Modelling Morality with Prospective Logic

Luís Moniz Pereira[1] and Ari Saptawijaya[2]

[1] CENTRIA, Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
lmp@di.fct.unl.pt
[2] Fakultas Ilmu Komputer, Universitas Indonesia, 16424 Depok, Jawa Barat, Indonesia
saptawijaya@cs.ui.ac.id

Abstract. This paper shows how moral decisions can be drawn computationally by using prospective logic programs. These are employed to model moral dilemmas, as they are able to prospectively look ahead at the consequences of hypothetical moral judgments. With this knowledge of consequences, moral rules are then used to decide the appropriate moral judgments. The whole moral reasoning is achieved via a priori constraints and a posteriori preferences on abductive stable models, two features available in prospective logic programming. In this work we model various moral dilemmas taken from the classic trolley problem and employ the principle of double effect as the moral rule. Our experiments show that preferred moral decisions, i.e. those following the principle of double effect, are successfully delivered.

1 Introduction

Morality no longer belongs only to the realm of philosophers. Recently, there has been a growing interest in understanding morality from the scientific point of view. This interest comes from various fields, e.g. primatology [4], cognitive sciences [11, 18], neuroscience [23], and other various interdisciplinary perspectives [12, 14]. The study of morality also attracts the artificial intelligence community from the computational perspective, and has been known by several names, including machine ethics, machine morality, artificial morality, and computational morality. Research on modelling moral reasoning computationally has been conducted and reported on, e.g. AAAI 2005 Fall Symposium on Machine Ethics [10, 22].

There are at least two reasons to mention the importance of studying morality from the computational point of view. First, with the current growing interest to understand morality as a science, modelling moral reasoning computationally will assist in better understanding morality. Cognitive scientists, for instance, can greatly benefit in understanding complex interaction of cognitive aspects that build human morality or even to extract moral principles people normally apply when facing moral dilemmas. Modelling moral reasoning computationally can also be useful for intelligent tutoring systems, for instance to aid in teaching morality to children. Second, as artificial agents are more and more expected to be fully autonomous and work on our behalf, equipping agents with the capability to compute moral decisions is an indispensable requirement. This is particularly true when the agents are operating in domains where moral dilemmas occur, e.g. in health care or medical fields.

Our ultimate goal within this topic is to provide a general framework to model morality computationally. This framework should serve as a toolkit to codify arbitrarily chosen moral rules as declaratively as possible. We envisage that logic programming is an appropriate paradigm to achieve our purpose. Continuous and active research in logic programming has provided us with necessary ingredients that look promising enough to model morality. For instance, default negation is suitable for expressing exception in moral rules, abductive logic programming [13, 15] and stable model semantics [8] can be used to generate possible decisions along with their moral consequences, and preferences are appropriate for preferring among moral decisions or moral rules [5, 6].

In this paper, we present our preliminary attempt to exploit these enticing features of logic programming to model moral reasoning. In particular, we employ prospective logic programming [16, 19], an on-going research project that incorporates these features. For the moral domain, we take the classic trolley problem of Philippa Foot [7]. This problem is challenging to model since it contains a family of complex moral dilemmas. To make moral judgments on these dilemmas, we model the principle of double effect as the basis of moral reasoning. This principle is chosen by considering empirical research results in cognitive science [11] and law [18], that show the consistency of this principle to justify similarities of judgments by diverse demographically populations when given this set of dilemmas.

Our attempt to model moral reasoning on this domain shows encouraging results. Using features of prospective logic programming, we can conveniently model both the moral domain, i.e. various moral dilemmas of the trolley problem, and the principle of double effect declaratively. Our experiments on running the model also successfully deliver moral judgments that conform to the human empirical research results.

We organize the paper as follows. First, we discuss briefly and informally prospective logic programming, in Section 2. Then, in Section 3 we explain the trolley problem and the principle of double effect. We detail how we model them in prospective logic programming together with the results of our experiments regarding that model, in Section 4. Finally, we conclude and discuss possible future work, in Section 5.

2 Prospective Logic Programming

Prospective logic programming enables an evolving program to look ahead prospectively its possible future states and to prefer among them to satisfy goals [16, 19]. This paradigm is particularly beneficial to the agents community, since it can be used to predict an agent's future by employing the methodologies from abductive logic programming [13, 15] in order to synthesize and maintain abductive hypotheses.

Figure 1 shows the architecture of agents that are based on prospective logic [19]. Each prospective logic agent is equipped with a knowledge base and a moral theory as its initial theory. The problem of prospection is then of finding abductive extensions to this initial theory which are both relevant (under the agent's current goals) and preferred (w.r.t. preference rules in its initial theory). The first step is to select the goals that the agent will possibly attend to during the prospective cycle. Integrity constraints are also considered here to ensure the agent always performs transitions into valid evolution states. Once the set of active goals for the current state is known, the next step

is to find out which are the relevant abductive hypotheses. This step may include the application of a priori preferences, in the form of contextual preference rules, among available hypotheses to generate possible abductive scenarios. Forward reasoning can then be applied to abducibles in those scenarios to obtain relevant consequences, which can then be used to enact a posteriori preferences. These preferences can be enforced by employing utility theory and, in a moral situation, also moral theory. In case additional information is needed to enact preferences, the agent may consult external oracles. This greatly benefits agents in giving them the ability to probe the outside environment, thus providing better informed choices, including the making of experiments. The mechanism to consult oracles is realized by posing questions to external systems, be they other agents, actuators, sensors or other procedures. Each oracle mechanism may have certain conditions specifying whether it is available for questioning. Whenever the agent acquires additional information, it is possible that ensuing side-effects affect its original search, e.g. some already considered abducibles may now be disconfirmed and some new abducibles are triggered. To account for all possible side-effects, a second round of prospection takes place.

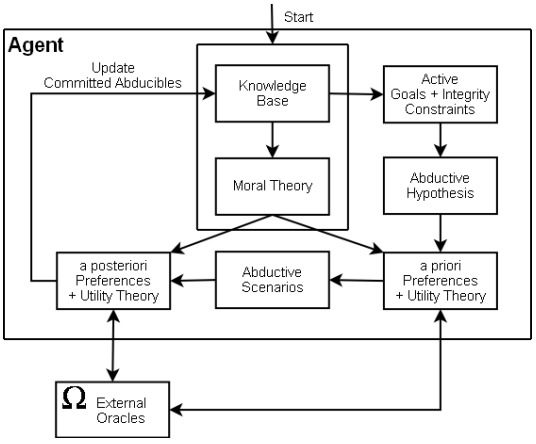

Fig. 1. Prospective logic agent architecture

ACORDA is a system that implements prospective logic programming and is based on the above architecture. ACORDA is implemented based on the implementation of EVOLP [1] and is further developed on top of XSB Prolog[3]. In order to compute abductive stable models [5, 6], ACORDA also benefits from the XSB-XASP interface to Smodels [4].

In this section, we discuss briefly and informally prospective logic programming and some constructs from ACORDA that are relevant to our work. For a more detailed

[3] XSB Prolog is available at `http://xsb.sourceforge.net`

[4] Smodels is available at `http://www.tcs.hut.fi/Software/smodels`

discussion on prospective logic programming and ACORDA, interested readers are referred to the original paper [16, 19].

2.1 Language

Let $\mathcal{L}$ be a first order language. A domain literal in $\mathcal{L}$ is a domain atom A or its default negation $not\ A$. The latter is to express that the atom is false by default (close world assumption). A domain rule in $\mathcal{L}$ is a rule of the form:

$$A \leftarrow L_1, \ldots, L_t. \qquad (t \geq 0)$$

where A is a domain atom and $L_1, \ldots, L_t$ are domain literals. An integrity constraint in $\mathcal{L}$ is a rule of the form:

$$\bot \leftarrow L_1, \ldots, L_t. \qquad (t > 0)$$

where $\bot$ is a domain atom denoting falsity, and $L_1, \ldots, L_t$ are domain literals. In ACORDA, $\leftarrow$ and $\bot$ are represented by `<-` and `falsum`, respectively.

A (logic) program P over $\mathcal{L}$ is a set of domain rules and integrity constraints, standing for all their ground instances.

2.2 Abducibles

Every program P is associated with a set of abducibles $A \subseteq \mathcal{L}$. Abducibles can be seen as hypotheses that provide hypothetical solutions or possible explanations of given queries.

An abducible A can be assumed only if it is a considered one, i.e. it is expected in the given situation, and moreover there is no expectation to the contrary [5, 6].

$$consider(A) \leftarrow expect(A), not\ expect\_not(A).$$

The rules about expectations are domain-specific knowledge contained in the theory of the program, and effectively constrain the hypotheses which are available.

In addition to mutually exclusive abducibles, ACORDA also allows sets of abducibles. Hence, an abductive stable model may contain more than a single abducible. To enforce mutually exclusive abducibles, ACORDA provides predicate `exclusive/2`. The use of this predicate will be illustrated later, when we model morality in a subsequent section.

2.3 A posteriori Preferences

Having computed possible scenarios, represented by abductive stable models, more favourable scenarios can be preferred among them a posteriori. Typically, a posteriori preferences are performed by evaluating consequences of abducibles in abductive stable models. The evaluation can be done quantitatively (for instance by utility functions) or qualitatively (for instance by enforcing some rules to hold). When currently available knowledge is insufficient to prefer among abductive stable models, additional information can be gathered, e.g. by performing experiments or consulting an oracle.

To realize a posteriori preferences, ACORDA provides predicate `select/2` that can be defined by users following some domain-specific mechanism for selecting favoured abductive stable models. The use of this predicate to perform a posteriori preferences in a moral domain will be discussed in a subsequent section.

3 The Trolley Problem and the Principle of Double Effect

Several interesting results have emerged from recent interdisciplinary studies on morality. One common result from these studies shows that morality has evolved over time. In particular, Hauser, in his recent work, argues that a moral instinct, playing the role of generating rapid judgments about what is morally right or wrong, has evolved in our species [11].

Hauser [11] and Mikhail [18] propose a framework of human moral cognition, known as universal moral grammar, analogously to Chomsky's universal grammar in language. Universal moral grammar, which can be culturally adjusted, provides universal moral principles that enable an individual to unconsciously evaluate what actions are permissible, obligatory, or forbidden. To support this idea, Hauser and Mikhail independently created a test to assess moral judgments of subjects from demographically diverse populations, using the classic trolley problem. Despite their diversity, the result shows that most subjects widely share moral judgments when given a moral dilemma from the trolley problem. Although subjects are unable to explain the moral rules in their attempts at justification, their moral judgments are consistent with a moral rule known as the principle of double effect.

The trolley problem presents several moral dilemmas that inquire whether it is permissible to harm one or more individuals for the purpose of saving others. In all cases, the initial circumstances are the same [11]:

> There is a trolley and its conductor has fainted. The trolley is headed toward five people walking on the track. The banks of the track are so steep that they will not be able to get off the track in time.

Given the above initial circumstance, in this work we consider six classical cases of moral dilemmas, employed for research on morality in people [18].

1. **Bystander.** Hank is standing next to a switch, which he can throw, that will turn the trolley onto a parallel side track, thereby preventing it from killing the five people. However, there is a man standing on the side track with his back turned. Hank can throw the switch, killing him; or he can refrain from doing this, letting the five die. Is it morally permissible for Hank to throw the switch?
2. **Footbridge.** Ian is on the footbridge over the trolley track. He is next to a heavy object, which he can shove onto the track in the path of the trolley to stop it, thereby preventing it from killing the five people. The heavy object is a man, standing next to Ian with his back turned. Ian can shove the man onto the track, resulting in death; or he can refrain from doing this, letting the five die. Is it morally permissible for Ian to shove the man?
3. **Loop Track.** Ned is standing next to a switch, which he can throw, that will temporarily turn the trolley onto a loop side track. There is a heavy object on the side track. If the trolley hits the object, the object will slow the train down, giving the five people time to escape. The heavy object is a man, standing on the side track with his back turned. Ned can throw the switch, preventing the trolley from killing the five people, but killing the man. Or he can refrain from doing this, letting the five die. Is it morally permissible for Ned to throw the switch?

4. **Man-in-front.** Oscar is standing next to a switch, which he can throw, that will temporarily turn the trolley onto a side track. There is a heavy object on the side track. If the trolley hits the object, the object will slow the train down, giving the five people time to escape. There is a man standing on the side track in front of the heavy object with his back turned. Oscar can throw the switch, preventing the trolley from killing the five people, but killing the man. Or he can refrain from doing this, letting the five die. Is it morally permissible for Oscar to throw the switch?
5. **Drop Man.** Victor is standing next to a switch, which he can throw, that will drop a heavy object into the path of the trolley, thereby stopping the trolley and preventing it from killing the five people. The heavy object is a man, who is standing on a footbridge overlooking the track. Victor can throw the switch, killing him; or he can refrain from doing this, letting the five die. Is it morally permissible for Victor to throw the switch?
6. **Collapse Bridge.** Walter is standing next to a switch, which he can throw, that will collapse a footbridge overlooking the tracks into the path of the trolley, thereby stopping the train and preventing it from killing the five people. There is a man standing on the footbridge. Walter can throw the switch, killing him; or he can refrain from doing this, letting the five die. Is it morally permissible for Walter to throw the switch?

Interestingly, although all cases have the same goal, i.e. to save five albeit killing one, subjects come to different judgments on whether the action to reach the goal is permissible or impermissible. As reported by Mikhail [18], the judgments appear to be widely shared among diverse demographically populations, the summary being given in Table 1.

Table 1. Summary of moral judgments for the trolley problem

Case	Judgment
1. Bystander	Permissible
2. Footbridge	Impermissible
3. Loop Track	Impermissible
4. Man-in-front	Permissible
5. Drop Man	Impermissible
6. Collapse Bridge	Permissible

Although subjects have difficulties to uncover which moral rules they apply for reasoning in these cases, their judgments appear to be consistent with the so-called the principle of double effect. The principle can be expressed as follows [11]:

> *Harming another individual is permissible if it is the foreseen consequence of an act that will lead to a greater good; in contrast, it is impermissible to harm someone else as an intended means to a greater good.*

The key expression here is “intended means”. We shall refer in the subsequent sections to the action of harming someone as an intended means, as an intentional killing.

4 Modelling Morality in ACORDA

It is interesting to model the trolley problem in ACORDA due to the intricacy that arises from the dilemma itself. Moreover, there are similarities and also differences between cases. Some cases even exhibit subtle differences. Consequently, this adds complexity to the process of modelling them in order to deliver appropriate moral decisions through reasoning. By appropriate moral decisions we mean the ones that conform with those the majority of people make, in adhering to the principle of double effect.

We model each case of the trolley problem in ACORDA separately. The principle of double effect is modelled via a priori constraints and a posteriori preferences. To assess how flexible is our model of the moral rule, we additionally model another variant for the cases of Footbridge and Loop Track. Even for these variants, our model of the moral rule allows the reasoning to deliver moral decisions as expected.

In each case of the trolley problem, there are always two possible decisions to make. One of these is the same for all cases, i.e. letting the five people die by merely watching the train go straight. The other decision depends on the cases, i.e. between throwing the switch or shoving a heavy man with the purpose to save the five people, but also harming a person in the process.

In this work, these two possible decisions are modelled in ACORDA as abducibles. Moral decisions are made by computing abductive stable models and then preferring among them those models with the abducibles and consequences that conform to the principle of double effect.

Due to space constraints, in subsequent sections we only detail the model for the cases of Bystander and Footbridge in ACORDA. We also show how to model the principle of double effect. Then we present some results of running our models in the ACORDA system.

4.1 Modelling the Bystander Case

Facts to describe that there is a man (here, named John) standing on the side track can be modelled simply as the following:

```
side_track(john).
human(john).
```

The clauses `expect(watching)` and `expect(throwing_switch)` in the following model indicate that watching and throwing the switch, respectively, are two available abducibles, that represent possible decisions Hank has. The other clauses represent the chain of actions and consequences for every abducible.

The predicate `end(die(5))` represents the final consequence if `watching` is abduced, i.e. it will result in five people dying, whereas `end(save_men,ni_kill(N))` represents the final consequence if `throwing_switch` is abduced, i.e. it will save the five people without intentionally killing someone. The way of representing these two

consequences is chosen differently, because the different nature of these two abducibles. Merely watching the trolley go straight is an omission of action that just has negative consequence, whereas throwing the switch is an action that is performed to achieve a goal and additionally has negative consequence. Since abducibles in other cases of the trolley problem also share this property, this way of representation will be used throughout them. The predicate `observed_end` is used to encapsulate these two different means of representation, useful later when we model the principle of double effect, to avoid floundering.

```
expect(watching).
train_straight <- consider(watching).
end(die(5)) <- train_straight.
observed_end <- end(X).

expect(throwing_switch).
redirect_train <- consider(throwing_switch).
kill(1) <- human(X), side_track(X), redirect_train.
end(save_men,ni_kill(N)) <- redirect_train, kill(N).
observed_end <- end(X,Y).
```

We can model the exclusiveness of the two possible decisions, i.e. Hank has to decide either to throw the switch or merely watch, by using the `exclusive/2` predicate of ACORDA:

```
exclusive(throwing_switch,decide).
exclusive(watching,decide).
```

Note that the exclusiveness between two possible decisions also holds in other cases.

4.2 Modelling the Footbridge Case

We represent the fact of a heavy man (here, also named John) on the footbridge standing near to Ian similarly to the Bystander case:

```
stand_near(john).
human(john).
heavy(john).
```

We can make this case more interesting by additionally having another (inanimate) heavy object, e.g. rock, on the footbridge near to Ian and see whether our model of the moral rule still allows the reasoning to deliver moral decisions as expected:

```
stand_near(rock).
inanimate_object(rock).
heavy(rock).
```

Alternatively, if we want only to have either a man or an inanimate object on the footbridge next to Ian, we can model it by using an even loop over default negation:

```
stand_near(john) <- not stand_near(rock).
stand_near(rock) <- not stand_near(john).
```

In the following we show how to model the action of shoving an object as an abducible, together with the chain of actions and consequences for this abducible. The model for the decision of merely watching is the same as in the case of Bystander. Indeed, since the decision of watching is always available for other cases, we use the same modelling in every case.

```
expect(shove(X)) <- stand_near(X).
on_track(X) <- consider(shove(X)).
stop_train(X) <- on_track(X), heavy(X).
kill(1) <- human(X), on_track(X).
kill(0) <- inanimate_object(X), on_track(X).
end(save_men,ni_kill(N)) <- inanimate_object(X), stop_train(X),
                            kill(N).
end(save_men,i_kill(N)) <- human(X), stop_train(X), kill(N).
observed_end <- end(X,Y).
```

Note that the action of shoving an object is only possible if there is an object near Ian to shove, hence the clause `expect(shove(X)) <- stand_near(X)`. We also have two clauses that describe two possible final consequences. The clause with the head `end(save_men,ni_kill(N))` deals with the consequence of reaching the goal, i.e. saving five, but not intentionally killing someone (in particular, without killing anyone in this case). To the contrary, the clause with the head `end(save_men,i_kill(N))` expresses the consequence of reaching the goal but involving an intentional killing.

4.3 Modelling the Principle of Double Effect

The principle of double effect can be modelled by using a combination of integrity constraints and a posteriori preferences.

Integrity constraints are used for two purposes. First, we need to observe the final consequences or endings of each possible decision to enable us later to morally prefer decisions by considering the greater good between possible decisions. To achieve this, we can use the integrity constraint `falsum <- not observed_end`. This integrity constraint enforces all available decisions to be abduced together with their consequences, by computing all possible observable hypothetical endings using all possible abductions. Indeed, to be able to reach a moral decision, all hypothetical scenarios afforded by the abducibles must lead to an observable ending. Second, we also need to rule out impermissible actions, i.e. actions that involve intentional killing in the process of reaching the goal. This can be enforced by specifying the integrity constraint `falsum <- intentional_killing`. Intentional killing can be easily defined as follows:

```
intentional_killing <- end(save_men,i_kill(Y)).
```

The above integrity constraints serve as the first filtering function of our abductive stable models, by ruling out impermissible actions (the latter being coded by abducibles). In other words, integrity constraints already afford us with just those abductive stable models that contain only permissible actions.

Additionally, one can prefer among permissible actions those resulting in greater good. This can be realized by a posteriori preferences that evaluate the consequences of permissible actions and then prefer the one with greater good. The following definition of `select/2` achieves this purpose. The first argument of this predicate refers to the set of initial abductive stable models to prefer, whereas the second argument refers to the preferred ones. The auxiliary predicate `select/3` only keeps abductive stable models that contain decisions with greater good of consequences. In the trolley problem, the greater good is evaluated by a utility function concerning the number of people that die as a result of possible decisions. This is realized in the definition of predicate `select/3` by comparing final consequences that appear in the initial abductive stable models. The first clause of `select/3` is the base case. The second clause and the third clause together eliminate abductive stable models containing decisions with worse consequences, whereas the fourth clause will keep those models that contain decisions with greater good of consequences.

```
select(Xs,Ys) :- select(Xs,Xs,Ys).

select([],_,[]).
select([X|Xs],Zs,Ys) :-
   member(end(die(N)),X),
   member(Z,Zs),
   member(end(save_men,ni_kill(K)),Z), N > K,
   select(Xs,Zs,Ys).
select([X|Xs],Zs,Ys) :-
   member(end(save_men,ni_kill(K)),X),
   member(Z,Zs),
   member(end(die(N)),Z), N =< K,
   select(Xs,Zs,Ys).
select([X|Xs],Zs,[X|Ys]) :- select(Xs,Zs,Ys).
```

Recall the variant of the case Footbridge, where either a man or an inanimate object is on the footbridge next to Ian. This exclusive alternative is specified by an even loop over default negation and we have an abductive stable model that contains the consequence of letting die the five people when a rock next to Ian. This model is certainly *not* the one we would like our moral reasoner to prefer. The following replacement definition of `select/2` accomplishes this case.

```
select([],[]).
select([X|Xs],Ys) :-
   member(end(die(N)),X),
   member(stand_near(rock),X),
   select(Xs,Ys).
select([X|Xs],[X|Ys]) :- select(Xs,Ys).
```

It is important to note that in this case, since either a man or a rock is near to Ian, and the model with shoving a man is already ruled out by our integrity constraint, there is no need to consider greater good in terms of the number of people that die. This means, as shown subsequently, that only two abductive stable models are preferred, i.e. the model

with watching as the abducible whenever a man is standing near to Ian, the other being the model with shoving the rock as the abducible.

4.4 Running the Models in ACORDA

We report now on the experiments of running our models in ACORDA. Table 2 gives a summary of all cases of the trolley problem. Column Initial Models contains info about the abductive stable models obtained before a posteriori preferences are applied, whereas column Final Models those after a posteriori preferences are applied. Here, only relevant literals are shown.

Note that entry Footbridge(a) refers to the variant of Footbridge where both a man and a rock are near to Ian, and Footbridge(b) where either a man or a rock is near to Ian. Loop Track(a) refers to the variant of Loop Track where there are two loop tracks, with a man on the left loop track and a rock on the right loop track. Loop Track(b) only considers one loop track where either a man or a rock is on the single loop track.

Table 2. Summary of experiments in ACORDA

Case	Initial Models	Final Models
Bystander	`[throwing_switch],[watching]`	`[throwing_switch]`
Footbridge(a)	`[watching],[shove(rock)]`	`[shove(rock)]`
Footbridge(b)	`[watching,stand_near(john)],` `[watching,stand_near(rock)],` `[shove(rock)]`	`[watching,stand_near(john)],` `[shove(rock)]`
Loop Track(a)	`[throwing_switch(right,rock)]` `[watching]`	`[throwing_switch(right,rock)]`
Loop Track(b)	`[watching,side_track(john)]` `[watching,side_track(rock)]` `[throwing_switch(rock)]`	`[watching,side_track(john)],` `[throwing_switch(rock)]`
Man-in-front	`[watching],` `[throwing_switch(rock)]`	`[throwing_switch(rock)]`
Drop Man	`[watching]`	`[watching]`
Collapse Bridge	`[watching]` `[throwing_switch(bridge)]`	`[throwing_switch(bridge)]`

These results comply with the results found for most people in morality laboratory experiments.

5 Conclusions and Future Work

We have shown how to model moral reasoning using prospective logic programming. We use various dilemmas of the trolley problem and the principle of double effect as the

moral rule. Possible decisions in a dilemma are modelled as abducibles. Abductive stable models are then computed which capture abduced decisions and their consequences. Models violating integrity constraints, i.e. models that contain actions involving intentional killing, are ruled out. Finally, a posteriori preferences are used to prefer models that characterize more preferred moral decisions, including the use of utility functions. These experiments show that preferred moral decisions, i.e. the ones that follow the principle of double effect, are successfully delivered. They conform to the results of empirical experiments conducted in cognitive science and law.

Much research has emphasized using machine learning techniques, e.g. statistical analysis [22], neural networks [10], case-based reasoning [17] and inductive logic programming [2] to model moral reasoning from examples of particular moral dilemmas. Our approach differs from them as we do not employ machine learning techniques to deliver moral decisions.

Powers proposes to use nonmonotonic logic to specifically model Kant's categorical imperatives [21], but it is unclear whether his approach has ever been realized in a working implementation. On the other hand, Bringsjord et. al. propose the use of deontic logic to formalize moral codes [3]. The objective of their research is to arrive at a methodology that allows an agent to behave ethically as much as possible in an environment that demands such behaviour. We share our objective with them to some extent as we also would like to come up with a general framework to model morality computationally. Different from our work, they use an axiomatized deontic logic to decide which moral code is operative to arrive at an expected moral outcome. This is achieved by seeking a proof for the expected moral outcome to follow from candidates of operative moral codes.

To arrive at our ultimate research goal, we envision several possible future directions. We would like to make a more declarative specification of a posteriori preferences, i.e. a specification that may encapsulate the details of predicate `select/2` from the viewpoint of users (cf. [20] for preliminary results). We also want to explore how to express metarule and metamoral injunctions. By metarule we mean a rule to resolve two existing conflicting moral rules in deriving moral decisions. Metamorality, on the other hand, is used to provide protocols for moral rules, to regulate how moral rules interact with one another. Another possible direction is to have a framework for generating precompiled moral rules. This will benefit fast and frugal moral decision making which is sometimes needed, cf. heuristics for decision making in law [9], rather than to have full deliberative moral reasoning every time.

We envision a final system that can be employed to test moral theories, and also can be used for training moral reasoning, including the automated generation of example tests and their explanation. Finally, we hope our research will help in imparting moral behaviour to autonomous agents.

References

[1] J. J. Alferes, A. Brogi, J. A. Leite, and L. M. Pereira. Evolving logic programs. In S. Flesca, S. Greco, N. Leone, and G. Ianni, editors, *Procs. 8th European Conf. on Logics in Artificial Intelligence (JELIA'02)*, LNCS 2424, pages 50–61. Springer, 2002.

[2] M. Anderson, S. Anderson, and C. Armen. MedEthEx: A prototype medical ethics advisor. In *Procs. 18th Conf. on Innovative Applications of Artificial Intelligence (IAAI-06)*, 2006.

[3] S. Bringsjord, K. Arkoudas, and P. Bello. Toward a general logicist methodology for engineering ethically correct robots. *IEEE Intelligent Systems*, 21(4):38–44, 2006.

[4] F. de Waal. *Primates and Philosophers, How Morality Evolved*. Princeton U. P., 2006.

[5] P. Dell'Acqua and L. M. Pereira. Preferential theory revision. In L. M. Pereira and G. Wheeler, editors, *Procs. Computational Models of Scientific Reasoning and Applications*, pages 69–84, 2005.

[6] P. Dell'Acqua and L. M. Pereira. Preferential theory revision (extended version). *Journal of Applied Logic (to appear)*, 2007.

[7] P. Foot. The problem of abortion and the doctrine of double effect. *Oxford Review*, 5:5–15, 1967.

[8] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. Kowalski and K. A. Bowen, editors, *5th Intl. Logic Programming Conf.* MIT Press, 1988.

[9] G. Gigerenzer and C. Engel, editors. *Heuristics and the Law*. MIT Press, 2006.

[10] M. Guarini. Particularism and generalism: how AI can help us to better understand moral cognition. In M. Anderson, S. Anderson, and C. Armen, editors, *Machine ethics: Papers from the AAAI Fall Symposium*. AAAI Press, 2005.

[11] M. D. Hauser. *Moral Minds, How Nature Designed Our Universal Sense of Right and Wrong*. Little Brown, 2007.

[12] R. Joyce. *The Evolution of Morality*. The MIT Press, 2006.

[13] A. Kakas, R. Kowalski, and F. Toni. The role of abduction in logic programming. In D. Gabbay, C. Hogger, and J. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 5, pages 235–324. Oxford U. P., 1998.

[14] L. D. Katz, editor. *Evolutionary Origins of Morality, Cross-Disciplinary Perspectives*. Imprint Academic, 2002.

[15] R. Kowalski. The logical way to be artificially intelligent. In F. Toni and P. Torroni, editors, *Procs. of CLIMA VI, LNAI*, page 122. Springer, 2006.

[16] G. Lopes and L. M. Pereira. Prospective logic programming with ACORDA. In *Procs. of the FLoC'06, Workshop on Empirically Successful Computerized Reasoning, 3rd Intl. J. Conf. on Automated Reasoning*, 2006.

[17] B. M. McLaren. Computational models of ethical reasoning: Challenges, initial steps, and future directions. *IEEE Intelligent Systems*, 21(4):29–37, 2006.

[18] J. Mikhail. Universal moral grammar: Theory, evidence, and the future. *Trends in Cognitive Sciences*, 11(4):143–152, April 2007.

[19] L. M. Pereira and G. Lopes. Prospective logic agents. In J. M. Neves, M. F. Santos, and J. M. Machado, editors, *Procs. 13th Portuguese Intl.Conf. on Artificial Intelligence (EPIA'07)*. Springer LNAI, December 2007.

[20] L. M. Pereira and A. Saptawijaya. Moral decision making with ACORDA. In N. Dershowitz and A. Voronkov, editors, *Short papers call, Local Procs. 14th Intl. Conf. on Logic for Programming Artificial Intelligence and Reasoning (LPAR'07)*, 2007.

[21] T. M. Powers. Prospects for a Kantian machine. *IEEE Intelligent Systems*, 21(4):46–51, 2006.

[22] R. Rzepka and K. Araki. What could statistics do for ethics? The idea of a commonsense-processing-based safety valve. In M. Anderson, S. Anderson, and C. Armen, editors, *Machine ethics: Papers from the AAAI Fall Symposium*. AAAI Press, 2005.

[23] L. Tancredi. *Hardwired Behavior, What Neuroscience Reveals about Morality*. Cambridge U. P., 2005.