general need to be able to see during the evening but do not need to see at night. A utilitarian reasoning system calculates utilities for the consequences of an action so we can imagine providing different utilities for "lights on" depending upon whether it is evening or night time. Such considerations mean that machine ethics systems need some account for how their reasoning adapts to changing contexts.

We here identify three key questions relating to this issue:

1. Is it possible to provide a general account for how to include context in an ethical encoding even though these encodings themselves vary widely from system to system?
2. Similarly, is it possible to provide a generic architecture which allows a machine ethics system to reason with context specific ethical encodings?
3. Assuming users are, at least in part, responsible for creating context-specific ethical encodings and may do so at runtime, are there techniques which can assist in checking that an encoding is consistent with their values and those of other stakeholders?

We propose that a context is represented as a logical formula which acts as a *guard* on an update function for ethical encodings. An ethical reasoner can then be embedded in a *reasoning cycle* that gathers contextual information via a combination of perception and reasoning, updates its ethical encoding using the update function, reasons and then acts. Concepts and tools from the cognitive agent community provide a natural architecture for such a system and provide a framework for verification.

We have implemented our proposal using the *Hybrid Ethical Reasoning Agents* system (HERA) (Lindner, Bentzen, and Nebel 2017) and the model-checking agent programming languages (MCAPL) framework (Dennis 2018). HERA has developed an approach to machine ethics that is rich enough to support reasoning in a variety of ethical systems. This allows us to evaluate our framework for context specific ethical reasoning using different ethical encodings. Our implementation forms JUNO – a cognitive agent based system that contains a HERA ethical reasoner which can be instantiated to reason using different ethical systems and so use different ethical encodings. We show how a JUNO agent can be formally verified to ensure that each context preserves key user values. All the code and examples in this paper are available in the MCAPL distribution [1].

2 Background

Automated reasoning systems represent information to be reasoned about in a pure logical format. For such systems the problem of context-specificity is minor. Logical implication ($\phi \implies \psi$) can be used to express naturally that some ψ

[1] https://autonomy-and-verification.github.io/tools/mcapl

is true in the context where ϕ is true. We take this as our starting point. A context is identified as a logical formula ϕ that holds true in that context. We can then use the truth of ϕ to control the ethical encoding used by our system.

The field of cognitive agents (Bratman 1987) provides many theoretical and practical tools that allow some agent, operating in a changing environment, to employ logical reasoning to guide its actions. *Beliefs*, *Desires*, and *Intentions* (BDI) agent programming languages are based on cognitive agents and draw heavily from the logic programming paradigm (Rao and Georgeff 1991, 1995). BDI agents make decisions based on intuitive concepts of how an agent's beliefs and desires lead to particular choices. There are *many* different agent programming languages and agent platforms based, at least in part, on the BDI approach (e.g., (Bordini, Hübner, and Wooldridge 2007; Rao 1996)). At the core of most lies a *reasoning cycle*. Details vary but key aspects include polling an external environment for new perceptions which become beliefs which are then used to select actions. The MCAPL framework (Dennis 2018) provides tools for implementing BDI languages so that their programs can be verified using model-checking.

Unlike may programming paradigms, cognitive agent programming languages are designed on the assumption that the program exists as part of a wider system. Perceptions come in from this system and the agent performs actions that have effects in this system. This system, external to the agent, is generally referred to as the *environment*. Environments may be purely software environments or may include actions and perceptions from the real world. We assume that a context is detected by using logical deduction on some set of predicates that have been perceived, and that this set has been assembled externally to the agent by its environment.

The use of context to control computational reasoning has been well-studied, particularly in the areas of pervasive and mobile systems and in agent reasoning. A context can be something as simple as the location of the computing device or agent, or can involve rich information about, for instance, a user's physical, social and emotional state. Many techniques exist for detecting, identifying, characterizing and reasoning with contexts (see Bolchini et al. (2007) and Perera et al. (2014) for surveys). In our work we assume that contexts are detected through a simple process of perception and logical reasoning. We use simple consistency checks to avoid encountering conflicting contexts.

HERA

We use the HERA system (Lindner, Bentzen, and Nebel 2017) as an embedded ethical reasoner. HERA implements (among others) reasoners for Utiliarianism (Lindner and Bentzen 2017) and Kant's second formulation of the categorical imperative (Lindner and Bentzen 2018) (henceforth referred to as the "Kantian" reasoner). HERA agents are specified as models.

Utilitarian Model A Utilitarian HERA model is a tuple $\langle A, B, C, F, u, W\rangle$ where A, B and C are disjoint sets of propositional variables representing respectively actions available to the system; background information; and the consequences of actions or other events. F is a set of *mechanisms* which describe how the truth value of each consequence $c \in C$ depends upon the interpretation of the other variables in $A \cup B \cup C$ – for example, people can see if it is day time or if the lights are on. This is written as $people\_can\_see := day \vee lights\_are\_on$. u (the *utilities*) is a mapping from each $v \in A \cup B \cup C$ to a real number. W is a set of *interpretations* for the variables in $A \cup B$ in which precisely one variable in A is interpreted as true in each $w \in W$.

Kantian Model A Kantian HERA model is a tuple $\langle A, B, C, F, P, K, G, W\rangle$ where: A, B, C, F, and W are defined as for the utilitarian reasoner. P is a set of variables indicating the moral patients. $K : (A \cup C) \times P \rightarrow \{+, -\}$ (the *affects mapping*) is a partial function indicating whether and how some action or consequence affects some moral patient $p \in P$. $G : A \times C$ (the *goal relation*) is a relation that captures the end goals of each $a \in A$;

Combined Model As implemented, HERA uses one model for all reasoners. A specific reasoner accesses only those parts of the model relevant to it. We refer to this as the *combined* HERA *model* which is a tuple $\langle A, B, C, F, P, K, G, u, W\rangle$.

For u, K, and G (the utility and affects mappings and the goal relation) appearing in some HERA model, M, we use the notation $M \models u(v) = d$ — the utility of v is d in model M, $M \models K(v, p) = +$ (resp. $-$) — v affects p positively (resp. negatively) in model M; and $M \models (a, c) \in G$ — c is a goal of a in M.

Causality In both models we need to reason about the outcomes of actions. HERA's model of causality is based on Halpern's use of the concept of a but-for-cause (Halpern 2016) and is set out in (Lindner, Bentzen, and Nebel 2017). We do not have space to describe this in full here but note that it involves using the mechanisms, F, to calculate all the consequences $c \in C$ that are true in some interpretation W in which an action $a \in A$ is in interpreted as true and contrasting this with the set of consequences that are true in an interpretation that is identical to W save that a is interpreted as false. We write $F, w \models a \rightsquigarrow c$ to indicate that a causes c in some interpretation w using mechanisms, F.

Permissibility The utilitarian reasoner calculates the sum of the utilities of all the consequences of an interpretation and compares it to the all other interpretations. The only permissible actions are those where the interpretation in which they are true has an overall utility which is greater than or equal to the utility of all the other interpretations in M. This is more fully set out in (Lindner and Bentzen 2017).

Kant's second formulation of the categorical imperative states that someone cannot be used as a means to an end. In terms of the HERA formalism some moral patient, p, is the end of an action, $M, w \models end(p)$ if there is some consequence c_g such that $(a, c_g) \in G$ and $K(c_g, p) = +$ (*i.e.,* one of the goals of the action affected them positively). A moral patient is a means to that end, $M, w \models means(p)$ if there is some consequence c_m such that $F, w \models a \rightsquigarrow c_m$ and $F, w \models c_m \rightsquigarrow c_g$ and $K(c_m, p) = +/-$ (*i.e.,* some consequence caused by the action and needed to bring about the goal affects the patient, either positively or negatively). For HERA some option, w (and hence the action $a \in A$ that is true in w), is morally acceptable according to the categorical imperative if

$$M, w \models \forall p.\, means(p) \implies end(p)$$

This is more fully set out in (Lindner and Bentzen 2018).

We write $perm(H, M, a)$ to mean that HERA reasoner H considers a to be permissible given model M.

3 Context Sensitive Ethical Reasoners

We use an ethical guard formula embedded in a reasoning cycle to control the adaptation of ethical reasoners depending upon context. To start our development we need to define our concepts of an ethical reasoner and an ethical encoding.

Definition 1 *An ethical reasoner, ER, is a system which uses an* ethical encoding *to recommend (or allow) some action (or set of actions) given some situation. Formally we represent situations as sets of formulae from a language, $\mathcal{L}$, ethical encodings as a set or type EE, and actions as a set, $\mathcal{A}$. So an ethical reasoner is a function $ER : (\mathcal{L} \times EE \times \mathcal{A}) \rightarrow \mathcal{P}(\mathcal{A})$, where $\mathcal{P}$ is the* powerset *function. We are deliberately imprecise about $\mathcal{L}$, EE and $\mathcal{A}$ due to the diverse nature of proposed ethical reasoning systems.*

Note here that we are using the terminology of a situation to represent a specific problem – for instance a particular time and place, set of actors and circumstances in which an ethical decision must take place while a context describes a set of situations (although it should be noted that a situation can belong to several contexts). A context is a generalisation of situations to ones with similar features in which the same ethical encoding applies.

A *context specification* describes how a context is detected and an ethical encoding can be changed.

Definition 2 *(Context Specification) A* context specification *is a tuple, $\langle \phi, f_{cx} \rangle$ where ϕ is a formula in $\mathcal{L}$ and $f_{cx} : EE \rightarrow EE$ is an* update function *on ethical encodings.*

Given an ethical reasoner, ER, an ethical encoding $ee \in EE$, and a context specification $\langle \phi, f_{cx} \rangle$, $f_{cx}(ee)$ is a variant of ee appropriate to situations where ϕ holds.

Example 1 *Let us consider a simple ethical reasoner that reasons using utilities about whether to turn out the lights. Three contexts have been identified, day, evening and night. The reasoner considers the utilities of $lights\_off$ and $poor\_visibility$. The utility of the lights being off is slightly positive in all situations (since the lights use up electricity). However the utility of $poor\_visibility$ varies being negative during the day and evening and zero at night. This gives us two ethical encodings of interest, c_1 and c_2:*

$$c_1 = \{u(lights\_off) = 1, u(poor\_visibility) = -10\}$$
$$c_2 = \{u(lights\_off) = 1, u(poor\_visibility) = 0\}$$

We define two simple update functions. f_{c_1} takes any ethical encoding and returns c_1. f_{c_2} takes any ethical encoding and returns c_2. Thus we can construct three context specifications:

$$\langle day, f_{c_1} \rangle, \langle evening, f_{c_1} \rangle, \langle night, f_{c_2} \rangle$$

These tell us that during the day and evening we should use ethical encoding c_1 and at night we should use c_2.

Definition 3 *(Context Sensitive Ethical Reasoner) A* context sensitive ethical reasoner *is a tuple $\langle \mathcal{B}, ee, ER, CX \rangle$ consisting of a belief base, $\mathcal{B}$, ethical encoding ee, an ethical reasoner ER, and a set of context specifications, CX. The reasoner polls some external environment in order to update $\mathcal{B}$, then it examines the context specifications, $\langle \phi, f_{cx} \rangle \in CX$ and, forms a set of new candidate encodings $EE = \{ee' \mid ee' = f_{cx}(ee)$ where $\langle \phi, f_{cx} \rangle \in CX \wedge \mathcal{B} \models \phi\}$. It then merges EE into a single encoding using some conflict resolution strategy. When it is required to make an ethical judgement about some set of options it passes its current ethical encoding to the ethical reasoner for use.*

We do not specify here how the system should resolve conflicts between multiple applicable contexts.

Example 2 *In Example 1 we considered three context specifications for an ethical reasoner, ER, that reasoned using utilities. Let us suppose ER is responsible for energy saving within a smart home. The system checks the state of the world every 10 minutes. If a light is on it calculates the utility of the state where the light is switched off and if that utility is greater than 0 then it turns the light off. In calculating the state after a light is switched off the system is able to deduce that switching a light off during the day makes no difference to visibility, while switching a light off during the evening or at night will lead to poor visibility.*

Assume that last time ER polled the state of the world, the lights were off, it was night time and there was no visibility. The context sensitive ethical reasoner built from ER is thus:

$\mathcal{B}$	$night, lights\_off, poor\_visibility$
ee	c_2
ER	ER
CX	$\langle day, f_{c_1} \rangle, \langle evening, f_{c_1} \rangle, \langle night, f_{c_2} \rangle$

The next time it polls its environment it is now day time and the lights have been turned on. The reasoner's belief base is now $\{day\}$ and this triggers a context update using the context specification $\langle day, f_{c_1} \rangle$:

$\mathcal{B}$	day
ee	c_1
ER	ER
CX	$\langle day, f_{c_1} \rangle, \langle evening, f_{c_1} \rangle, \langle night, f_{c_2} \rangle$

Although the reasoner now assigns a negative utility to poor visibility, its internal reasoning determines that this will not be effected by switching off the lights and so the overall utility will still be greater than zero if the lights are switched off. It does this and the state becomes:

$\mathcal{B}$	$day, lights\_off$
ee	c_1
ER	ER
CX	$\langle day, f_{c_1}\rangle, \langle evening, f_{c_1}\rangle, \langle night, f_{c_2}\rangle$

4 JUNO

We embed HERA in a cognitive agent system. We describe this system by means of a formal operational semantics which we subsequently used to create an implementation. A key element of the semantics is a JUNO *model* from which we can construct a HERA model each time ethical reasoning is required. In order to do this we need to provide some machinery for constructing HERA models. We consider first the construction of an interpretation given some action a and some background facts, L, believed by the agent.

Definition 4 *(Interpretation for a) For some set of atoms, L, and an atom $a \notin L$ we define the interpretation construction function f_w for a as follows:*

$$f_w(a, L) = w_a$$

where

$$\begin{aligned} w_a(a) =& \top && \textit{(a is interpreted as true)} \\ w_a(b) =& \top && \textit{if } b \in L \\ w_a(b) =& \bot && \textit{otherwise} \end{aligned}$$

In a HERA's Kantian model, the user specifies the goals (the relation G) of each action using the goal relation. This potentially requires users to have a strong grasp of the causal networks that might be constructed via these mechanisms. From the pragmatic point of view we quickly realised it was difficult for users to specify these easily. However it was possible to create a framework that would allow us to infer these relations from a more easily specified concept of system goals, $\mathcal{G}$; the mechanisms, F; and the interpretation under consideration.

Definition 5 *(Goal Inference) For some set of mechanisms, F, system goals, $\mathcal{G}$, atoms L, and some action a, we define the goal inference function $f_G : (a, F, \mathcal{G}, L) \to G$ such that*

$$\forall a, c.\, (a, c) \in f_G(F, \mathcal{G}, L) \textit{ iff } c \in \mathcal{G} \wedge F, f_w(a, L) \models a \rightsquigarrow c$$

We write $f_G(J)$ for the goal inference function derived from some JUNO *model, J.*

Thus, some consequence is deemed a goal of an action, if it is caused by the action and is a system goal.

Definition 6 *(*JUNO *model) A* combined JUNO model *is a tuple $\langle A, B, C, F, P, K, \mathcal{G}, u, L\rangle$ where A, B, C, F, P, K and u are defined as for a* HERA *model. $\mathcal{G} \subseteq C$ are the system goals and $L \subseteq B$ is a set of atoms that are believed to be true at the current point in time.*

When we need to refer to some particular element, X, of a JUNO model, J, we will use the notation $J|_X$. So $J|_A$ is the set of action variables, A, in J and so on.

We now explain how we can construct a HERA model from a JUNO model.

Definition 7 *(Constructed* HERA *models) The set of interpretations for a* HERA *model constructed from a* JUNO *model J are: $f_W(J) = \{f_w(a, J|_L) \mid a \in J|_A\}$.*

M_J is a HERA *model constructed from a* JUNO *model J:*

$$M_J = \langle J|_A, J|_B, J|_C, J|_F, J|_P, J|_K, f_G(J), J|_u, f_W(J)\rangle$$

So a JUNO model shares its propositional variables, mechanisms, utility and affects mapping with the HERA model created from it. The goal relation is inferred from the system goals (Definition 5) and the set of interpretations are created using f_W (Definition 7). There is one interpretation for each action in the JUNO model.

The complexity of constructing the goal relation is cubic in $\mid A \cup C \mid$ (for each action a and each goal g we must determine whether $F, a, L \models a \rightsquigarrow g$.). The complexity of constructing the interpretations is linear (there is one for each action). Therefore the overall complexity of constructing a HERA model is polynomial in $\mid A \cup C \mid$.

Context Specifications in JUNO

Recall from Definition 2 that a context specification is a pair of a guard formula, ϕ, and an update function, f_{cx}.

Definition 8 *The* JUNO *update function f_{cx} is a tuple $\langle K_{cx}, \mathcal{G}_{cx}, u_{cx}, L_{cx}\rangle$. Given a combined* JUNO *model $J = \langle A, B, C, F, P, K, \mathcal{G}, u, L\rangle$, $f_{cx}(J) = \langle A, B, C, F, P, K', \mathcal{G}', u', L'\rangle$ where:*

- *$K_{cx}(A \cup C, P) \to \{+, -\}$ is a partial function which updates the affects mapping such that*

  $$\forall v \in A \cup C, p \in P.$$

  $$K'(v, p) = \begin{cases} K_{cx}(v, p) & \textit{if } (v, p) \in \mathcal{D}(K_{cx}) \\ \textit{K(v, p)} & \textit{otherwise} \end{cases}$$

- *$\mathcal{G}_{cx} \subseteq C$ is a set of system goals that extend G: $\mathcal{G}' = \mathcal{G} \cup \mathcal{G}_{cx}$*
- *$u_{cx} : C \to \mathbb{R}$ is a partial function which updates the utilities such that*

  $$\forall c \in C.\, u'(c) = \begin{cases} u_{cx}(c) & \textit{if } c \in \mathcal{D}(u_{cx}) \\ u(c) & \textit{otherwise} \end{cases}$$

- *$L_{cx} \subseteq C$ is a set of ground first-order predicates that extend L: $L' = L_{cx} \cup L$*

Because $\mathcal{G}_{cx}$ and L_{cx} extend the sets in the model, the model they apply to needs to be minimal. Therefore JUNO agents operate with a *default model*, J_d to which all context updates are applied.

Example 3 *Let us consider the default model for K, $\mathcal{G}$, u and L in some smart home system that can evacuate the building, and controls the lights. The moral patients are the residents of the house:*

K	$K(safe, residents) = +$
$\mathcal{G}$	$\{safe\}$
u	$u(safe) = 50$ $u(electricity\_consumed) = -1$ $u(can\_see) = 10$
L	$\emptyset$

Let us consider a day time context:

ϕ	day
K_{cx}	
$\mathcal{G}_{cx}$	$\emptyset$
u_{cx}	
L_{cx}	$\{can\_see\}$

So the day time context does not create any goals, change utilities or affects relations but a daytime context does mean that people can see (i.e., without needing to turn lights on).

A night context might be:

ϕ	$night$
K_{cx}	$K_{night}(awake, residents) \quad = \quad -$
$\mathcal{G}_{cx}$	$\emptyset$
u_{cx}	$u_{night}(can\_see) \quad = \quad 0$ $u_{night}(awake) \quad = \quad -1$
L_{cx}	$\emptyset$

So at night the residents of the house are badly affected by being awake and gain no utility from being able to see.

If we update our model with the day context it becomes:

K	$K(safe, residents) = +$
$\mathcal{G}$	$\{safe\}$
u	$u(safe) = 50$ $u(electricity\_consumed) = -1$ $u(can\_see) = 10$
L	$\{can\_see\}$

Because each component of the update function describes *only* how the new model should differ from the default Juno model, we deal with multiple applicable contexts by applying multiple update functions one after the other. In order to avoid inconsistencies arising from the order in which update functions are applied we will require our context specifications to be *consistent*.

Definition 9 *(Consistent Context Specification) We say a set of context specifications, CX, is* consistent *if for $\langle \phi, f_{cx} \rangle$, $\langle \phi', f_{cx'} \rangle \in CX$ if $\phi \wedge \phi' \neq \bot$* (i.e., *there is some situation in which both update functions apply) then*

1. $\forall (v, p) \in D(K_{cx}) \cap D(K_{cx'}).\ K_{cx}(v, p) = K_{cx'}(v, p)$

2. $\forall c \in D(u_{cx}) \cap D(u_{cx'}).\ u_{cx}(c) = u_{cx'}(c)$

Note that because K_{cx} and u_{cx} are partial functions this does not imply that $K_{cx} = K_{cx'}$ nor that $u_{cx} = u_{cx'}$ only that they are identical where their domains intersect.

Given our assumption that contexts are consistent, the update process is simple and need only consider each context once in order to create a new Juno model. This therefore scales well as new contexts are added. The complexity of determining whether any given context specification, $\langle \phi, f_{cx} \rangle$ applies, depends upon the complexity of deciding ϕ. In our implementation the language, $\mathcal{L}$, of ϕ is propositional logic and a formula is evaluated against a belief base of propositions using the closed world assumption. Thus, deciding if a given context formula ϕ applies can be answered in linear time by fixing the truth value of the literals in ϕ to the ones they have in the agent's belief base.

We extend Hera to a cognitive agent system Juno:

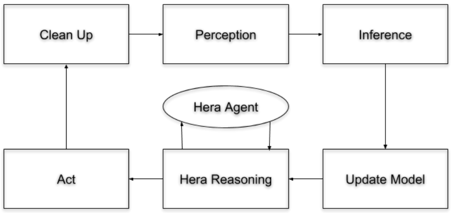

Figure 1: The Juno Reasoning Cycle

Definition 10 (Juno *agent)*

A Juno agent *is a tuple $\langle \mathcal{B}, J_d, J, H, \Delta, CX, a \rangle$ where $\mathcal{B}$ is a belief base, J_d is a* default Juno model*, J is the current* Juno *model, H is a* Hera *reasoner, Δ is a set of inference rules, CX is a set of consistent context specifications and a is the action last selected for execution.*

The inference rules Δ are structured as two sets, Δ_B and Δ_A. Each set consists of tuples $\langle \phi, B \rangle$ where ϕ is a formula and B is a set of atomic formulae in the language of $\mathcal{B}$.

Juno agents are context-sensitive ethical reasoners as defined in Definition 3, where J_d and J combine to form the ethical encoding and H is the ethical reasoner.

We use the Juno models J_d and J to construct Hera models for H as described in Definitions 6 and 8. The inference rules, Δ, allow Juno to create and manage temporal aspects of contexts. Juno agents use these rules to create obligations, represented as propositions, for some future point which are discharged by actions at that point [2].

A high level view of Juno's reasoning cycle is shown in Figure 1. Juno imports facts from the environment through a process of perception to form beliefs. Inference may occur (using rules from Δ_B) to generate additional beliefs [3]. Juno determines which context(s) it is currently in and updates the model by replacing the current model J with the default model J_d, then repeatedly applying all applicable update functions (Definition 8). A Hera model is constructed from the Juno model and passed to the Hera reasoner which produces a set of ethically acceptable actions. Juno selects an action from this set and executes it. After the action phase additional reasoning using rules from Δ_A may be applied to remove beliefs. The cycle then repeats.

This reasoning cycle is formally defined through an operational semantics that comprises a transition system on Juno agents and is shown in figure 2 where each equation refers to one phase in the reasoning cycle[4]. In these

[2] Much richer treatments of temporal aspects of contexts are obviously possible but we wished to keep things simple here.

[3] We note here that we have two sets of beliefs – those in the agent and those in the model. This need not necessarily be the case though has some advantages in terms of separating considerations around perception from those invovled with ethical reasoning.

[4] We obscure some details such has how transition between stages of the cycle is handled, and the details of the interaction with the environment. These are implemented in the same way as

transition rules ζ is the environment external to the agent which can return a set of *percepts* (which are labelled as from source $percept$) and execute an action. Hence (1) updates the beliefs with new percepts while removing any perception beliefs that are no longer observed. Rule (2) considers all the rules $\langle \phi, B \rangle \in \Delta_B$. Where ϕ follows by logical inference from the belief base the beliefs in B are added. Rule (3) repeatedly updates the JUNO model using all the context specifications that apply. (4) selects an ethical action from a set of permissible actions, determined using a HERA model, M_J, created from the JUNO model. If there is no permissible action then a default do_nothing action is selected. (5) executes the selected action. After an action is executed, (6) considers the rules $\langle \phi, B \rangle \in \Delta_A$. If ϕ is the last action executed then the beliefs in B are removed.

Several properties follow from this reasoning cycle. For instance: Any action performed by the system has been judged permissible by the HERA reasoner. Formally, if $select(Act) \in Act$ then if $\zeta.execute(a)$ is performed and $a \neq$ do_nothing then $perm(H, M_J, a)$.

We want this process to allow HERA to reason correctly in each context. This is difficult to capture formally but it is possible to prove a set of theorems that state that if some context update function specifies some property (for example, that a variable has a particular utility) then the HERA model will have that property when in the relevant context. Moreover all the HERA models assign either the value from the default model to these properties or a value from an applicable context update function.

5 Case Study

We consider the case of a JUNO agent operating in a smart home based on (Bentzen et al. 2018). It is able to turn lights on, turn the games console on, or evacuate the house in the case of a fire. If children are playing with the games console then they are quiet. Full formal models for this case study can be found in the MCAPL distribution. We consider key aspects informally here:

Day. During the day people are able to see. In the default JUNO model there is a small disutility for turning on the lights (it is, after all, bad for the environment).

Night $\wedge$ Awake. If it is night time and people are awake then people are not able to see (unless the lights are on), but they do gain utility from being able to see. Children are affected positively by playing games and there is no negative utility from them doing so.

Fire. If there is a fire then there is danger in the house. By default there is a small disutility for leaving the house, this becomes 0 in the case of fire.

Parent watching Television/Wrapping Presents. If a parent is watching television/wrapping presents (it is near Christmas!) then the system's goal is that (s)he should be able to watch TV/wrap presents; it is a background fact that (s)he wants to watch TV/wrap presents and there is utility from the children being quiet. In the default model, children are affected positively by a parent wrapping presents, but not by them watching TV [5].

Noise. If the children are not noisy then they are quiet.

Utilitarian Reasoning As contexts change in a JUNO agent that uses a utilitarian HERA reasoner, it chooses to refrain from action during a normal day (since people are safe — there is no fire — and they can see), turns on lights in the evening, evacuates in the case of fire, and switches on the video game if the children are being noisy and the parent wants to watch TV or wrap presents (unless there is a fire in which case it evacuates the house).

Kantian Reasoning Recall that in the Kantian reasoner a moral patient may not be used as the means to an end. The Kantian principle therefore only vetoes options, it has no mechanism for selecting one "good" option from among many. So, for instance in the evening, if the children are quiet, the Kantian principle does not distinguish between any of the available actions (it is as likely to evacuate the house as to turn on the lights).

Some other system is therefore needed to reason among the ethically acceptable options. In the operational semantics (Equation (4)), this process of selecting some action a' from among a set of ethically acceptable actions is written $a' = select(Act)$. For the purposes of testing we opted to reuse the utilitarian reasoner for this. So, first, the Kantian reasoner vetoes actions and then a utilitarian reasoner selects the most preferable from among those that remain.

Where changing contexts do have an effect, a Kantian JUNO agent, in contrast to the utilitarian agent, only turns on the games console if the children are noisy and the parent wants to wrap presents (since in this case the children benefit from the goal of the action). Even if it was evening (when the children were allowed screen time), the Kantian agent would not turn on the video game if the purpose was to keep the children quiet so that a parent could watch TV.

Verifying Models

We hypothesise that users may wish to update the ethical encodings in a system as their perception of the world changes. For instance, their attitude towards screen time may change. In some cases it will be possible to use a trial and error approach to evaluate the outcomes of such change.

In other situations, particularly where safety is involved, such a trial and error approach won't be suitable. Furthermore there may be many situations where it is not the user alone who determines what counts as ethical: there may be legal and societal constraints the system must obey irrespective of the user's personal ethics and preferences.

Formal verification is the process of assessing whether a formal specification is satisfied on a particular formal description of a system. For a specific logical property, φ, there are many different approaches to this (Fetzer 1988; DeMillo, Lipton, and Perlis 1979; Boyer and Strother Moore 1981). We explore here the use of *model checking* (Clarke, Grumberg, and Peled 1999). This takes a model of the system

for the GWENDOLEN language (Dennis 2017).

[5] This assumption could be argued against but a discussion about the nuances of good parenting are not relevant here.

$$\frac{P = \zeta.Percepts \quad OP = \{b \mid b \in \mathcal{B} \backslash P \wedge source\_of(b) = percept\}}{\langle \mathcal{B}, J_d, J, H, CX, a \rangle \rightarrow \langle \mathcal{B} \backslash OP \cup P, J_d, J, H, CX, a \rangle} \tag{1}$$

$$\frac{B = \{b \mid \langle \phi, \Phi \rangle \in \Delta_B \wedge \mathcal{B} \models \phi \wedge b \in \Phi\}}{\langle \mathcal{B}, J_d, J, H, CX, a \rangle \rightarrow \langle \mathcal{B} \cup B, J_d, J, H, CX, a \rangle} \tag{2}$$

$$\frac{J' = J_d \quad \texttt{for each} \ \langle \phi, f_{cx} \rangle \in CX \ \texttt{if} \ \mathcal{B} \models \phi \ \texttt{then} \ J' := f_{cx}(J')}{\langle \mathcal{B}, J_d, J, H, CX, a \rangle \rightarrow \langle \mathcal{B}, J_d, J', H, CX, a \rangle} \tag{3}$$

$$\frac{Act = \{a \mid a \in J|_A.\, perm(H, M_J, a)\} \quad \texttt{if} \ Act \neq \emptyset \ \texttt{then} \ a' = select(Act) \ \texttt{else} \ a' = \mathtt{do\_nothing}}{\langle \mathcal{B}, J_d, J, H, CX, a \rangle \rightarrow \langle \mathcal{B}, J_d, J, H, CX, a' \rangle} \tag{4}$$

$$\frac{\zeta.execute(a)}{\langle \mathcal{B}, J_d, J, H, CX, a \rangle \rightarrow \langle \mathcal{B}, J_d, J, H, CX, a \rangle} \tag{5}$$

$$\frac{B = \{b \mid \langle a, \Phi \rangle \in \Delta_A \wedge b \in \Phi\}}{\langle \mathcal{B}, J_d, J, H, CX, a \rangle \rightarrow \langle \mathcal{B} \backslash B, J_d, J, H, CX, a \rangle} \tag{6}$$

Figure 2: JUNO's Operational Semantics. Each equation applies to one phase in the reasoning cycle shown in figure 1. The expressions below the line indicate how the JUNO agent is transformed as a result of the transition, while above the line are given equations that instantiate variables, or side effects of the transition. (1) defines Perception; (2) defines Inference; (3) defines Update Model; (4) defines HERA reasoning; (5) defines Act; and (6) defines Clean Up.

in question, defining all the possible executions, and then checks a logical property against this model. One particularly attractive property of model checking is its ability to return a counter-example. In the case of a user changing the context specifications of a system such a counter-example will be able to inform them of a particular situation in which their changes cause some property to be violated and so will help them in understanding and correcting the problem.

The MCAPL Framework (Dennis 2018) in which we implemented JUNO supports a model-checking approach for the verification of agent-based systems outlined in (Dennis et al. 2016b). This considers the decisions taken by the system given any combination of incoming information. In the case of JUNO agents we consider this information to be the set of interpretations of guard formulae in context specifications and Δ_B then analyse all possible sequences of changes in these interpretations and thus all possible combinations of contexts and sequences of changing contexts.

For our case study we considered a simple safety property: In the case of a fire the agent will evacuate the house, expressed in AJPF's property specification language as:

$$\Box(\mathcal{B}(\mathit{fire}) \implies \Diamond\mathcal{D}(evacuate))$$

AJPF's property specification language is based on linear temporal logic where $\Box$ means *it is always the case that*, while $\Diamond$ means *it is eventually the case that*. AJPF provides special connectives to indicate agent concepts. $\mathcal{B}(\phi)$ means that ϕ is in the agent's belief base and $\mathcal{D}(\phi)$ means that the agent has attempted to perform the action ϕ.

We were able to automatically prove that the above property held in both our Utilitarian and Kantian agents.

6 Related Work

While there are several systems that implement explicit machine ethics none of these, to our knowledge, have considered in depth the interplay between the system's context, its ethical encoding and reasoning as the context changes. Of these systems $\mathcal{DCEC}_{CL}$ (Bringsjord, Arkoudas, and Bello 2008) reasons using automated reasoning with an explicit logical formalism. This means it can naturally incorporate our guard formulae into the rules provided to the reasoner.

Berreby, Bourgne, and Ganascia (2017) presents a framework based on answer set programming that can be used to model reasoning in a variety of ethical theories. In principle this should allow the natural incorporation of guard formulae, however the various implementations discussed assume unchanging representations of ethical encodings. An extension of these implementations to cope with variable contexts would, we contend, require a mechanism similar to ours.

The ETHAN system (Dennis et al. 2016a) does explicitly allow that its encoding is context specific – it anticipates some context-specific ranking of ethical concerns being supplied by some external agent and then reasons using that ranking but no account is provided for this process.

7 Further Work and Conclusions

We have described a theoretical framework for context-sensitive ethical encodings. The framework is abstract and generic in nature and while we have supplied a proof-of-concept instantiation of the framework which demonstrates it works with at least two ethical theories, there are a number of interesting avenues not explored by this proof of concept, in particular the issue of handling conflicting contexts.

Nevertheless our proof-of-concept demonstrates that an explicit ethical reasoners can be embedded in a *reasoning cycle* which controls updates to its ethical encodings by reasoning about contexts in a manner that is verifiable.

Acknowledgements This work was funded by EPSRC Grant EP/L024845/1 *Verifiable Autonomy* and EPSRC Grant EP/V026801/1 *Trustworthy Autonomous Systems Verifiability Node*. The work was enabled by a visiting researcher grant from the University of Liverpool. Fisher was funded by the Royal Academy of Engineering through a Chair in Emerging Technologies.

References

Bentzen, M.; Lindner, F.; Dennis, L.; and Fisher, M. 2018. Moral Permissability of Actions in Smart Home Systems. In *Proc. FLoC Workshop on Robots, Morality, and Trust through the Verification Lens*.

Berreby, F.; Bourgne, G.; and Ganascia, J.-G. 2017. A Declarative Modular Framework for Representing and Applying Ethical Principles. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '17, 96–104. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. URL http://dl.acm.org/citation.cfm?id=3091125.3091145.

Bolchini, C.; Curino, C. A.; Quintarelli, E.; Schreiber, F. A.; and Tanca, L. 2007. A Data-oriented Survey of Context Models. *SIGMOD Rec.* 36(4): 19–26. ISSN 0163-5808. doi:10.1145/1361348.1361353. URL http://doi.acm.org/10.1145/1361348.1361353.

Bordini, R.; Hübner, J.; and Wooldridge, M. 2007. *Programming Multi-agent Systems in AgentSpeak Using Jason*. Wiley.

Boyer, R. S.; and Strother Moore, J., eds. 1981. *The Correctness Problem in Computer Science*. Academic Press.

Bratman, M. E. 1987. *Intentions, Plans, and Practical Reason*. Harvard University Press. ISBN 1575861925.

Bringsjord, S.; Arkoudas, K.; and Bello, P. 2008. Toward a General Logicist Methodology for Engineering Ethically Correct Robots. *IEEE Intelligent Systems* 21(4): 38–44.

Clarke, E. M.; Grumberg, O.; and Peled, D. 1999. *Model Checking*. MIT Press.

DeMillo, R. A.; Lipton, R. J.; and Perlis, A. 1979. Social Processes and Proofs of Theorems of Programs. *ACM Communications* 22(5): 271–280.

Dennis, L.; Fisher, M.; Slavkovik, M.; and Webster, M. 2016a. Formal Verification of Ethical Choices in Autonomous Systems. *Robotics and Autonomous Systems* 77: 1–14. ISSN 0921-8890.

Dennis, L. A. 2017. Gwendolen Semantics: 2017. Technical Report ULCS-17-001, University of Liverpool, Department of Computer Science.

Dennis, L. A. 2018. The MCAPL Framework including the Agent Infrastructure Layer and Agent Java Pathfinder. *The Journal of Open Source Software* 3(24).

Dennis, L. A.; Fisher, M.; Lincoln, N.; Lisitsa, A.; and Veres, S. 2016b. Practical Verification of Decision-making in Agent-based Autonomous systems. *Automated Software Engineering* 23(3): 305–359. ISSN 0928-8910.

Fetzer, J. H. 1988. Program Verification: The Very Idea. *ACM Communications* 31(9): 1048–1063.

Halpern, J. Y. 2016. *Actual Causality*. MIT Press.

Lindner, F.; and Bentzen, M. 2017. The Hybrid Ethical Reasoning Agent IMMANUEL. In *Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 187–188.

Lindner, F.; and Bentzen, M. 2018. A Formalization of Kant's Second Formulation of the Categorical Imperative. In *Proceedings of The 14th International Conference on Deontic Logic and Normative Systems (DEON 2018)*.

Lindner, F.; Bentzen, M.; and Nebel, B. 2017. The HERA Approach to Morally Competent Robots. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*.

Moor, J. 2006. The Nature, Importance, and Difficulty of Machine Ethics. *IEEE Intelligent Systems* 21(4): 18–21. ISSN 1541-1672.

Nallur, V. 2020. Landscape of Machine Implemented Ethics. *Science and Engineering Ethics* doi:10.1007/s11948-020-00236-y. URL https://doi.org/10.1007/s11948-020-00236-y.

Perera, C.; Zaslavsky, A.; Christen, P.; and Georgakopoulos, D. 2014. Context Aware Computing for The Internet of Things: A Survey. *IEEE Communications Surveys Tutorials* 16(1): 414–454. doi:10.1109/SURV.2013.042313.00197.

Rao, A. 1996. AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. In *Agents Breaking Away: Proceedings 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, volume 1038 of *LNCS*, 42–55. Springer.

Rao, A. S.; and Georgeff, M. P. 1991. Modeling Agents within a BDI-Architecture. In *Proceedings 2nd International Conference Principles of Knowledge Representation and Reasoning (KR&R)*, 473–484. Morgan Kaufmann.

Rao, A. S.; and Georgeff, M. P. 1995. BDI Agents: From Theory to Practice. In *Proceedings 1st International Conference Multi-Agent Systems (ICMAS)*, 312–319. San Francisco, USA.

Tolmeijer, S.; Kneer, M.; Sarasua, C.; Christen, M.; and Bernstein, A. 2020. Implementations in Machine Ethics: A Survey. ArXiv:2001.07573 [cs.AI].

8 Ethics Statement

Our work fits within a general programme of ensuring that computational systems behave ethically and has been undertaken in the hopes that it will contribute to the aims of such research. However we are well aware that "intent is not magic". The particular approach taken to ethical behaviour in this paper presumes the ability of some person, or group of people, to provide an ethical encoding that captures stakeholder values. In parts of this paper we discuss the need for such encodings to be flexible over time, and for users to have some control over these, but we also touch on the need for wider societal values to be respected as well.

This naturally raises wider concerns such as:

1. *Whose ethics* are we capturing. Who is consulted as a stakeholder as part of the "ethical engineering" work and, more importantly, who is excluded from such consultation?
2. When should societal values take precedence over individual stakeholder values?

3. What are the consequences, and more importantly the mitigation measures in place, if the encoding fails to adequately capture stakeholder values?

While we believe that transparent, context-dependent expression of stakeholder values are important, we do not believe in any way that our work provides a complete solution to guaranteeing the ethical behaviour of such systems. It provides some technical machinery that we believe will assist in providing solutions but does not make the wider psychological, philosophical and social science problems disappear and should be viewed as one piece of the solution within a wider context.

It should also be noted that the same techniques we propose here for encoding context-dependent ethical behaviour could be analogously applied to provided context-dependent *unethical* behaviour. We believe that transparent, clear encodings make it *harder* within a societal context to develop and deploy deliberately unethical systems and are worth pursuing for that reason, but we can not pretend that, in and of themselves, these techniques prevent the possibility of misuse.

In short, the danger of work such as this is that it is viewed as a complete solution to the question of ethical artificial intelligence, rather than understood as a potential tool for providing solutions that must operate within a much wider multi-disciplinary context. As authors, we do our best to make this understanding clear when we engage with our peers, students and the general public and endeavour to take this wider context into account within our research programmes as a whole.