To Do Something Else

Fengkui Ju

School of Philosophy, Beijing Normal University, Beijing, China
fengkui.ju@bnu.edu.cn

Jan van Eijck

Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
Institute for Logic, Language & Computation, University of Amsterdam
jve@cwi.nl

Abstract

This paper presents two deontic logics following an old idea: normative notions can be defined in terms of the consequences of performing actions. The two deontic logics are based on two special propositional dynamic logics; they interpret actions as sets of state sequences and have a process modality. The difference between the two deontic logics is that they contain different formalizations of refraining to do an action. Both of the two deontic logics have a propositional constant for marking the bad states. The normative notions are expressed by use of the process modality and this propositional constant.

Keywords: deontic logic, dynamic logic, process modality, negative action

1 Background

There is an old idea in the field of deontic logic: an action is *prohibited* if doing it would cause a bad thing; it is *permitted* if performing the action is possible without causing a bad thing; it is *obligated* if refraining to do it would cause a bad thing. This idea is intuitive in some sense; the point of it is that the three fundamental normative notions, *prohibition*, *permission* and *obligation*, can be defined in terms of the consequences of doing actions. According to [4], this idea can be traced back to Leibniz.

[1] and [9] independently develop this idea along similar lines. The resulting deontic logic has a modal operator $\Box$, the classical alethic modality whose dual is $\Diamond$. It also has a propositional constant $\mathcal{V}$ which intuitively means that what morality prescribes has been violated. The three normative notions are defined as follows: $\Box(\phi \to \mathcal{V})$ says that the proposition ϕ is prohibited, $\Diamond(\phi \wedge \neg\mathcal{V})$ says that ϕ is permitted and $\Box(\neg\phi \to \mathcal{V})$ says that ϕ is obligated. This logic applies deontic operators to propositions and does not really analyze actions. As mentioned in the literature, e.g., [10], this approach leads to quite a few problems.

Starting from the same idea, [11] proposes a different approach with emphasis on the analysis of actions in terms of their postconditions. In his dynamic logic $[\alpha]\phi$ expresses that no matter how the action α is performed, ϕ will be the case afterwards. The dual of $[\alpha]\phi$ is $\langle\alpha\rangle\phi$, which expresses that there is a way to perform α s.t. ϕ will be the case after α is done. The logic presented by [11] has a propositional constant $\mathcal{V}$ saying, again, that this is a *undesirable* state. By use of $[\alpha]\phi$ and $\mathcal{V}$, the three normative notions can be expressed: $[\alpha]\mathcal{V}$, meaning that α is prohibited, $\langle\alpha\rangle\neg\mathcal{V}$ indicating that α is permitted and $[\overline{\alpha}]\mathcal{V}$ denoting that α is obligated. By $\overline{\alpha}$, [11] intends to express this: to perform $\overline{\alpha}$ is to refrain from doing α. This work applies deontic operators to actions and many problems with previous deontic logics are avoided this way. [11] is a seminal paper that has given rise to a class of *dynamic deontic logics* following this approach.

There are two problems with [11]. The first one concerns the three normative notions. Whether an action α is prohibited/permitted/obligated or not is completely determined by whether the output of performing α is undesirable or not, and has nothing to do with what happens during the performance of α. As pointed out by [14], this is problematic, because it entails that while killing the president is prohibited, killing him and then surrendering to the police may not be, that while smoking in this room is not permitted, smoking in this room and then leaving it may be permitted, that while rescuing the injured and then calling an ambulance is obligated, rescuing the injured may not be. None of this sounds reasonable.

The second problem with [11] lies in how it technically deals with $\overline{\alpha}$. It presents a complicated semantics for actions. In short, it firstly assigns each action a so called s-trace-set; then it links each s-trace-set to a binary relation. In this way each action is interpreted as a binary relation. Essentially, this is like the standard semantics for actions from propositional dynamic logic (PDL). Under the semantics defined by [11], although $\overline{\alpha}$ is not the complement of α, still the behaviour of $\overline{\alpha}$ is not quite in line with the intuition of refraining from α. Firstly, the intersection of the interpretations of $\overline{\alpha}$ and α is not always empty, which would mean that in some states there may be ways to refrain from α while at the same time doing α. Secondly, the intersection of the interpretations of $\overline{\alpha}$ and $\alpha;\beta$ is not always empty, which would mean that in some cases, performing $\alpha;\beta$ is a way to refrain from doing α. This runs counter to our intuition about refraining from doing an action.

Indeed, [11] shows clear awareness of the requirement that $\overline{\alpha}$ and α should be disjoint and that $\overline{\alpha}$ and $\alpha;\beta$ should be disjoint as well. The correspondence between actions and s-trace-sets was designed to achieve this, but the assignment of binary relations to s-trace-sets results in some crucial information loss.

Dynamic logics in the style of PDL interpret actions as binary relations and can not deal with the progressive behaviour of actions. To solve this problem, so-called *process logics* take the intermediate states of doing actions into consideration and view actions as sets of sequences of states. Based on a process logic from [12], [14] proposes a deontic logic which aims to handle *free choice permission* and *lack-of-prohibition permission* in one setting. The sentence “you can use my pen or pencil” involves the former permission and “you can use his pen or pencil” involves the latter permission. The first sentence gives the addressee the permission to use *the* pen, but the second one does not. To see that the latter is the case, imagine a situation where the speaker of the second sentence is just reporting something by this sentence, and he/she knows that the owner of the pen and pencil allows the addressee to use the pen or pencil but does not know exactly which. Unlike [11], [14] does not introduce undesirable states, but uses undesirable transitions instead. The resulting logic allows description of the states during execution of actions and it avoids the first problem with [11]. However, the focus is on permission only, and there is no attempt to deal with refraining to do an action or with obligation.

Realizing that the formalization of refraining to do an action in [11] is problematic, [2] and [13] present alternative proposals, both based on a relational semantics for actions. The motivation of [2] is that the formalization in [11] can not be easily generalized to encompass iteration and converse of actions. [2] views $\overline{\alpha}$ as a constrained complement of α: $\overline{\alpha}$ is not the complement of α w.r.t. the universal relation, but the complement of α w.r.t. the set consisting of all the transitions resulting from performing actions constructed without use of $\overline{\ }$. Under this treatment, the intersection of the interpretations of $\overline{\alpha}$ and α is always empty; however, the problem with the intersection of the interpretations of $\overline{\alpha}$ and $\alpha;\beta$ remains: this intersection might not be empty. [13] thinks that the sentence “you are permitted either to eat the dessert or not” has different meaning from “you are permitted either to kiss me or not”, as the latter implies that the addressee may kiss the speaker but the former does not. The two sentences turn out equivalent. To remedy this, [13] interprets $\overline{\alpha}$ in a so called *stratified* way. Firstly, for any atomic action a with the interpretation R_a, it defines $R_{\overline{a}}$, the interpretation of $\overline{a}$, in the following way: a transition (w, u) is in $R_{\overline{a}}$ if and only if (w, u) is not in R_a but (w, x) is in R_a for some x; then by four inductive rules taken from [16], it defines the interpretation of $\overline{\alpha}$ for any compound action α. However, this approach suffers from the same problem as [11]: neither the intersection of $\overline{\alpha}$ and α nor the intersection of $\overline{\alpha}$ and $\alpha;\beta$ is always empty.

It is our aim in this paper to propose two deontic logics that follow the general approach of [11] but resolve the problems mentioned above.

2 Two Challenges

Two challenges are crucial in dynamic deontic logics: how to formalize refraining to do an action and how to handle the normative notions. We here state our ideas for these two issues, as a prelude to the two deontic logics to be presented below.

To refrain to do an action is *to do something else*. We think that to do something else meets the principle of symmetry: if doing α is doing something else than β, then doing β is also doing something else than α. We also think it is reasonable to impose the principle of perfect tense: deeds that are done remain done forever. In other words, for any action, if the agent has done it, then he/she will always have done it. Under the two principles, we do not have many choices in analyzing to do something else.

Let's look at an example. Let a and b be two different actions. Fix a start point. When would we say that the agent has done something else than $a;b$? Clearly, if the agent has done a, he/she has done something else than b. By the principle of the perfect tense, if he/she has done $a;b$, he/she has done something else than b. By the symmetry principle of to do something else, if he/she has done b, he/she has done something else than $a;b$. We can not say that if the agent has done a, he/she has done something else than $a;b$. Why? Because if an agent has done $a;b$ she has done a, by the principle of perfect tense. So if she has done a then it cannot be the case that she has done something else than $a;b$. We must therefore conclude that doing b is doing something else than doing $a;b$, but doing a is not doing something else than doing $a;b$.

About the issue of normative notions, we propose a sharpened version of the old idea mentioned in the previous section. There are a class of states, a group of people and an agent who might not belong to this group. The agent doing an action at a state might change this state to a different one. Some states are *bad* and others are *fine* for this group. An action of the agent is *prohibited* at a state *relative to* this group if the state will be bad at some point during any performance of this action. An action is *permitted* at a state if the state will always be fine during some performance of this action. An action is *obligated* at a state if the state will be bad at some point during any performance of *anything else*.

Next, how to formalize these ideas? In process logics such as those of [12] and [3], atomic actions are interpreted as sets of state sequences which might not be binary relations. [7] presents a simple process logic where atomic actions are viewed as binary relations and the action constructors of composition, union and iteration are treated in the usual way. We will follow this to formalize the notion of to do something else. Actually we will work this out in two different ways. As a follow-up to [7], [6] proposes two *process modalities* to describe what happens during execution of actions. One of them is called the $\forall\exists$ process modality. Below, we will use this modality plus a propositional constant to express the three normative notions.

3 A Deontic Logic Based on Process Theory

Let Π_0 be a *finite* set of atomic actions and Φ_0 a countable set of atomic propositions. Let a range over Π_0 and p over Φ_0. The sets Π_{PDL} of actions and Φ_{PDDL} of propositions are defined as follows:

$$\alpha ::= a \mid 0 \mid (\alpha;\alpha) \mid (\alpha \cup \alpha) \mid \alpha^*$$
$$\phi ::= p \mid \top \mid \mathfrak{b} \mid \neg\phi \mid (\phi \wedge \phi) \mid \|\alpha\|\phi$$

Here in "Φ_{PDDL}", "P" is for "process" and "DDL" for "dynamic deontic logic". 0 is the *impossible* action. $\mathfrak{b}$ means that this is a *bad* state. $\mathfrak{f}$, this is a *fine* state, is defined as $\neg\mathfrak{b}$. $\|\alpha\|\phi$ indicates that for any way to perform α, ϕ will be the case at some point in the process. The dual $\langle\!\langle\alpha\rangle\!\rangle\phi$ of $\|\alpha\|\phi$ is defined as $\neg\|\alpha\|\neg\phi$, which says that there is a way to perform α s.t. ϕ will be the case at all the points in the process. $F\alpha$, α is prohibited, is defined as $\|\alpha\|\mathfrak{b}$; it means that no matter how to perform α, the state will be bad at some point in the process. $P\alpha$, α is permitted, is defined as $\langle\!\langle\alpha\rangle\!\rangle\mathfrak{f}$; it means that there is a way to perform α s.t. the state will always be fine in the process. Other standard syntactic abbreviations apply here.

In next section, for any action α in Π_{PDL}, we will specify a β in Π_{PDL} and claim that to do something else but α is to do β. The special action 0 will be needed there. After that we will specify the formula saying that it is obligated to perform α.

$\mathfrak{M} = (W, \{R_a \mid a \in \Pi_0\}, B, V)$ is a model if

1. W is a nonempty set of states
2. for any $a \in \Pi_0$, $R_a \subseteq W \times W$, and for any $a, b \in \Pi_0$, $R_a \cap R_b = \varnothing$
3. $B \subseteq W$
4. V is a function from Φ_0 to 2^W

Atomic relations are pairwise disjoint. This constraint guarantees that syntactically different atomic actions are genuinely different. B is a set of bad states. $\overline{B}$, the complement of B, is the set of fine states. Note that there is no constraint on B; it could be the whole universe and could also be the empty set. A model is just a so called *interpreted labeled transition system* with the constraint that the relations are pairwise disjoint, plus a set of bad states.

Fix a model $\mathfrak{M} = (W, \{R_a \mid a \in \Pi_0\}, B, V)$. Define $\mathcal{R} = \bigcup\{R_a \mid a \in \Pi_0\}$. A sequence $w_0 \dots w_n$ of states is called a *trace* if $w_0\mathcal{R}\dots\mathcal{R}w_n$. Specially, for any $w \in W$, w is a trace. A trace represents a transition sequence made by doing a series of basic actions. A special trace w means doing nothing. Let $\mathcal{T}$ be the set of traces. Define a partial binary function *ext* on $\mathcal{T}$ as follows: $ext(u_0 \dots u_n, v_0 \dots v_m)$ equals $u_0 \dots u_n v_1 \dots v_m$ if $u_n = v_0$, otherwise it is undefined. Let S and T be two sets of traces. Define a function $\otimes$, called *fusion*, like this: $S \otimes T = \{ext(\kappa, \lambda) \mid \kappa \in S \;\&\; \lambda \in T$, and $ext(\kappa, \lambda)$ is defined $\}$. Each action α is interpreted as a set S_α of traces in the following way:

1. $S_a = R_a$
2. $S_{\beta;\gamma} = S_\beta \otimes S_\gamma$
3. $S_{\beta\cup\gamma} = S_\beta \cup S_\gamma$
4. $S_{\alpha^*} = W \cup S_\alpha \cup S_{\alpha;\alpha} \cup \dots$

This semantics for actions is called *trace semantics*. This semantics has the following feature: for any basic actions $a_1, \dots, a_n$, all the traces in $S_{a_1;\dots;a_n}$ contain $n+1$ states, provided it is given that $S_{a_1;\dots;a_n}$ is not empty.

$\mathfrak{M}, w \Vdash \phi$, ϕ being true at w in $\mathfrak{M}$, is defined as follows:

1. $\mathfrak{M}, w \Vdash p \Leftrightarrow w \in V(p)$
2. $\mathfrak{M}, w \Vdash \top$ always holds
3. $\mathfrak{M}, w \Vdash \mathfrak{b} \Leftrightarrow w \in B$
4. $\mathfrak{M}, w \Vdash \neg\phi \Leftrightarrow$ not $\mathfrak{M}, w \Vdash \phi$

5. $\mathfrak{M}, w \Vdash (\phi \wedge \psi) \Leftrightarrow \mathfrak{M}, w \Vdash \phi$ and $\mathfrak{M}, w \Vdash \psi$
6. $\mathfrak{M}, w \Vdash \|\alpha\|\phi \Leftrightarrow$ for any trace $w_0 \dots w_n$, if $w_0 = w$ and $w_0 \dots w_n \in S_\alpha$, then $\mathfrak{M}, w_i \Vdash \phi$ for some i s.t. $1 \leq i \leq n$

Recall the definitions of $F\alpha$ and $P\alpha$ above. It can be verified that

7. $\mathfrak{M}, w \Vdash \mathfrak{f} \Leftrightarrow w \in \overline{B}$
8. $\mathfrak{M}, w \Vdash \langle\!\langle\alpha\rangle\!\rangle\phi \Leftrightarrow$ there is a trace $w_0 \dots w_n$ s.t. $w_0 = w$, $w_0 \dots w_n \in S_\alpha$ and $\mathfrak{M}, w_i \Vdash \phi$ for any i s.t. $1 \leq i \leq n$
9. $\mathfrak{M}, w \Vdash F\alpha \Leftrightarrow$ for any trace $w_0 \dots w_n$, if $w_0 = w$ and $w_0 \dots w_n \in S_\alpha$, then $\mathfrak{M}, w_i \Vdash \mathfrak{b}$ for some i s.t. $1 \leq i \leq n$
10. $\mathfrak{M}, w \Vdash P\alpha \Leftrightarrow$ there is a trace $w_0 \dots w_n$ s.t. $w_0 = w$, $w_0 \dots w_n \in S_\alpha$ and $\mathfrak{M}, w_i \Vdash \mathfrak{f}$ for any i s.t. $1 \leq i \leq n$

Note that the semantics views the ending point of doing α as a point during the process of doing α but does not view the starting point as a point of the process.

The notions of *validity* and *satisfiability* are defined as usual. This logic is called PDDL. Illustrations of this logic will be given in **section 5** after we make it clear which formula expresses the obligation to do α.

4 To Do Something Else

In this section, we provide a formalization for the notion of *to do something else* following the idea stated in **section 2**.

A finite sequence of atomic actions is called a *computation sequence*, abbreviated as *seq*. The empty seq is denoted by ϵ and the set of seqs denoted by $\mathcal{CS}$. Each seq corresponds to a composition of atomic actions and seqs are understood by their corresponding actions. For any sets Δ and Θ of seqs, let $\Delta;\Theta = \{\gamma\delta \mid \gamma \in \Delta \ \&\ \delta \in \Theta\}$. $CS(\alpha)$, the set of the seqs of α, is defined as follows:

1. $CS(a) = \{a\}$
2. $CS(0) = \varnothing$
3. $CS(\alpha;\beta) = CS(\alpha);CS(\beta)$
4. $CS(\alpha \cup \beta) = CS(\alpha) \cup CS(\beta)$
5. $CS(\alpha^*) = \{\epsilon\} \cup CS(\alpha) \cup CS(\alpha;\alpha) \cup \dots$

Each seq of α represents a way to perform α. α is an *empty action* if $CS(\alpha) = \varnothing$. In the sequel, for any seq σ and set Δ of seqs, we use $\sigma\Delta$ to denote the set $\{\sigma\tau \mid \tau \in \Delta\}$. For any model, define S_ϵ, the interpretation of ϵ in this model, as the whole universe. It can be shown that $S_\alpha = \bigcup\{S_\sigma \mid \sigma \in CS(\alpha)\}$.

In the semantics defined in last section, atomic actions are interpreted as pairwise disjoint binary relations and compound actions are interpreted as sets of traces. As a result, the following proposition holds (assume again that we have fixed a model $\mathfrak{M}$, with traces computed in that model):

Proposition 4.1 *For any α and β, if $CS(\alpha) \cap CS(\beta) = \varnothing$, then $S_\alpha \cap S_\beta = \varnothing$.*

Proof. Assume $S_\alpha \cap S_\beta \neq \varnothing$. Let $w_0 \dots w_n$ be a trace in $S_\alpha \cap S_\beta$. Then there is a seq $a_1 \dots a_n$ in $CS(\alpha)$ and a seq $b_1 \dots b_n$ in $CS(\beta)$ s.t. $w_0 \dots w_n$ is in $S_{a_1;\dots;a_n}$ and $S_{b_1;\dots;b_n}$. Then for any i s.t. $1 \le i \le n$, $w_{i-1}w_i$ is in S_{a_i} and S_{b_i}. As atomic actions are pairwise disjoint, $a_i = b_i$ for any i s.t. $1 \le i \le n$. Then $a_1 \dots a_n = b_1 \dots b_n$. This means $CS(\alpha) \cap CS(\beta) \neq \emptyset$. □

This is a crucial fact for this work.

Let $\sqsubseteq$ denote the relation of *initial segment* for sequences and $\sqsupseteq$ the converse of $\sqsubseteq$, called *extension*.

Definition 4.2 [Mutual extension, x-difference] Let σ and τ be two seqs. Then $\sigma \approx \tau$ if if $\sigma \sqsubseteq \tau$ or $\tau \sqsubseteq \sigma$. Call this the relation of mutual extension. Say that σ is x-different from τ if $\sigma \not\approx \tau$.

For example, ac is x-different from ab, but a is not x-different from ab, as $a \sqsubseteq ab$. cab is also x-different from ab, as $ab \not\sqsubseteq cab$ and $cab \not\sqsubseteq ab$, although ab is a segment of cab. Here are some basic facts about the relation of x-difference. As ϵ is an initial segment of any seq, no seq is x-different from ϵ. x-difference is closed under extension: if $\sigma \not\approx \tau$ and $\tau \sqsubseteq \tau'$, then $\sigma \not\approx \tau'$. The relation of mutual extension is closed under initial segment: if $\sigma \approx \tau$ and $\tau' \sqsubseteq \tau$, then $\sigma \approx \tau'$. If σ is x-different from τ, then there is no way to extend σ s.t. the extension of σ is identical to τ, and there is also no way to extend τ s.t. the extension of τ is identical to σ. The notion of x-difference is intuitively understood as follows. Assume that σ is *x-different* from τ. Then there is no moment during the performance of σ at which the agent has done τ, and there is also no moment after the performance of σ at which the agent has done τ, no matter what he/she does afterwards.

For any actions α and β, α is *x-different* from β, $\alpha \not\approx \beta$, if for any seqs $\sigma \in CS(\alpha)$ and $\tau \in CS(\beta)$, $\sigma \not\approx \tau$. The relation of x-difference for actions formalizes the word "else" in the imperatives such as "don't watch cartoons anymore and do something else". β is *something else* but α if β is x-different from α. Note that given an action α, there might be many actions each of which is something else. For example, both b and c are something else for a. This means that the relation of x-different itself is not enough to handle the notion of to do something else, as the latter also involves a quantifier over actions. Luckily, for any α, among the actions which are something else, there is a *greatest* one in the sense that it is the union of all of them. This lets us deal with the notion of to do something else without introducing any quantifier over actions.

Definition 4.3 [The function of opposite] Let Δ be a set of seqs. $\widetilde{\Delta}$, the opposite of Δ, is defined as the set $\{\tau \mid \tau \not\approx \sigma \textit{ for any } \sigma \in \Delta\}$.

$\widetilde{\Delta}$ is always closed under extension; this is an important feature of the function of opposite. Opposite is different from complement: $\widetilde{\Delta}$ is always a subset of $\overline{\Delta}$, but not vice versa. Here is a counter-example: let $\Delta = \{ab\}$; then $a \in \overline{\Delta}$ but $a \notin \widetilde{\Delta}$. Opposite has certain connection with complement. Define Δ^T as the set of the seqs which are x-equal to some seq in Δ. Δ^T is called the *tree* generated from Δ. It can be seen that $\widetilde{\Delta} = \overline{\Delta^T}$. About Δ^T, there is a different way to look at it. Let Δ' be the smallest set which contains Δ and is closed under extension, and Δ'' the smallest set containing Δ' which is closed under initial segments. It can be verified that $\Delta'' = \Delta^T$. This result will be used later. Note that Δ^T might not be closed under extension.

The following proposition specifies some important properties of the function of opposite:

Proposition 4.4

1. $\Delta \cap \widetilde{\Delta} = \varnothing$
2. $\widetilde{\Delta} \cap (\Delta;\Theta) = \varnothing$
3. $\widetilde{\Delta \cup \Theta} = \widetilde{\Delta} \cap \widetilde{\Theta}$
4. $\Delta \subseteq \widetilde{\widetilde{\Delta}}$
5. $\widetilde{\Delta;\Theta} \subseteq \widetilde{\Delta} \cup (\Delta;\widetilde{\Theta})$ *if* $\Theta \neq \varnothing$
6. $\widetilde{\Delta} \subseteq \widetilde{\Delta;\Theta}$

Proof.

1. This is easy to show.

2. By the sixth item of this proposition, $\widetilde{\Delta} \subseteq \widetilde{\Delta;\Theta}$. As $\widetilde{\Delta;\Theta} \subseteq \overline{\Delta;\Theta}$, $\widetilde{\Delta} \subseteq \overline{\Delta;\Theta}$. Then $\widetilde{\Delta} \cap (\Delta;\Theta) = \varnothing$.

3. $\sigma \in \widetilde{\Delta \cup \Theta} \Leftrightarrow \sigma \not\approx \tau$ for any $\tau \in \Delta \cup \Theta \Leftrightarrow \sigma \not\approx \tau$ for any $\tau \in \Delta$ and $\sigma \not\approx \tau$ for any $\tau \in \Theta \Leftrightarrow \sigma \in \widetilde{\Delta}$ and $\sigma \in \widetilde{\Theta}$.

4. Let $\sigma \in \Delta$. Assume $\sigma \notin \widetilde{\widetilde{\Delta}}$. Then there is a $\tau \in \widetilde{\Delta}$ s.t. $\sigma \approx \tau$. This is impossible.

5. Let $\sigma \in \widetilde{\Delta;\Theta}$. Then $\sigma \not\approx \tau$ for any $\tau \in \Delta;\Theta$. Assume $\sigma \notin \widetilde{\Delta}$. We want to show $\sigma \in (\Delta;\widetilde{\Theta})$. Then there is a $\kappa \in \Delta$ s.t. $\sigma \sqsubseteq \kappa$ or $\kappa \sqsubseteq \sigma$. Assume $\sigma \sqsubseteq \kappa$. Let $x \in \Theta$, as $\Theta \neq \varnothing$. Then $\kappa x \in \Delta;\Theta$. As $\sigma \sqsubseteq \kappa$, $\sigma \sqsubseteq \kappa x$. Then $\sigma \approx \kappa x$. This is impossible, as $\sigma \in \widetilde{\Delta;\Theta}$. Then $\kappa \sqsubseteq \sigma$. Let $\sigma = \kappa\lambda$. We want to show $\lambda \in \widetilde{\Theta}$. Assume not. Then there is a $\tau \in \Theta$ s.t. $\lambda \approx \tau$. Then $\kappa\lambda \approx \kappa\tau$. Then $\kappa\tau \in \Delta;\Theta$. Then $\kappa\lambda \notin \widetilde{\Delta;\Theta}$. This is impossible. Then $\lambda \in \widetilde{\Theta}$. Then $\kappa\lambda \in (\Delta;\widetilde{\Theta})$, that is, $\sigma \in (\Delta;\widetilde{\Theta})$.

6. Let $\sigma \in \widetilde{\Delta}$. Then $\sigma \not\approx \tau$ for any $\tau \in \Delta$. Let $\tau' \in \Delta;\Theta$. Then there is a $\tau \in \Delta$ s.t. $\tau \sqsubseteq \tau'$. As $\not\approx$ is closed under extension, $\sigma \not\approx \tau'$. Then $\sigma \in \widetilde{\Delta;\Theta}$. □

The converse of the fourth item does not hold generally. As for any Δ, $\widetilde{\widetilde{\Delta}}$ is closed under extension, we can get that for any Δ, if Δ is not closed under extension, then $\widetilde{\widetilde{\Delta}} \nsubseteq \Delta$. Here is an example: let $\Pi_0 = \{a, b\}$ and $\Delta = \{aa, ab\}$; then $\widetilde{\Delta} = b\Pi_0^*$ and $\widetilde{\widetilde{\Delta}} = a\Pi_0^*$; then $aaa \in \widetilde{\widetilde{\Delta}}$ but $aaa \notin \Delta$. The converse of the fifth item does not hold either and the reason is that $(\Delta;\widetilde{\Theta}) \subseteq \widetilde{\Delta;\Theta}$ might not hold. What follows is a counter-example: let $\Pi_0 = \{a, b\}$, $\Delta = \{aa, a\}$ and $\Theta = \{ab\}$; then $\widetilde{\Theta} = b\Pi_0^* \cup aa\Pi_0^*$; then $aab \in \Delta;\widetilde{\Theta}$; as $aab \in \Delta;\Theta$, $aab \notin \widetilde{\Delta;\Theta}$. The fifth item has a condition, that is, $\Theta \neq \varnothing$. This item does not hold without the condition. For a counter-example, let $\Pi_0 = \{a, b\}$ and $\Delta = \{ab\}$. Then $\widetilde{\Delta;\Theta} = \mathcal{CS}$, as $\Delta;\Theta = \varnothing$. We see that $a \notin \widetilde{\Delta}$ and $a \notin \Delta;\widetilde{\Theta}$.

Proposition 4.5 *For any* $\alpha \in \Pi_{PDL}$, *there is a* $\beta \in \Pi_{PDL}$ *s.t.* $CS(\beta) = \widetilde{CS(\alpha)}$.

Proof. As shown in the literature of *automata theory*, a set Δ of seqs is a so called *regular language* if and only if there is a $\alpha \in \Pi_{\mathrm{PDL}}$ s.t. $CS(\alpha) = \Delta$ [1]. Therefore, it suffices to show that $\widetilde{CS(\alpha)}$ is a regular language. As mentioned in **section 4**, $\widetilde{CS(\alpha)} = \overline{CS(\alpha)^T}$ where $CS(\alpha)^T$ is the tree generated from $CS(\alpha)$. Then it suffices to show that $\overline{CS(\alpha)^T}$ is a regular language. Let Θ be the smallest set which contains $CS(\alpha)$ and is closed under extension. It can be seen that $CS(\alpha; (a_1 \cup \cdots \cup a_n)^*) = \Theta$ where $\Pi_0 = \{a_1, \ldots, a_n\}$. Then Θ is a regular language. Let Θ' be the smallest set containing Θ which is closed under initial segments. By [5], the closure of a regular language under initial segments is also a regular language. Then Θ' is a regular language. As stated in **section 4**, this Θ' equals to $CS(\alpha)^T$. Then $CS(\alpha)^T$ is a regular language. By [5], the complement of a regular language is also a regular language. Then $\overline{CS(\alpha)^T}$ is a regular language. □

This β is called the *opposite* of α, denoted by $\widetilde{\alpha}$. Here is an example: let $\Pi_0 = \{a, b, c\}$; then $\widetilde{a} = (b \cup c); (a \cup b \cup c)^*$. It can be easily shown that $CS(\widetilde{\alpha}) = \bigcup\{CS(\gamma) \mid \gamma \nsim \alpha\}$. Hence, $\widetilde{\alpha}$ is the union of all the actions which are something else but α. To refrain to do α is to do something else; to do *anything else* is to do $\widetilde{\alpha}$.

As mentioned in the introduction, it is reasonable to require that anything else but α has empty intersections with α and with $\alpha;\beta$. The following proposition states that this is indeed the case:

Proposition 4.6 *$S_{\widetilde{\alpha}} \cap S_\alpha = \varnothing$ and $S_{\widetilde{\alpha}} \cap S_{\alpha;\beta} = \varnothing$.*

This result can be proved by use of **proposition 4.1 and 4.4**.

In standard relational semantics, an action α is interpreted as a binary relation R_α. Then neither $R_{\widetilde{\alpha}} \cap R_\alpha = \varnothing$ nor $R_{\widetilde{\alpha}} \cap R_{\alpha;\beta} = \varnothing$ is generally the case even if atomic actions are pairwise disjoint. Here is a counter-example for both. Let a, b and c be three atomic actions. Let $R_a = \{(w_1, w_2)\}$, $R_b = \{(w_2, w_3)\}$ and $R_c = \{(w_1, w_3)\}$. We see that the three atomic actions are pairwise disjoint. As c is x-different from $a;b$ and $\widetilde{a;b}$ is the union of all the actions x-different from $a;b$, we know $R_c \subseteq R_{\widetilde{a;b}}$. As $R_c \cap R_{a;b} = \{(w_1, w_3)\}$, $R_{\widetilde{a;b}} \cap R_{a;b} \neq \varnothing$. c is x-different from a, then $R_c \subseteq R_{\widetilde{a}}$. $R_c \cap R_{a;b} = \{(w_1, w_3)\}$, then $R_{\widetilde{a}} \cap R_{a;b} \neq \varnothing$. In usual process logics, atomic actions are viewed as sets of state sequences which might not be binary relations. Then $S_{\widetilde{\alpha}} \cap S_\alpha = \varnothing$ and $S_{\widetilde{\alpha}} \cap S_{\alpha;\beta} = \varnothing$ do not generally hold, given that atomic actions are pairwise disjoint. What follows is a counter-example for both. Let $S_a = \{w_1 w_2\}$, $S_b = \{w_2 w_3\}$ and $S_c = \{w_1 w_2 w_3\}$. a, b and c are pairwise disjoint. c is x-different from $a;b$, then $S_c \subseteq S_{\widetilde{a;b}}$. $S_c \cap S_{a;b} = \{w_1 w_2 w_3\}$, then $S_{\widetilde{a;b}} \cap S_{a;b} \neq \varnothing$. c is x-different from a, then $S_c \subseteq S_{\widetilde{a}}$. $S_c \cap S_{a;b} = \{w_1 w_2 w_3\}$, then $S_{\widetilde{a}} \cap S_{a;b} \neq \varnothing$.

By **proposition 4.4** we can get that $S_\alpha \subseteq S_{\widetilde{\widetilde{\alpha}}}$ and $S_{\widetilde{\alpha;\beta}} \subseteq S_{\widetilde{\alpha}} \cup S_{\alpha;\widetilde{\beta}}$. It can be verified that neither of the converses of the two results holds. Considering that opposite is some type of negation, one might wonder about this. However, when restricted to the class of *normatively concise* actions, the two converses

[1] Regular languages are defined in terms of *finite deterministic automata*. For details of this, we refer to [5].

hold. What is a normatively concise action? Here we just show its idea by an example and does not give its formal definition. Assume that there are only two atomic actions: a and b. Look at the two sentences: "the agent ought to do $a; a$ or $a; b$" and "the agent ought to do a". The two sentences have the same meaning but the first one is not given concisely. In this sense, we say that the action $(a; a) \cup (a; b)$ is not normatively concise but a is. We leave exploring this issue further as our future work.

5 Validity

By means of *to do anything else*, we now can express obligations. $O\alpha$, α is *obligated*, is defined as $\|\widetilde{\alpha}\|\mathfrak{b}$; it means that no matter what alternative β to α is done, and now matter how β is performed, at some point in the process a bad state will be encountered. The truth condition of $O\alpha$ is as follows:

11. $\mathfrak{M}, w \Vdash O\alpha \Leftrightarrow$ for any trace $w_0 \ldots w_n$, if $w_0 = w$ and $w_0 \ldots w_n \in S_{\widetilde{\alpha}}$, then $w_i \Vdash \mathfrak{b}$ for some i s.t. $1 \leq i \leq n$

By now all the three normative notions are defined and we can illustrate the logic PDDL a bit.

PDDL has the following two features: its semantics does not take the starting point of doing an action as a point of the process of doing this action; whether an action is allowed is totally determined by what happens during the process of doing this action. The two features together imply whether an action is allowed at a state has nothing to do with this state. One may wonder what if the starting point of doing an action counts in the process of doing this action. Suppose so. Then $\phi \to \|\alpha\|\phi$ would be valid for any α and ϕ. Then both $\mathfrak{b} \to F\alpha$ and $\mathfrak{b} \to O\alpha$ would be valid. This means that in bad states, everything is forbidden and everything is obligated. This is of course undesirable. Our present definition at least has the advantage that it is possible to escape from a bad state with a good action.

There is some bonus which we can get from the two features mentioned above. For ease of stating our core points for refraining to do something, we in this work does not introduce the action constructor *test*. A test $\phi?$ in trace semantics is a set of states in which ϕ is true. As the starting point of doing an action does not count in the process of doing this action, the action of testing does not have a process. Then trivially, $\|\phi?\|\psi$ is not satisfiable and $\langle\!\langle \phi? \rangle\!\rangle\psi$ is valid. As a result, $F(\phi?)$ is not satisfiable and $P(\phi?)$ is valid. This means that there is no restriction on testing and testing is always free. Considering that testing is just some mental action and does not directly change the world, we think that this is desirable.

The following valid formulas express some connections between the deontic operators:

1. $P\alpha \leftrightarrow \neg F\alpha$
2. $O\alpha \leftrightarrow F\widetilde{\alpha}$
3. $P\alpha \to \langle\!\langle \alpha \rangle\!\rangle \top$

The first formula says that an action is permitted if and only if it is not forbidden. In addition, we can verify that $P(a \cup b) \to (Pa \wedge Pb)$ is not a valid formula. Putting the two facts together we can get that the operator P introduced in this work is not for the so called *free choice permission* but for *lack-of-prohibition permission*. The second formula tells that an action is obligated if and only if not doing it is forbidden. If an action is permitted, then it is *doable*; this is what the last formula says. *Kant's Law*, whatever should be done can possibly be done, expressed as $O\alpha \to \langle\!\langle \alpha \rangle\!\rangle \top$, does not generally hold in PDDL. To see this, imagine a model with a *dead* state, that is, one from which no transition starts. Then for any atomic action a, a is obligated trivially but not doable at this dead state.

What follows are some valid formulas which essentially involve action constructors:

1. $O\alpha \to O(\alpha \cup \beta)$
2. $F\alpha \to F(\alpha; \beta)$
3. $P(\alpha; \beta) \to P\alpha$
4. $O(\alpha; \beta) \to O\alpha$

The first formula shows that *Ross's Paradox* is not avoided: the agent has the duty to post the letter; therefore, he/she has the duty to post it or burn it. As argued in [8], we do not think that this is a problem. By the second formula, if killing is prohibited, then killing and then surrendering is also prohibited. But note this does not mean that if killing is prohibited, then surrendering is prohibited after killing. Indeed, it can be verified that $Fk \wedge \langle k \rangle Ps$ is satisfiable where k and s represent the actions of killing and surrendering respectively. By the third formula, if smoking and then leaving is permitted, then smoking is permitted. From the fourth formula we can get that the duty of rescuing the injured is implied by the duty of rescuing the injured and then calling an ambulance. These examples show that our logic does not suffer from the problem with [11] that was mentioned in the introduction.

Let's say that a state of a model is an *awkward* state if doing any atomic action at it will end in a bad state. Then at such states, for any atomic action a, a is not allowed. Then at them, nothing is allowed except those actions such as α^* and $\phi?$ which contain *one-element* traces. As a result, neither $O\alpha \to P\alpha$ nor $P\alpha \vee P\widetilde{\alpha}$ is valid.

6 A Variation

We put some constraints on the logic PDDL: in syntax, there are finitely many atomic actions and a special action 0; in semantics, atomic actions are pairwise disjoint. These constraints give PDDL the power to express *to do something else*. This is an implicit way to deal with to do something else. There is a different way to handle it, that is, explicitly introducing an action constructor for it.

Let Π_0 be a *countable* set of atomic actions and Φ_0 a countable set of atomic propositions. Let a range over Π_0 and p over Φ_0. The sets Π_{OPDL} of actions and Φ_{PoDDL} of propositions are defined as follows:

$$\alpha ::= a \mid (\alpha;\alpha) \mid (\alpha \cup \alpha) \mid \alpha^* \mid \widetilde{\alpha}$$
$$\phi ::= p \mid \top \mid \mathfrak{b} \mid \neg\phi \mid (\phi \wedge \phi) \mid \|\alpha\|\phi$$

Here in "Π_{OPDL}" and "Φ_{PoDDL}", "O" is for "opposite". The action $\widetilde{\alpha}$ is called the *opposite* of α; to do $\widetilde{\alpha}$ is to do something else but α. The intuitive reading of this language is as the language Φ_{PDDL} specified in **section 3**. $F\alpha$ and $P\alpha$ are defined as before and $O\alpha$ is directly defined as $\|\widetilde{\alpha}\|\mathfrak{b}$. Compared with Φ_{PDDL}, Φ_{PoDDL} has infinitely many atomic actions and does not have the empty action 0.

$\mathfrak{M} = (W, \{R_a \mid a \in \Pi_0\}, B, V)$ is a model where

1. W is a nonempty set of states
2. for any $a \in \Pi_0$, $R_a \subseteq W \times W$
3. $B \subseteq W$
4. V is a function from Φ_0 to 2^W

Models are understood as before. Here we do not require that atomic actions are pairwise disjoint.

Fix a model $\mathfrak{M} = (W, \{R_a \mid a \in \Pi_0\}, B, V)$. Recall that a sequence $w_0 \dots w_n$ of states is called a trace if $w_0 \mathcal{R} \dots \mathcal{R} w_n$ where $\mathcal{R} = \bigcup\{R_a \mid a \in \Pi_0\}$. Let $\mathcal{T}$ denote the set of traces as before. In **section 4**, we define a relation *x-different* on $\mathcal{CS}$ which is the set of computation sequences. Here we define it on $\mathcal{T}$ in a similar way: for any traces σ and τ, σ is x-different from τ, $\sigma \not\approx \tau$, if $\sigma \notin \tau$ and $\tau \notin \sigma$. By use of the relation x-different, we in **section 4** define a function *opposite* on the power set of $\mathcal{CS}$. We here define it on the power set of $\mathcal{T}$ similarly: for any set Δ of traces, let $\widetilde{\Delta}$, called the opposite of Δ, be the set $\{\tau \in \mathcal{T} \mid \tau \not\approx \sigma \textit{ for any } \sigma \in \Delta\}$. This opposite function also has the properties specified in **proposition 4.4**.

Each $\alpha \in \Pi_{\mathrm{OPDL}}$ is interpreted as a set S_α of traces in the following way:

1. $S_a = R_a$
2. $S_{\beta;\gamma} = S_\beta \otimes S_\gamma$
3. $S_{\beta\cup\gamma} = S_\beta \cup S_\gamma$
4. $S_{\alpha^*} = W \cup S_\alpha \cup S_{\alpha;\alpha} \cup \dots$
5. $S_{\widetilde{\alpha}} = \widetilde{S_\alpha}$

Here the operation $\otimes$ is defined as in **section 3**. We make a few points in this place. In **section 4**, we assign each α in Π_{PDL} an action $\widetilde{\alpha}$ in Π_{PDL}. The assignment makes use of the relation x-different and the function opposite; the action $\widetilde{\alpha}$ follows our idea for to do something else stated in **section 2**. In this section, $\widetilde{\alpha}$ is directly given in syntax; however, $S_{\widetilde{\alpha}}$, the interpretation of $\widetilde{\alpha}$, uses the relation of x-different and the function of opposite. Here $\widetilde{\alpha}$ also follows our idea for to do something else. $\mathcal{T}$ is the set of state sequences which can be made by performing basic actions. It can be seen that for any α, $S_{\widetilde{\alpha}} \subseteq \mathcal{T}$. This means that the action constructor $\sim$ does not essentially introduce new actions in this sense: whichever state can be reached by performing an action with $\sim$

can be reached by performing an action without $\tilde{}$.

$\mathfrak{M}, w \Vdash \phi$, ϕ being true at w in $\mathfrak{M}$, is defined as in **section 3**. The notion of *validity* is defined as usual. This logic is called PoDDL. A check of the formulas from **section 5** shows that the new approach does not make a difference for the validity/invalidity of these formulas.

7 Connections and Future Work

If we accept a state based approach of good and evil, it would be interesting to find out how the two ways of formalizing the notion of refraining to do something are related. Do they have the same expressive power or not? Next, it would be interesting to give complete axiomatisations.

The state based approach to the distinction between good and evil has some inherent limitations that carry over to our proposals above. As mentioned in **section 5**, almost nothing is allowed in the states we called awkward states. In reality, we never stop acting. Even if we are doing *nothing*, we are still doing *something*. There may be cases where, in order to act, we have to violate some prohibition. So what is prudent action in such situations? How should agents act in awkward states? Intuitively, they should transit to those states which are relatively *better* than others. Instead of a black and white division of evil and good states, we need some shades of grey, or even better a relational approach where some states are better than others. This is future work.

Since morality has to do with our interaction with others, another important step to take is from single agent to multiple agent deontic logic. Even more realistic seems an approach where obligations are relational, and where an obligation of some agent A to do something or to refrain from doing something is always an obligation to some other agent B. A proposal for a formalization of this idea in terms of propositional dynamic logic is given in [15]. One of the attractions of this is that it allows us to model conflicts of duty, such as the conflicts between professional obligations and family obligations that we all know so well.

Acknowledgment

Fengkui Ju was supported by the National Social Science Foundation of China (No. 12CZX053) and the Fundamental Research Funds for the Central Universities (No. SKZZY201304). We would like to thank the audience of the Workshop in Memory of Kang Hongkui and the anonymous referees for their useful comments and suggestions.

References

[1] Anderson, A. R., *Some nasty problems in the formal logic of ethics*, Noûs **1** (1967), pp. 345–360.

[2] Broersen, J., *Action negation and alternative reductions for dynamic deontic logics*, Journal of Applied Logic **2** (2004), pp. 153–168.

[3] Harel, D., D. Kozen and R. Parikh, *Process logic: Expressiveness, decidability, completeness*, Journal of computer and system sciences **25** (1982), pp. 144–170.

[4] Hilpinen, R., *Deontic logic*, in: L. Goble, editor, *The Blackwell Guide to Philosophical Logic*, Blackwell Publishing, 2001 pp. 159–182.
[5] Hopcroft, J., R. Motwani and J. Ullman, "Introduction to Automata Theory, Languages, and Computation," Pearson, 2006.
[6] Ju, F. and J. Cai, *Two process modalities in trace semantics* (2015), manuscript.
[7] Ju, F., N. Cui and S. Li, *Trace semantics for IPDL*, in: W. van der Hoek, W. H. Holliday and W. Wang, editors, *Logic, Rationality, and Interaction*, Lecture Notes in Computer Science **9394**, Springer Berlin Heidelberg, 2015 pp. 169–181.
[8] Ju, F. and L. Liang, *A dynamic deontic logic based on histories*, in: D. Grossi, O. Roy and H. Huang, editors, *Logic, Rationality, and Interaction*, Lecture Notes in Computer Science **8196**, Springer Berlin Heidelberg, 2013 pp. 176–189.
[9] Kanger, S., *New foundations for ethical theory*, in: R. Hilpinen, editor, *Deontic Logic: Introductory and Systematic Readings*, Springer Netherlands, 1971 pp. 36–58.
[10] McNamara, P., *Deontic logic*, in: E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, 2014, winter 2014 edition .
[11] Meyer, J.-J. C., *A different approach to deontic logic: deontic logic viewed as a variant of dynamic logic*, Notre Dame Journal of Formal Logic **29** (1988), pp. 109–136.
[12] Pratt, V., *Process logic*, in: *Proceedings of the 6th ACM Symposium on Principles of Programming Languages* (1979), pp. 93–100.
[13] Sun, X. and H. Dong, *Deontic logic based on a decidable PDL with action negation* (2015), manuscript.
[14] van der Meyden, R., *The dynamic logic of permission*, Journal of Logic and Computation **6** (1996), pp. 465–479.
[15] van Eijck, J. and F. Ju, *Modelling legal relations* (2016), under submission.
[16] Wansing, H., *On the negation of action types: constructive concurrent PDL*, in: P. Hájek, L. Valdes-Villanueva and D. Westerstahl, editors, *Proceedings of the Twelfth International Congress of Logic, Methodology and Philosophy of Science* (2005), pp. 207–225.