# A temporal dynamic deontic logic

FENGKUI JU, *School of Philosophy, Beijing Normal University, Beijing, China.*
Email: fengkui.ju@bnu.edu.cn

JAN VAN EIJCK, *Centrum Wiskunde and Informatica, Amsterdam, The Netherlands and Institute for Logic, Language and Computation, University of Amsterdam, Amsterdam, The Netherlands.*
Email: jve@cwi.nl



## Abstract

This paper presents a formalization of refraining from actions and a deontic logic based on a process logic. The notion of refraining is needed to handle obligated actions. To refrain to do an action is to do something else. The process logic used is a mix of dynamic logic and temporal logic: actions in it are interpreted as sets of paths and temporal formulas describe the process of performing actions. The deontic logic has a temporal propositional constant saying that a bad thing will be done in the next moment. Normative properties of actions can be defined according to what happens in the process of performing actions.

*Keywords*: To do something else, process logic, bad transitions, deontic logic.

## 1 Background

There is an old idea in deontic logic: an action is *prohibited* if doing it would bring about a *morally wrong* state; it is *permitted* if performing it is possible without having this effect; it is *obligated* if refraining to do it would bring about a morally wrong state. 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 [8], this idea can be traced back to Leibniz. In ethics, there are two opposing theories on moral properties of acts: *consequentialism* and *deontology*. The former holds that acts are morally assessed by the effects that they bring about. The latter claims that some acts cannot be justified by their effects no matter how good they are. In some sense, emphasis on consequences of actions is closer to consequentialism, emphasis on properties of actions is more closely related to deontology. But the system we will present in this paper is compatible with both traditions in moral philosophy.

Anderson [1] and Kanger [11] independently develop the idea of determining the moral value of actions by their consequences 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 $\phi$ is prohibited, $\Diamond(\phi \wedge \neg\mathcal{V})$ says that $\phi$ is permitted and $\Box(\neg\phi \to \mathcal{V})$ says that $\phi$ is obligated. This logic applies deontic operators to propositions and does not really analyse actions. As mentioned in the literature, e.g. [12], this approach leads to quite a few problems such as *the good samaritan paradox* and *the paradox of epistemic obligation*.

Propositional Dynamic Logic (PDL), introduced in [4], is a formal system for reasoning about the input/output behaviour of programmes of computers. The featured formulas of PDL are $[\alpha]\phi$

Vol. 29, No. 2, © The Author(s) 2019. Published by Oxford University Press. 
Published online 28 March 2018 doi:10.1093/logcom/exy008

and $\langle\alpha\rangle\phi$; the former expresses that no matter how the programme $\alpha$ is executed $\phi$ will be the case afterwards; the latter says that there is a way to execute $\alpha$ s.t. $\phi$ will be the case afterwards. PDL interprets programmes as binary relations. A binary tuple $(w, u)$ of states in the interpretation of a programme $\alpha$ means that the execution of $\alpha$ can cause the transition from $w$ to $u$. Programmes can also be viewed as actions of agents, therefore, PDL has potential application in other areas as well.

Based on a variant of PDL, [13] proposes a deontic logic following the ideas of [1] and [11]. This logic has a propositional constant $\mathcal{V}$ saying, again, that this is an *undesirable* state. The three normative notions can be expressed as follows: $[\alpha]\mathcal{V}$ meaning that $\alpha$ is prohibited, $\langle\alpha\rangle\neg\mathcal{V}$ indicating that $\alpha$ is permitted and $[\overline{\alpha}]\mathcal{V}$ denoting that $\alpha$ is obligated. By $\overline{\alpha}$, [13] intends to express that to perform $\overline{\alpha}$ is to refrain from doing $\alpha$. This work applies deontic operators to actions and many problems with previous deontic logics are avoided this way. The study in [13] is a seminal paper that has given rise to a class of *dynamic deontic logics* following this approach. Examples of this are [19] and [2].

There are two problems with [13]. The first one concerns the three normative notions. Whether an action $\alpha$ is prohibited/permitted/obligated or not is completely determined by whether the output of performing $\alpha$ is undesirable or not and has nothing to do with what happens during the performance of $\alpha$. As pointed out by [19], this is problematic. We remedy this by predicating 'badness' of actions rather than of their results. The action of killing the president is bad. After having killed the president, surrendering to the police is good. But this does not mean that first killing the president and then surrendering to the police is good.

The second problem with [13] 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 in PDL. Under the semantics defined by [13], although $\overline{\alpha}$ is not the complement of $\alpha$, still the behaviour of $\overline{\alpha}$ is not quite in line with the intuition of refraining from $\alpha$. In this semantics, 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 $\alpha$. Actually, even the intersection of the interpretations of $\overline{\alpha}$ and $\alpha$ is not always empty, which would mean that in some states there may be ways to refrain from $\alpha$ while at the same time doing $\alpha$. These outcomes run counter to our intuition about refraining from an action. Indeed, [13] shows clear awareness of the requirement that $\overline{\alpha}$ and $\alpha;\beta$ should be disjoint. In fact, the correspondence between actions and s-trace-sets was designed to achieve this, but unfortunately 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 cannot deal with the progressive behaviour of actions e.g. $\phi$ is true at some point during the performance of the action $\alpha$. To solve this problem, *process logics* are proposed which take the intermediate states of performing actions into consideration and view actions as sets of sequences of states. There are a variety of process logics including [15], [14], [7] and [21] and they are different from each other.

Observing that the first problem with [13] lies in that the internal structures of actions are absent in semantics, [19] proposes a deontic logic based on a process logic from [15]. This deontic logic aims to handle *free-choice* permission and *lack-of-prohibition* permission in one setting. The sentence 'you can sleep on our couch or on our sofa' involves the former permission and '(no doubt) you can sleep on their couch or on their sofa' involves the latter permission. The first sentence gives the addressee the permission to use either place to sleep, but the second one does not. Imagine a situation where the speaker of the second sentence is just reporting something, and he knows that the owner of the couch and sofa allows the addressee to sleep on one of those but does not know exactly which. Unlike [13], [19] does not introduce undesirable states but uses *permitted transitions* instead. The resulting logic allows description of the states during execution of actions and avoids

the first problem with [13]. However, the focus is on permission only, and there is no attempt to deal with refraining from an action or with obligation. Pucella and Weissman [16] extend the logic in [19] by introducing two dynamic operators: one adds and another removes permitted transitions. The two operators are used to model the dynamics of the so-called *policies*, which concern what is and what is not permitted.

Realizing that the formalization of refraining from an action in [13] is problematic, [2] and [17] present alternative proposals, both based on a relational semantics for actions. The motivation of [2] is that the formalization in [13] cannot be easily generalized to encompass iteration and converse of actions. Broersen [2] views $\overline{\alpha}$ as a constrained complement of $\alpha$: $\overline{\alpha}$ is not the complement of $\alpha$ w.r.t. the universal relation but the complement of $\alpha$ w.r.t. the set consisting of all the transitions resulting from performing actions constructed without use of the operator -. Under this treatment, the intersection of the interpretations of $\overline{\alpha}$ and $\alpha$ is always empty; however, the problem with the intersection of the interpretations of $\overline{\alpha}$ and $\alpha;\beta$ remains: the intersection might not be empty. The motivation of [17] concerns a puzzle about the normative sentences 'you are permitted either to eat the dessert or not' and 'you are permitted either to kiss me or not'. In the free-choice permission reading, the latter implies that the addressee may kiss the speaker but the former does not. However, the two factual sentences embedded in the two normative sentences, 'you eat the dessert or not' and 'you kiss me or not', are tautologies and equivalent.

To remedy this, [17] interprets $\overline{\alpha}$ in a so-called *stratified* way. Firstly, for every 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 $R_{\overline{a}}$ iff $(w, u)$ is not in $R_a$ but $(w, x)$ is in $R_a$ for some $x$; then by four inductive rules taken from [22], it defines the interpretation of $\overline{\alpha}$ for every compound action $\alpha$. However, this approach suffers from the same problem as [13]: neither the intersection of $\overline{\alpha}$ and $\alpha;\beta$ nor the intersection of $\overline{\alpha}$ and $\alpha$ is always empty.

We try to take one further step along this dynamic direction of deontic logic. We do two things. A new formalization for refraining from actions is proposed. We think that this formalization has intuitive support. A new deontic logic based on a variant of the process logic from [14] is presented. This process logic is a natural mix of PDL and Full Computation Tree Logic (CTL$^*$), introduced in [3], and can be used to talk about what happens during the performance of actions. By use of a temporal propositional constant saying that a bad behaviour will be made, this process logic can express more normative properties of actions than previous works. This paper mainly consists of two parts: Sections 2, 3 and 4 focus on refraining from an action, Sections 5, 6 and 7 on the deontic logic.

As Peter Thomas Geach discusses in his classic paper [6], many of the so-called paradoxes of deontic logic were caused by a shift from viewing *obligatory* and *permitted* as qualifications of *actions* to viewing them as qualifications of *result states of actions*. The present paper is in the tradition where this aberration is corrected, and as a result many of the paradoxes that were mentioned by Geach have disappeared. Geach calls the step from qualifying actions to qualifying states a 'fatal false step', and we agree.

Our framework deals with composite actions, so we can say things like *after doing $\alpha$, the agent is obliged to do $\beta$*. It does not matter whether $\alpha$ is good or bad. In case $\alpha$ is bad, talking about what is obliged after it gets performed is talking about contrary to duty obligation. For if $\alpha$ is bad, the agent has an obligation to avoid $\alpha$. Still, if the agent, contrary to duty, performs $\alpha$, we end up in a new situation where new obligations may hold. So this is a kind of conditional obligation. Still, in our framework alternative states of affairs are only accessible through action, so we cannot say things like *the world is $\phi$, but if the world would be $\neg\phi$ (different from how it actually is), then the agent would be obliged to do $\alpha$*. So in this sense, our framework does not handle conditional obligation.

## 2 Models

Let $\Pi_0$ be a *finite* set of atomic actions and $a$ range over $\Pi_0$. Define the set $\Pi_{\mathsf{PDL}}$ of actions as follows:

$$\alpha ::= a \mid \mathbf{0} \mid \mathbf{id} \mid (\alpha;\alpha) \mid (\alpha \cup \alpha) \mid \alpha^*$$

**0** indicates the *impossible* action. Doing **id** means doing *nothing*. **0** and **id** are also called atomic actions in the sequel. Define **1** as the action $\left(\bigcup \Pi_0\right) \cup \mathbf{0} \cup \mathbf{id}$; doing it means doing an atomic action.

Let $\Phi_0$ be a countable set of atomic propositions. A tuple $\mathfrak{M} = \left(W, \left\{R_a \mid a \in \Pi_0\right\}, R_{\mathbf{0}}, R_{\mathbf{id}}, B, V\right)$ is a model if

1. $W$ is a nonempty set of states,
2. for every $a \in \Pi_0$, $R_a \subseteq W \times W$; for every distinct $a, b \in \Pi_0$, $R_a \cap R_b = \varnothing$,
3. $R_{\mathbf{0}} = \varnothing$,
4. $R_{\mathbf{id}} = \left\{(x,x) \mid x \in W\right\}$,
5. $B \subseteq R$: for every $w$, there is a $u$ s.t. $(w,u) \in R - B$ where $R = \left(\bigcup \left\{R_a \mid a \in \Pi_0\right\}\right) \cup R_{\mathbf{0}} \cup R_{\mathbf{id}}$,
6. $V$ is a function from $\Phi_0$ to $2^W$.

Atomic actions are pairwise disjoint; this special constraint guarantees that syntactically different atomic actions are semantically different. Later we will see that this constraint serves for the formalization of refraining from something. $R_{\mathbf{id}}$ is the identity relation. So the action **id** just leaves states as they are. $R$ is *serial*, as $R_{\mathbf{id}}$ is. At every state, there is always something to do. $B$ is the set of *bad* transitions and $R - B$ the set of *fine* transitions. The extra constraint on $B$ is called *normative seriality*; it indicates that there is no state at which no fine transition can be made. The models defined here are the same as the models for PDL and CTL* if we ignore bad transitions.[1] Behind models, we presuppose that there are a group of people and an agent. The agent doing an action at a state might cause a transition to another state. Some transitions are bad and others fine for the group as a whole.

It is common in process logics that atomic actions are interpreted as arbitrary sets of sequences of states. Here we treat atomic actions as binary relations. The difference between the two ways is as follows. Interpreting atomic actions as binary relations indicates that reasoning is happening at the lowest level. Interpreting atomic actions as sets of state sequences indicates that reasoning is happening at a higher level and atomic actions are actually action schemas. However, as [21] argues, even if atomic actions are viewed as schemas, they are still not arbitrary.

Note that to consider an action as atomic is not to deny that the action may have internal structure; it simply means that this internal structure is supposed to be irrelevant. When we consider a traffic accident to be an atomic action, this does not mean that it does not have internal structure, it just means that the precise details of just how we wrecked our car do not matter.

Fix a model $\mathfrak{M} = \left(W, \left\{R_a \mid a \in \Pi_0\right\}, R_{\mathbf{0}}, R_{\mathbf{id}}, B, V\right)$. A *finite* sequence $w_0 \ldots w_n$ of states is called a *path* if $w_0 R \ldots R w_n$. Specially, $w$ is a path for every $w \in W$. A path represents a transition sequence made by performing a series of basic actions. The special path $w$ means doing nothing. Let $\mathcal{T}$ be the set of paths. Define a partial binary function *ext* on $\mathcal{T}$ as follows: $ext(u_0 \ldots u_n, v_0 \ldots v_m)$ equals to $u_0 \ldots u_n v_1 \ldots v_m$ if $u_n = v_0$, otherwise it is undefined. Let $S$ and $T$ be two sets of paths. Define a function $\otimes$ as follows: $S \otimes T = \left\{ext(\kappa, \lambda) \mid \kappa \in S \,\&\, \lambda \in T\right\}$. Each action $\alpha$ is interpreted as a set $S_\alpha$ of paths in the following way:

1. $S_a = R_a$
2. $S_{\beta;\gamma} = S_\beta \otimes S_\gamma$

[1] Atomic actions are implicit in CTL*.

3. $S_{\beta\cup\gamma} = S_\beta \cup S_\gamma$
4. $S_{\alpha^*} = W \cup S_\alpha \cup S_{\alpha;\alpha} \cup \ldots$.

A path in the interpretation of an action means that performing this action can cause the series of transitions represented by this path. This semantics for actions has the following feature: for every atomic actions $a_1, \ldots, a_n$, all the paths in $S_{a_1;\ldots;a_n}$ contain $n+1$ states if $S_{a_1;\ldots;a_n}$ is not empty.

Actions are interpreted as sets of paths; this makes it possible to talk about the process of performing actions. By resorting to whether a bad transition is made in the process of performing an action, we can define some normative properties of this action. This is what we do. But before getting there, let us first deal with the notion of refraining from an action, because we think that obligation is dependent on this notion.

## 3 Refraining from actions

We think that at every situation, all that the agent is able to do is just basic actions. To refrain from an action is not an independent action; it is just an economical expression of *to do something else*. A similar idea is also held by [2]. We think that to do something else meets *the principle of symmetry*: if doing $\alpha$ is doing something else than $\beta$, then doing $\beta$ is also doing something else than $\alpha$. It is reasonable to impose *the principle of perfect tense*: deeds that are done remain done forever. In other words, for every action, if the agent has done it, then he will always have done it. Under the two principles, we do not have many choices in analysing 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 has done something else than $b$. By the principle of the perfect tense, if he has done $a;b$, he has done something else than $b$. By the symmetry principle, if he has done $b$, he has done something else than $a;b$. We cannot say that if the agent has done $a$, he has done something else than $a;b$. Why? Assume so. Then by the principle of perfect tense, if he has done $a;b$, he has done something else than $a;b$, which is strange. We must therefore conclude that doing $b$ is doing something else than $a;b$, but doing $a$ is not doing something else than $a;b$. In what follows, for every $\alpha$ in $\Pi_{\mathsf{PDL}}$, we specify a $\beta$ in $\Pi_{\mathsf{PDL}}$ which means to do something else than $\alpha$ in the above sense.

Each finite sequence of atomic actions is called a *computation sequence*, abbreviated as a *seq*. Seqs are just strings of symbols. For example, *aabb* is a seq. The empty seq is denoted by $\epsilon$ and the set of seqs denoted by $\Pi_0^*$. Each nonempty seq corresponds to an action in a natural way. For example, *aabb* corresponds to $a;a;b;b$. For all sets $\Delta$ and $\Theta$ of seqs, let $\Delta;\Theta = \{\gamma\delta \mid \gamma \in \Delta \ \&\ \delta \in \Theta\}$. $CS(\alpha)$, the set of the seqs of $\alpha$, is defined as follows:

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

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

Atomic actions are interpreted as pairwise disjoint binary relations and compound actions are interpreted as sets of paths. As a result, the following proposition holds:

PROPOSITION 3.1
For every $\alpha$ and $\beta$, 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 path 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 every $i$ s.t. $1 \leq i \leq n$, $w_{i-1}w_i$ is in $S_{a_i}$ and $S_{b_i}$. As atomic actions are pairwise disjoint, $a_i = b_i$ for every $i$ s.t. $1 \leq i \leq n$. Then $a_1 \dots a_n = b_1 \dots b_n$. This means $CS(\alpha) \cap CS(\beta) \neq \varnothing$. □

This is a crucial fact for this work.

Let $\sqsubseteq$ denote the relation of *initial segment* and $\sqsupseteq$ the converse of it, called *extension*. In the sequel, we use this relation in the contexts of seqs and also state sequences. Let $\sigma$ and $\tau$ be two seqs. Define $\sigma \approx \tau$ iff $\sigma \sqsubseteq \tau$ or $\tau \sqsubseteq \sigma$. Call $\approx$ the relation of mutual extension. Say that $\sigma$ is *x-different* from $\tau$ if $\sigma \not\approx \tau$.[2] 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 $cab \not\sqsupseteq ab$, although *ab* is a segment of *cab*.

Here are some basic facts about *x*-difference. As $\epsilon$ is an initial segment of every seq, no seq is *x*-different from $\epsilon$. *x*-difference is closed under extension: if $\sigma \not\approx \tau$ and $\tau \sqsubseteq \tau'$, then $\sigma \not\approx \tau'$. Mutual extension is closed under initial segments: if $\sigma \approx \tau$ and $\tau' \sqsubseteq \tau$, then $\sigma \approx \tau'$. If $\sigma$ is *x*-different from $\tau$, then there is no way to extend $\sigma$ s.t. the extension of $\sigma$ is identical to $\tau$, and there is also no way to extend $\tau$ s.t. the extension of $\tau$ is identical to $\sigma$.

For all actions $\alpha$ and $\beta$, we say that $\alpha$ is *x-different* from $\beta$, $\alpha \not\approx \beta$, if for all 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'. $\beta$ is *something else* but $\alpha$ if $\beta$ is *x*-different from $\alpha$.

Given an action $\alpha$, there might be many actions each of which is something else than $\alpha$. For example, both *b* and *c* are something else for *a*. This means that the relation of *x*-difference itself is not enough to handle the notion of to do something else, as the latter also involves a quantifier over actions. Luckily, for every $\alpha$, among the actions which all are something else, there is a *greatest* one in the sense that the set of its seqs contains all the seqs of the others. This lets us deal with the notion of to do something else without introducing any quantifier.

DEFINITION 3.2 (The function of opposite)
Let $\Delta$ be a set of seqs. $\widetilde{\Delta}$, the opposite of $\Delta$, is defined as $\{\tau \mid \tau \not\approx \sigma \textit{ for every } \sigma \in \Delta\}$.

$\widetilde{\Delta}$ is always closed under extension; this is an important feature of it.

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 $\Delta^T$ as the set of the seqs which is *x*-equal to some seq in $\Delta$. $\Delta^T$ is called the *tree* generated from $\Delta$. It can be seen that $\widetilde{\Delta} = \overline{\Delta^T}$.

There is a different way to look at $\Delta^T$. Let $\Delta'$ be the smallest set which contains $\Delta$ and is closed under extension, and $\Delta''$ the smallest set which contains $\Delta'$ and is closed under initial segments. It can be verified that $\Delta'' = \Delta^T$. This result will be used later. Note that $\Delta^T$ might not be closed under extension.

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

[2]The reason we use the name '*x*-difference' is that we do not have a better name for this special notion of different.

PROPOSITION 3.3

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. $\widetilde{\Delta; \Theta} \subseteq \widetilde{\Delta} \cup \left(\Delta; \widetilde{\Theta}\right)$ if $\Theta \neq \varnothing$
6. $\widetilde{\Delta} \subseteq \widetilde{\Delta; \Theta}$.

PROOF.

1. Assume that there is a seq $\tau$ in $\Delta \cap \widetilde{\Delta}$. Then $\tau \not\approx \sigma$ for any $\sigma \in \Delta$. Then $\tau \not\approx \tau$. We have a contradiction.
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 every $\tau \in \Delta \cup \Theta \Leftrightarrow \sigma \not\approx \tau$ for every $\tau \in \Delta$ and $\sigma \not\approx \tau$ for every $\tau \in \Theta \Leftrightarrow \sigma \in \widetilde{\Delta}$ and $\sigma \in \widetilde{\widetilde{\Theta}}$.
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 every $\tau \in \Delta; \Theta$. Assume $\sigma \notin \widetilde{\Delta}$. We want to show $\sigma \in \left(\Delta; \widetilde{\Theta}\right)$. 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 $\kappa\lambda \in \left(\Delta; \widetilde{\Theta}\right)$, i.e. $\sigma \in \left(\Delta; \widetilde{\Theta}\right)$.
6. Let $\sigma \in \widetilde{\Delta}$. Then $\sigma \not\approx \tau$ for every $\tau \in \Delta$. 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. As for every $\Delta$, $\widetilde{\widetilde{\Delta}}$ is closed under extension, we can get that for every $\Delta$, if $\Delta$ is not closed under extension, then $\widetilde{\widetilde{\Delta}} \not\subseteq \Delta$. 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 $aaa \notin \Delta$. To simplify our statements, we do not consider the atomic actions **id** and **0** in this example. We will do this again in some examples in the sequel. The converse of the fifth item does not hold either and the reason is that $\left(\Delta; \widetilde{\Theta}\right) \subseteq \widetilde{\Delta; \Theta}$ 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, i.e. $\Theta \neq \varnothing$. This item does not hold without this condition. For a counter-example, let $\Pi_0 = \{a, b\}$ and $\Delta = \{ab\}$. Then $\widetilde{\Delta; \Theta} = \Pi_0^*$, as $\Delta; \Theta = \varnothing$. We see that $a \notin \widetilde{\Delta}$ and $a \notin \Delta; \widetilde{\Theta}$.

PROPOSITION 3.4
For every $\alpha \in \Pi_{\mathsf{PDL}}$, there is a $\beta \in \Pi_{\mathsf{PDL}}$ s.t. $CS(\beta) = \widetilde{CS(\alpha)}$.

PROOF. As shown in the literature of *automata theory*, a set $\Delta$ of seqs is a so-called *regular language* if and only if there is an $\alpha \in \Pi_{\mathsf{PDL}}$ s.t. $CS(\alpha) = \Delta$.[3] Therefore, it suffices to show that $\widetilde{CS(\alpha)}$ is a regular language. As mentioned in Section 3, $\widetilde{CS(\alpha)} = \overline{CS(\alpha)^T}$ 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 $\Theta$ be the smallest set

[3] Regular languages are defined in terms of *finite deterministic automata*. For details of this, we refer to [9].

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 $\Theta$ is a regular language. Let $\Theta'$ be the smallest set containing $\Theta$ which is closed under initial segments. By [9], the closure of a regular language under initial segments is also a regular language. Then $\Theta'$ is a regular language. $\Theta'$ equals to $CS(\alpha)^T$. Then $CS(\alpha)^T$ is a regular language. By [9], the complement of a regular language is also a regular language. Then $\overline{CS(\alpha)^T}$ is a regular language. □

This $\beta$ is called the *opposite* of $\alpha$, 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 $CS(\widetilde{\alpha}) = \bigcup \{CS(\gamma) \mid \gamma \not\approx \alpha\}$. Hence, $\widetilde{\alpha}$ is the union of all the actions which are something else but $\alpha$. To refrain to do $\alpha$ is to do something else; to do *anything else* is to do $\widetilde{\alpha}$.

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

PROPOSITION 3.5
$S_{\widetilde{\alpha}} \cap S_{\alpha;\beta} = \varnothing$ and $S_{\widetilde{\alpha}} \cap S_{\alpha} = \varnothing$.

This result can be proved by use of Propositions 3.1 and 3.3.

In standard relational semantics, an action $\alpha$ is just interpreted as a binary relation $R_\alpha$ instead of a set of paths. Then neither $R_{\widetilde{\alpha}} \cap R_\alpha = \varnothing$ nor $R_{\widetilde{\alpha}} \cap R_{\alpha;\beta} = \varnothing$ is 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. $c$ is $x$-different from $a$, then $R_c \subseteq R_{\widetilde{a}}$. $R_c \cap R_{a;b} = \{(w_1, w_3)\}$, then $R_{\widetilde{a}} \cap R_{a;b} \neq \varnothing$. As $c$ is $x$-different from $a;b$, $R_c \subseteq R_{\widetilde{a;b}}$. As $R_c \cap R_{a;b} = \{(w_1, w_3)\}$, $R_{\widetilde{a;b}} \cap R_{a;b} \neq \varnothing$.

In reality, we always view doing nothing as a way to refrain from something. This is what we introduce the special action **id** for.

Atomic actions in process logics are usually interpreted as arbitrary sets of state sequences. One may wonder whether the above formalization of refraining still works if so. Actually, there is no problem if we put as a constraint *the generalized pairwise disjointness* on atomic actions. Previously we specify a relation $x$-different: two sets $S$ and $T$ of seqs are $x$-different if there is no $\sigma \in S$ and $\tau \in T$ s.t. one is an initial segment of another. Define it among sets of state sequences in a similar way. Let $\widetilde{\alpha}$ be defined as above. It can be verified that Proposition 3.5 always holds once atomic actions are $x$-different from each other.

An interesting thing can happen to the notion of refraining when infinity is involved. We look at a special action $a^*; b$. To perform it is to perform $a$ for finitely many times and then perform $b$ once. It can be verified that $\widetilde{a^*; b}$ is an empty action. It can be seen that the infinite seq $aaa\ldots$ does not take any seq of $a^*; b$ as an initial segment. This means that if the agent keeps doing $a$ forever, there will never be a moment when we can say that he has done $a^*; b$ and a moment when we can say that he has done something else.

## 4 Concise actions

There is a special class of actions that we want to identify in the context of refraining. For an action $\alpha$, there might be another action $\beta$ s.t. refraining from $\alpha$ is the same as refraining from $\beta$. So the obligation to do $\alpha$ would be equivalent to the obligation to do $\beta$. For example, if $a$ and $b$ are the only atomic actions, then $\widetilde{a} = \widetilde{(a; a \cup a; b)} = b; (a \cup b)^*$ and $\widetilde{a \cup b} = \widetilde{a^*; b} = \mathbf{0}$. However, there is some difference between $\widetilde{a}$ and $a;a \cup a;b$ and between $a \cup b$ and $a^*; b$. $a$ is simpler than $a;a \cup a;b$ in a

natural sense and there is no simpler action $\gamma$ than $a$ s.t. $\widetilde{\gamma} = \widetilde{a}$. $a \cup b$ is simpler than $a^*;b$ and there is no simpler action than it sharing the same refraining condition. This means that the obligation to do $a;a \cup a;b$ is not *concise* but the obligation to do $a$ is. This is also the case for the obligations to do $a^*;b$ and $a \cup b$. In what follows, we formalize this type of conciseness and show that for every $\alpha$, there is a concise $\beta$ having the same refraining condition as $\alpha$. It is implied that if we consider only obligations, we do not have to consider all the actions and only the concise actions are enough.

In the last section, for a set $X$ of seqs, we use $X^T$, called the tree generated from $X$, to denote $\{\tau \mid \tau \approx \sigma \textit{ for some } \sigma \in X\}$. Let $\Delta$ and $\Theta$ be two sets of seqs. We say that $\Delta$ and $\Theta$ are $y$-equivalent if $\Delta^T = \Theta^T$.[4] For example, if $\Pi_0 = \{a, b\}$, $\Delta = \{a\}$ and $\Theta = \{aa, ab\}$, then $\Delta$ and $\Theta$ are $y$-equivalent. Note $\widetilde{\Delta} = \overline{\Delta^T}$. For every $\alpha$ and $\beta$, if $CS(\alpha)$ and $CS(\beta)$ are $y$-equivalent, then refraining from $\alpha$ is the same as refraining from $\beta$. Here are two basic facts about trees, which will be used later:

PROPOSITION 4.1

1. $\Delta^T = \bigcup_{\sigma \in \Delta} \{\sigma\}^T$.
2. If $\Delta^T = \Theta^T$, then for every $\sigma \in \Delta$, there is a $\tau \in \Theta$ s.t. $\sigma \approx \tau$, and for every $\tau \in \Theta$, there is a $\sigma \in \Delta$ s.t. $\sigma \approx \tau$.

Note that for all seqs $\sigma$ and $\tau$, if $\tau \sqsubseteq \sigma$, then $\{\sigma\}^T \subseteq \{\tau\}^T$.

DEFINITION 4.2 (Concise actions)
A set $\Delta$ of seqs is concise if for no $\sigma \in \Delta$ there is a $\sigma' \sqsubset \sigma$ s.t. $\{\sigma'\}^T \subseteq \Delta^T$. An action $\alpha$ is concise if $CS(\alpha)$ is concise.

For example, $\{a\}$ is concise but $\{aa, ab\}$ is not, as $a \sqsubset aa$ and $\{a\}^T = \{aa, ab\}^T = a\{a, b\}^*$. The notion of conciseness can be understood in the following way. Assume that $\Delta$ contains a seq $\sigma$ s.t. there is a $\sigma' \sqsubset \sigma$ s.t. $\{\sigma'\}^T \subseteq \Delta^T$. By replacing $\sigma$ by $\sigma'$ in $\Delta$, we get $\Delta'$. By Proposition 4.1, $\Delta^T = \Delta'^T$. $\Delta'$ is simpler than $\Delta$ in the sense that $\sigma'$ is shorter than $\sigma$.

PROPOSITION 4.3
For every set $\Delta$ of seqs, there is a unique set $\Theta$ of seqs which is concise and $y$-equivalent to $\Delta$.

PROOF. Define a set $\Theta$ of seqs as follows: for every $\sigma$, $\sigma \in \Theta$ iff (i) $\sigma \sqsubseteq \sigma'$ for some $\sigma' \in \Delta$, (ii) $\{\sigma\}^T \subseteq \Delta^T$ and (iii) there is no $\sigma'' \sqsubset \sigma$ s.t. $\{\sigma''\}^T \subseteq \Delta^T$. We claim that $\Theta$ is what we are looking for.

Firstly, it can be seen that $\Theta$ is concise.

Secondly, we show $\Theta^T = \Delta^T$. By the first item of Proposition 4.1, $\Theta^T = \bigcup_{\sigma \in \Theta}\{\sigma\}^T$. For every $\sigma \in \Theta$, $\{\sigma\}^T \subseteq \Delta^T$, therefore, $\Theta^T \subseteq \Delta^T$. Let $\tau \in \Delta^T$. Then there is a $\sigma \in \Delta$ s.t. $\tau \in \{\sigma\}^T$. Assume that there is no $\sigma' \sqsubset \sigma$ s.t. $\{\sigma'\}^T \subseteq \Delta^T$. Then $\sigma \in \Theta$ and $\{\sigma\}^T \subseteq \Theta^T$. Then $\tau \in \Theta^T$. Assume that there is a $\sigma' \sqsubset \sigma$ s.t. $\{\sigma'\}^T \subseteq \Delta^T$. Then there is a $\sigma''$ s.t. $\sigma'' \sqsubset \sigma$ and $\sigma'' \in \Theta$. As $\{\sigma\}^T \subseteq \{\sigma''\}^T$, $\tau \in \{\sigma''\}^T$. Then $\tau \in \Theta^T$. This implies $\Delta^T \subseteq \Theta^T$.

Thirdly, we show the uniqueness of $\Theta$. Let $X$ be a concise set of seqs s.t. $X^T = \Delta^T$. We want to show $X = \Theta$. As $\Theta^T = \Delta^T$, $X^T = \Theta^T$. Let $\sigma \in \Theta$. By the second item of Proposition 4.1, there is a $\tau \in X$ s.t. $\sigma \sqsubseteq \tau$ or $\tau \sqsubseteq \sigma$. Assume $\sigma \sqsubset \tau$. As $\sigma \in \Theta$, $\{\sigma\}^T \subseteq \Theta^T$. Then $\{\sigma\}^T \subseteq X^T$. Then $X$

[4] $y$ does not have any meaning and is used just to create a name.

is not concise and we get a contradiction. Then $\sigma \not\sqsubset \tau$. In a similar way, we can get $\tau \not\sqsubset \sigma$. Then $\sigma = \tau$. Then $\sigma \in X$. This implies $\Theta \subseteq X$. Similarly, we know $X \subseteq \Theta$. □

Such a $\Theta$ is called the *core* of $\Delta$, denoted as $\Delta^c$. For example, if $\Delta = \{aa, ab\}$, then $\Delta^c = \{a\}$.

LEMMA 4.4

1. $\Delta^c = \Theta^c$ if $\Delta^T = \Theta^T$
2. $\widetilde{\Delta^c} = \widetilde{\Delta}$.

Cores behave well:

PROPOSITION 4.5

1. $\Delta^c = \widetilde{\widetilde{\Delta^c}}^c$
2. $\widetilde{\Delta^c \cup \Theta^c} = \widetilde{\Delta^c} \cap \widetilde{\Theta^c}$
3. $\widetilde{\Delta^c; \Theta^c} = \widetilde{\Delta^c} \cup \left(\Delta^c; \widetilde{\Theta^c}\right)$ where $\Theta \neq \varnothing$.

PROOF.

1. By the first item of Lemma 4.4, it suffices to show $\Delta^T = \widetilde{\widetilde{\Delta^c}}^T$. By the second item of Lemma 4.4, $\widetilde{\Delta^c} = \widetilde{\Delta}$. Then $\widetilde{\widetilde{\Delta^c}} = \widetilde{\widetilde{\Delta}}$. Then it suffices to show $\Delta^T = \widetilde{\widetilde{\Delta}}^T$. By Proposition 3.3, $\Delta \subseteq \widetilde{\widetilde{\Delta}}$. Then $\Delta^T \subseteq \widetilde{\widetilde{\Delta}}^T$. Let $\sigma \in \widetilde{\widetilde{\Delta}}^T$. Then $\sigma \approx \tau$ for some $\tau \in \widetilde{\widetilde{\Delta}}$. Assume $\sigma \notin \Delta^T$. Then $\sigma \in \overline{\Delta^T} = \widetilde{\Delta}$. Then $\tau \not\approx \sigma$. We get a contradiction. Then $\sigma \in \Delta^T$. This means $\widetilde{\widetilde{\Delta}}^T \subseteq \Delta^T$.
2. This is a special case of the third item in Proposition 3.3.
3. By the fifth item of Proposition 3.3, $\widetilde{\Delta^c; \Theta^c} \subseteq \widetilde{\Delta^c} \cup \left(\Delta^c; \widetilde{\Theta^c}\right)$. Let $\sigma \in \widetilde{\Delta^c}$. Then $\sigma \not\approx \delta$ for every $\delta \in \Delta^c$. Assume $\sigma \notin \widetilde{\Delta^c; \Theta^c}$. Then there is a $\delta \in \Delta^c$ and a $\theta \in \Theta^c$ s.t. $\sigma \approx \delta\theta$. Then $\sigma \approx \delta$. This is impossible. Let $\sigma \in \Delta^c; \widetilde{\Theta^c}$. Then there is a $\delta \in \Delta^c$ and a $\tau \in \widetilde{\Theta^c}$ s.t. $\sigma = \delta\tau$. Assume $\sigma \notin \widetilde{\Delta^c; \Theta^c}$. Then there is an $x \in \Delta^c$ and a $y \in \Theta^c$ s.t. $\delta\tau \approx xy$. Then $\delta\tau \approx x$ and $\delta \approx x$. Then $\delta \sqsubseteq x$ or $x \sqsubseteq \delta$. As both $\delta$ and $x$ are in $\Delta^c$, neither $\delta \sqsubset x$ nor $x \sqsubset \delta$. Then $\delta = x$ and $\tau \approx y$. As $\tau \in \widetilde{\Theta^c}$ and $y \in \Theta^c$, we get a contradiction. Then $\sigma \in \widetilde{\Delta^c; \Theta^c}$. Then $\widetilde{\Delta^c} \cup \left(\Delta^c; \widetilde{\Theta^c}\right) \subseteq \widetilde{\Delta^c; \Theta^c}$. □

LEMMA 4.6
Let $\Delta$ be a set of seqs. Let $\min\left(\widetilde{\Delta}\right) = \left\{\sigma \in \widetilde{\Delta} \mid \textit{there is no } \sigma' \in \widetilde{\Delta} \textit{ s.t. } \sigma' \sqsubset \sigma\right\}$. Then $\left(\widetilde{\Delta}\right)^c = \min\left(\widetilde{\Delta}\right)$.

PROOF. Let $\sigma \in \left(\widetilde{\Delta}\right)^c$. Assume $\sigma \notin \widetilde{\Delta}$. Then there is a $\delta \in \Delta$ s.t. $\sigma \approx \delta$. Then $\delta \in \left\{\sigma\right\}^T$. By the first item of Proposition 4.1, $\left\{\sigma\right\}^T \subseteq \left(\widetilde{\Delta}\right)^{cT}$. As $\left(\widetilde{\Delta}\right)^{cT} = \widetilde{\Delta}^T$, $\left\{\sigma\right\}^T \subseteq \widetilde{\Delta}^T$. Then $\delta \in \widetilde{\Delta}^T$. Then there is a $\tau \in \widetilde{\Delta}$ s.t. $\delta \approx \tau$. This is impossible, as $\delta \in \Delta$. Then $\sigma \in \widetilde{\Delta}$. Assume $\sigma \notin \min\left(\widetilde{\Delta}\right)$. Then there is a $\sigma' \in \widetilde{\Delta}$ s.t. $\sigma' \sqsubset \sigma$. Since $\left\{\sigma'\right\}^T \subseteq \widetilde{\Delta}^T$, $\sigma \notin \left(\widetilde{\Delta}\right)^c$. We get a contradiction. Then $\sigma \in \min\left(\widetilde{\Delta}\right)$.

Let $\sigma \in \min\left(\widetilde{\Delta}\right)$. Then $\sigma \in \widetilde{\Delta}$. Assume $\sigma \notin \left(\widetilde{\Delta}\right)^c$. Then there is a $\sigma' \sqsubset \sigma$ s.t. $\left\{\sigma'\right\}^T \subseteq \widetilde{\Delta}^T$. Then $\sigma' \notin \widetilde{\Delta}$, as $\sigma \in \min\left(\widetilde{\Delta}\right)$. Then $\sigma' \approx \delta$ for some $\delta \in \Delta$. Then $\delta \in \left\{\sigma'\right\}^T$. Then $\delta \in \widetilde{\Delta}^T$. Then $\delta \approx \theta$ for some $\theta \in \widetilde{\Delta}$. This is impossible, as $\delta \in \Delta$. □

PROPOSITION 4.7
For every $\alpha \in \Pi_{\mathsf{PDL}}$, there is a $\beta \in \Pi_{\mathsf{PDL}}$ s.t. $CS(\beta) = CS(\alpha)^c$.

PROOF. Define regular languages as in the proof of Proposition 3.4. It suffices to show that $CS(\alpha)^c$ is a regular language. By the proof of the first item of Proposition 4.5, $CS(\alpha)^c = \widetilde{\widetilde{CS(\alpha)}}^c$. By Lemma 4.6, $\widetilde{\widetilde{CS(\alpha)}}^c = min(\widetilde{\widetilde{CS(\alpha)}})$. Then it suffices to show that $min(\widetilde{\widetilde{CS(\alpha)}})$ is a regular language. As $CS(\alpha)$ is a regular language, $\widetilde{CS(\alpha)}$ is a regular language by Proposition 3.4. Then $\widetilde{\widetilde{CS(\alpha)}}$ is a regular language as well. By a result in [9], $min(\widetilde{\widetilde{CS(\alpha)}})$ is a regular language. □

Such a $\beta$ is called the *core* of $\alpha$, denoted by $\alpha^c$.

Proposition 3.3 states a few typical properties of the function of opposite. The fourth and fifth are not good enough: the converses of the implications do not hold. Consequently, $\widetilde{\widetilde{\alpha}}$ is not equivalent to $\alpha$ and $\widetilde{\alpha;\beta}$ not equivalent to $\widetilde{\alpha}\cup\alpha;\widetilde{\beta}$. Considering that refraining is some type of negation, one might wonder about this. In fact, when restricted to concise actions, we can have a notion of refraining which behaves well. Let $\Pi^c_{\mathsf{PDL}}$ denote the set of concise actions of $\Pi_{\mathsf{PDL}}$. For every $\alpha \in \Pi^c_{\mathsf{PDL}}$, define $\widehat{\alpha} = \widetilde{\alpha}^c$. By Proposition 4.5, what follows is the case: $\widehat{\widehat{\alpha}} \equiv \alpha$ and $\widehat{\alpha;\beta} \equiv \widehat{\alpha} \cup \alpha;\widehat{\beta}$ where $\beta$ is a nonempty action.

## 5 Deontic logic based on a process logic

PDL handles the input/output behaviour of *terminating* actions well and CTL* is useful for reasoning about *progressive* behaviour of nonterminating actions. Nishimura [14] proposes a process logic which is a natural mix of PDL and CTL* and can deal with the progressive behaviour of terminating actions. We present a deontic logic based on a variant of this process logic.

### *5.1 Syntax*

Recall that $\Pi_{\mathsf{PDL}}$ is a set of actions and $\Phi_0$ a countable set of atomic propositions. Let $\alpha$ range over $\Pi_{\mathsf{PDL}}$ and $p$ over $\Phi_0$. Define a set $\Phi_{\mathsf{TDDL}}$ of formulas as follows:

$$\phi ::= p \mid \top \mid \mathfrak{b} \mid \neg\phi \mid (\phi \wedge \phi) \mid \mathrm{X}\phi \mid (\phi \mathrm{U} \phi) \mid \lceil\!\lfloor \alpha \rfloor\!\rceil \phi.$$

The reading of the featured formulas is as follows:

1. $\mathfrak{b}$: something *bad* will be done in the next moment.
2. $\mathrm{X}\phi$: $\phi$ will be the case in the next moment.
3. $(\phi \mathrm{U} \psi)$: $\phi$ will be the case until $\psi$.
4. $\lceil\!\lfloor \alpha \rfloor\!\rceil \phi$: no matter how the agent will perform $\alpha$, $\phi$ is the case *now*.

X and U are *temporal* operators and $\mathrm{X}\phi$ and $(\phi \mathrm{U} \psi)$ are temporal formulas. $\mathfrak{b}$ is a temporal propositional constant. $p$ and $\lceil\!\lfloor \alpha \rfloor\!\rceil \phi$ are *state* formulas. Later we will see that temporal formulas are essentially evaluated at states relative to paths and state formulas are evaluated at states not relative to any specific paths.

It seems weird to say that no matter how the agent will perform $\alpha$, $\phi$ is the case now. In fact, this is fine, as whether a temporal sentence is true or not now might be dependent on how the agent will act in the future. For example, whether a student will pass an exam is dependent on how he will study. In order to make a temporal sentence true now, the agent has to act in some way in the future.

The other routine propositional connectives and the falsity $\bot$ are defined in the usual way. Here are some special derivative expressions:

1. $\mathfrak{f} := \neg\mathfrak{b}$: something fine will be done in the next moment if there is a next moment.
2. $\mathrm{F}\phi := (\top \mathrm{U} \phi)$: $\phi$ will be the case.
3. $\mathrm{G}\phi := \neg\mathrm{F}\neg\phi$: $\phi$ will always be the case.
4. $\mathrm{D}\phi := \mathrm{F}(\neg \mathrm{X}\top \wedge \phi)$: $\phi$ will be the case at the end.
5. $(\!|\alpha|\!) := \neg\lceil\alpha\rfloor\neg\phi$: the agent has a way to perform $\alpha$ s.t. $\phi$ is the case now.
6. $[\alpha]\phi := \lceil\alpha\rfloor \mathrm{D}\phi$: no matter how the agent will perform $\alpha$, $\phi$ will be the case after $\alpha$ is done. This is the classical *box* modality.
7. $\langle\alpha\rangle\phi := (\!|\alpha|\!)\mathrm{D}\phi$: the agent has a way to perform $\alpha$ s.t. $\phi$ will be the case after $\alpha$ is done. This is the classical *diamond* modality.

$\mathfrak{f}$, $\mathrm{F}\phi$, $\mathrm{G}\phi$ and $\mathrm{D}\phi$ are temporal formulas, and $(\!|\alpha|\!)\phi$, $[\alpha]\phi$ and $\langle\alpha\rangle\phi$ are state formulas.

### *5.2 Semantics*

$\mathfrak{M}, w_0 \ldots w_n \Vdash \phi$, the formula $\phi$ being true at the state $w_0$ relative to the path $w_0 \ldots w_n$ in the model $\mathfrak{M}$, is defined as follows:

1. $\mathfrak{M}, w_0 \ldots w_n \Vdash p \Leftrightarrow w_0 \in V(p)$
2. $\mathfrak{M}, w_0 \ldots w_n \Vdash \top$ always holds
3. $\mathfrak{M}, w_0 \ldots w_n \Vdash \mathfrak{b} \Leftrightarrow 0 < n$ and $(w_0, w_1) \in B$
4. $\mathfrak{M}, w_0 \ldots w_n \Vdash \neg\phi \Leftrightarrow$ not $\mathfrak{M}, w_0 \ldots w_n \Vdash \phi$
5. $\mathfrak{M}, w_0 \ldots w_n \Vdash (\phi \wedge \psi) \Leftrightarrow \mathfrak{M}, w_0 \ldots w_n \Vdash \phi$ and $\mathfrak{M}, w_0 \ldots w_n \Vdash \psi$
6. $\mathfrak{M}, w_0 \ldots w_n \Vdash \mathrm{X}\phi \Leftrightarrow 0 < n$ and $\mathfrak{M}, w_1 \ldots w_n \Vdash \phi$
7. $\mathfrak{M}, w_0 \ldots w_n \Vdash (\phi \mathrm{U} \psi) \Leftrightarrow$ there is an $i \leq n$ s.t. $\mathfrak{M}, w_i \ldots w_n \Vdash \psi$ and $\mathfrak{M}, w_j \ldots w_n \Vdash \phi$ for every $j < i$
8. $\mathfrak{M}, w_0 \ldots w_n \Vdash \lceil\alpha\rfloor\phi \Leftrightarrow$ for every path $u_0 \ldots u_m$ in $S_\alpha$ starting at $w_0$, $\mathfrak{M}, u_0 \ldots u_m \Vdash \phi$.

The path $w_0 \ldots w_n$ in $\mathfrak{M}, w_0 \ldots w_n$ represents that the present state is $w_0$ and it will evolve to $w_1$ in the next moment and so on. We see that the operator $\lceil\alpha\rfloor$ is a universal quantifier over the paths in $\alpha$ starting at a state. $\lceil\alpha\rfloor$ is called a path modality. It can be verified that what follows are the truth conditions of the derivative expressions:

1. $\mathfrak{M}, w_0 \ldots w_n \Vdash \mathfrak{f} \Leftrightarrow (w_0, w_1) \notin B$ if $0 < n$
2. $\mathfrak{M}, w_0 \ldots w_n \Vdash \mathrm{F}\phi \Leftrightarrow$ there is an $i \leq n$ s.t. $\mathfrak{M}, w_i \ldots w_n \Vdash \phi$
3. $\mathfrak{M}, w_0 \ldots w_n \Vdash \mathrm{G}\phi \Leftrightarrow$ for every $i \leq n$, $\mathfrak{M}, w_i \ldots w_n \Vdash \phi$
4. $\mathfrak{M}, w_0 \ldots w_n \Vdash \mathrm{D}\phi \Leftrightarrow \mathfrak{M}, w_n \Vdash \phi$
5. $\mathfrak{M}, w_0 \ldots w_n \Vdash (\!|\alpha|\!)\phi \Leftrightarrow$ there is a path $u_0 \ldots u_m$ in $S_\alpha$ starting at $w_0$ s.t. $\mathfrak{M}, u_0 \ldots u_m \Vdash \phi$
6. $\mathfrak{M}, w_0 \ldots w_n \Vdash [\alpha]\phi \Leftrightarrow$ for every path $u_0 \ldots u_m$ in $S_\alpha$ starting at $w_0$, $\mathfrak{M}, u_m \Vdash \phi$
7. $\mathfrak{M}, w_0 \ldots w_n \Vdash \langle\alpha\rangle\phi \Leftrightarrow$ there is a path $u_0 \ldots u_m$ in $S_\alpha$ starting at $w_0$ s.t. $\mathfrak{M}, u_m \Vdash \phi$.

Whether a state formula is true or not at a state relative to a path has nothing to do with that path; therefore, state formulas are essentially evaluated at states. However, without being relative to a specific path, we cannot say whether a temporal formula is true. Note that the propositional constant

$\mathfrak{b}$ is a genuine temporal formula. A formula $\phi$ is *valid* if for every model $\mathfrak{M}$ and path $w_0 \dots w_n$, $\mathfrak{M}, w_0 \dots w_n \Vdash \phi$. We use TDDL to denote the set of valid formulas.

We put aside the propositional constant $\mathfrak{b}$ temporarily. The logic TDDL is a merge of PDL and CTL*: the path quantifier **A** of CTL* is replaced by $\lceil\alpha\rfloor$ which refers to the action $\alpha$, and temporal formulas talk about what happens during the execution of $\alpha$, instead of nonterminating computations. TDDL is a *minimal* merge of PDL and CTL* in this sense: its models are just the models of PDL and CTL*; its dynamic factor is just the path modality $\lceil\alpha\rfloor$ and its temporal factor is just the temporal operators X and U. The main difference between TDDL and the process logic in [14] lies in that the latter interprets atomic actions as arbitrary sets of sequences. For a comparison between the process logic in [14] and other process logics, we refer to [7].

## 6 Normative properties of actions

TDDL is able to describe what happens during the performance of actions. The propositional constant $\mathfrak{b}$ says that a bad behaviour will be made. This makes it possible that TDDL can define some normative properties of actions according to whether something bad happens and how it happens during their performance.

The temporal formula G$\mathfrak{f}$ intuitively means that at every point in the future, if it is not the end, something fine will be done in the next moment. $\mathrm{F}(\mathrm{X}\top \wedge \mathfrak{f})$ indicates that at some point in the future something fine will be done in the next moment. Note that F$\mathfrak{f}$ is always true but $\mathrm{F}(\mathrm{X}\top \wedge \mathfrak{f})$ is not; so they are different. By combining $\lceil\alpha\rfloor$, $(\!|\alpha|\!)$, G$\mathfrak{f}$ and F(X$\top \wedge \mathfrak{f}$), we can have four notions of permission:

1. $\mathcal{P}^A\alpha := (\!|\alpha|\!)\mathrm{G}\mathfrak{f}$
2. $\mathcal{P}^B\alpha := \lceil\alpha\rfloor\mathrm{G}\mathfrak{f}$
3. $\mathcal{P}^C\alpha := (\!|\alpha|\!)\mathrm{F}(\mathrm{X}\top \wedge \mathfrak{f})$
4. $\mathcal{P}^D\alpha := \lceil\alpha\rfloor\mathrm{F}(\mathrm{X}\top \wedge \mathfrak{f})$.

By negating a permission we get a prohibition. A prohibition of refraining expresses an obligation. We can get four notions of prohibition and obligation:

1. $\mathcal{F}^X\alpha := \neg\mathcal{P}^X\alpha$
2. $\mathcal{O}^X\alpha := \mathcal{F}^X\widetilde{\alpha}$

where $X \in \{A, B, C, D\}$.

Fix a model. We say that a path $w_0 \dots w_n$ is *legal* if it contains no bad transition, and *evil* if it contains no fine transition. Trivially, $w$ is both legal and evil path. It can be seen that $w_0 \dots w_n$ is legal iff G$\mathfrak{f}$ is true at it, and evil iff $\mathrm{G}(\mathrm{X}\top \rightarrow \mathfrak{b})$ is true at it. The truth conditions of the four groups of normative formulas can be stated by use of legal and evil paths:

1. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{P}^A\alpha \Leftrightarrow$ some path in $S_\alpha$ starting at $w_0$ is legal.
2. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{P}^B\alpha \Leftrightarrow$ all the paths in $S_\alpha$ starting at $w_0$ are legal.
3. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{P}^C\alpha \Leftrightarrow$ not all the paths in $S_\alpha$ starting at $w_0$ are evil.
4. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{P}^D\alpha \Leftrightarrow$ no path in $S_\alpha$ starting at $w_0$ is evil.
5. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{O}^A\alpha \Leftrightarrow$ no path in $S_{\widetilde{\alpha}}$ starting at $w_0$ is legal.
6. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{O}^B\alpha \Leftrightarrow$ not all the paths in $S_{\widetilde{\alpha}}$ starting at $w_0$ are legal.
7. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{O}^C\alpha \Leftrightarrow$ all the paths in $S_{\widetilde{\alpha}}$ starting at $w_0$ are evil.
8. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathcal{O}^D\alpha \Leftrightarrow$ some path in $S_{\widetilde{\alpha}}$ starting at $w_0$ is evil.

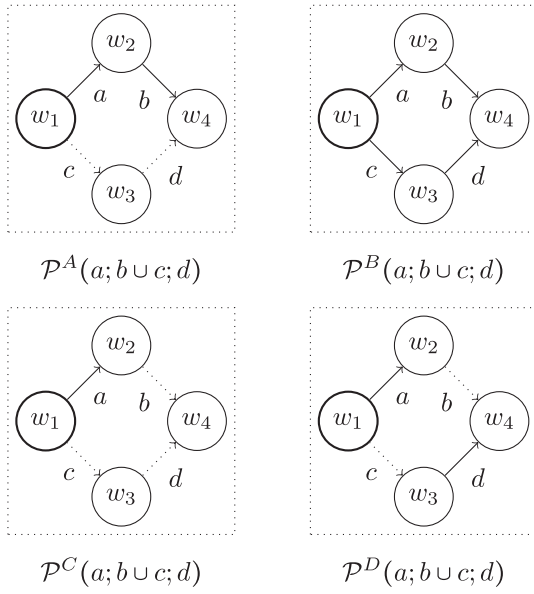


FIGURE 1. This figure shows the difference between the four notions of permission. Arrows represent transitions; solid ones represent fine transitions; dotted ones mean that it does not matter whether they are fine or not. The four formulas are true at the four $w_1$s, respectively.

$\mathcal{F}^X\alpha$ is just the negation of $\mathcal{P}^X\alpha$ and its semantics is simply omitted. All the normative formulas are state formulas. Figure 1 illustrates the difference among the four notions of permission. $\mathcal{P}^A\alpha$ is the *lack-of-prohibition* permission and $\mathcal{P}^B\alpha$ the *free-choice* permission. As far as we know, neither $\mathcal{P}^C\alpha$ nor $\mathcal{P}^D\alpha$ corresponds to a notion of permission in reality.

The following valid formulas tell some basic properties of the normative notions.

1. $\mathcal{P}^X\mathbf{1}$ where $X \in \{A, C\}$,
2. $\langle\alpha\rangle\top \to \left(\mathcal{O}^X\alpha \to \mathcal{P}^X\alpha\right)$ where $X \in \{A, C\}$.

Recall that $\mathbf{1}$ is the union of all atomic actions. The first formula indicates that there is always something allowed to do. The second formula means that if $\alpha$ is doable, then the obligation to do it implies the permission to do it.

Here are some features of the four notions of permission which involve action constructors:

1. $\mathcal{P}^A(\alpha \cup \beta) \leftrightarrow \left(\mathcal{P}^A\alpha \vee \mathcal{P}^A\beta\right)$
2. $\mathcal{P}^A(\alpha; \beta) \to \left(\mathcal{P}^A\alpha \wedge \langle\alpha\rangle\mathcal{P}^A\beta\right)$
3. $\mathcal{P}^B(\alpha \cup \beta) \leftrightarrow \left(\mathcal{P}^B\alpha \wedge \mathcal{P}^B\beta\right)$
4. $[\alpha]\langle\beta\rangle\top \to \left(\mathcal{P}^B(\alpha; \beta) \leftrightarrow \left(\mathcal{P}^B\alpha \wedge [\alpha]\mathcal{P}^B\beta\right)\right)$
5. $\mathcal{P}^C(\alpha \cup \beta) \leftrightarrow \left(\mathcal{P}^C\alpha \vee \mathcal{P}^C\beta\right)$
6. $[\alpha]\langle\beta\rangle\top \to \left(\mathcal{P}^C(\alpha; \beta) \leftrightarrow \left(\mathcal{P}^C\alpha \vee \langle\alpha\rangle\mathcal{P}^C\beta\right)\right)$
7. $\mathcal{P}^D(\alpha \cup \beta) \leftrightarrow \left(\mathcal{P}^D\alpha \wedge \mathcal{P}^D\beta\right)$
8. $\left(\mathcal{P}^D\alpha \vee [\alpha]\mathcal{P}^D\beta\right) \to \mathcal{P}^D(\alpha; \beta)$.

The third formula shows that $\mathcal{P}^B$ has the feature of free-choice permission.

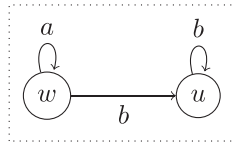


FIGURE 2. Assume that all the transitions are legal. Then the infinite legal path $w\ldots$ does not initially pass by any path in $S_{a^*;b}$.

What follows are some valid formulas concerning obligation:

1. $[\alpha]\langle\beta\rangle\top \rightarrow \left(\mathcal{O}^A(\alpha;\beta) \leftrightarrow \left(\mathcal{O}^A\alpha \wedge [\alpha]\mathcal{O}^A\beta\right)\right)$
2. $\mathcal{O}^B(\alpha \cup \beta) \leftrightarrow \left(\mathcal{O}^B\alpha \wedge \mathcal{O}^B\beta\right)$
3. $\mathcal{O}^B\alpha \rightarrow \mathcal{O}^B(\alpha;\beta)$
4. $\mathcal{O}^C(\alpha;\beta) \rightarrow \mathcal{O}^C\alpha$
5. $\mathcal{O}^D(\alpha \cup \beta) \leftrightarrow \left(\mathcal{O}^D\alpha \wedge \mathcal{O}^D\beta\right)$
6. $\mathcal{O}^D\alpha \rightarrow \mathcal{O}^D(\alpha;\beta)$.

Note that none of the converses of the implications hold.

In Section 1 we mention a few inferences which are useful to test whether a deontic logic in the dynamic approach considers intermediate states of performing actions.

1. Killing is prohibited; therefore, killing and then surrendering are also prohibited.
2. Smoking and then leaving are permitted; therefore, smoking is permitted.
3. Rescuing the injured and then calling an ambulance are obligatory; therefore, rescuing the injured is obligatory.

These inferences are valid w.r.t. the first group of normative notions. Note that the prohibition of killing does not imply the prohibition of surrendering after killing. Indeed, it can be verified that $\mathcal{F}^A k \wedge \langle k\rangle\mathcal{P}^A s$ is satisfiable where $k$ and $s$ represent killing and surrendering, respectively.

Here is some comparison between TDDL and the deontic logic proposed by [19], which is called Dynamic Logic of Permission (DLP). DLP interprets atomic actions as arbitrary sets of state sequences. TDDL is different from DLP at this point. DLP makes a distinction between fine and bad transitions. TDDL also does this. TDDL has temporal operators and can directly describe the process of performing actions. Deontic operators are defined in TDDL. DLP does not have any temporal operator. Instead, it has two primitive deontic operators $\Diamond$ and $\pi$ which are for lack-of-prohibition and free-choice permission, respectively. Generally speaking, we think that TDDL is more flexible than DLP in dealing with normative properties of actions.

As mentioned previously, [19] focuses on permission only and does not handle refraining from actions. However, at the end of it, a notion of obligation is briefly proposed. Actually, this notion has some connection with our first notion of obligation. Previously, paths are always finite state sequences. Assume that infinite paths are also allowed. We say that a legal path is *maximal* if it is not a proper initial segment of any legal paths. A maximal legal path *passes by* a finite path if the finite path is an initial segment of the maximal legal path. The notion of obligation that [19] proposes is as follows: *an action $\alpha$ is obligated at a state $w$ iff all maximal legal paths starting at $w$ initially pass by a path in the interpretation of $\alpha$*. This intuitively says that $\alpha$ is obligatory iff no matter how the agent will legally act, he will perform $\alpha$. It can be verified that the direction from right to left holds in TDDL. However, the other direction does not hold. Let $\alpha = a^*;b$. As $\widetilde{a^*;b} = \mathbf{0}$, trivially

$\mathfrak{M}, w \Vdash \mathcal{O}a^*; b$ for every $\mathfrak{M}$ and $w$. But it is not the case that for any $\mathfrak{M}$ and $w$, all maximal legal paths starting at $w$ initially pass by a path in $S_{a^*;b}$. A counter-example is illustrated by Figure 2. This example raises an interesting thing that is similar with the one raised by the example at the end of Section 3. Assume that the agent is at state $w$. There is no way for him to violate the obligation to perform $a^*; b$, but there is a way for him to avoid it. It seems that there is something between meeting an obligation and violating it, i.e. avoiding it.

## 7 Bisimulations

When defining models of TDDL, we put the constraint of pairwise disjointness on the interpretation of atomic actions. In fact, this constraint does not make any difference in logic. This section shows this result.

We say that a structure $\mathfrak{M} = (W, \{R_a \mid a \in \Pi_0\}, R_{\mathbf{0}}, R_{\mathbf{id}}, B, V)$ is a *pre-model* if it is like a model except that pairwise disjointness of atomic actions might not hold. Note that models are always pre-models but not vice versa.

DEFINITION 7.1 (P-bisimulations)
Let $\mathfrak{M} = \left(W, \left\{R_a \mid a \in \Pi_0\right\}, R_{\mathbf{0}}, R_{\mathbf{id}}, B, V\right)$ and $\mathfrak{M}' = \left(W', \left\{R'_a \mid a \in \Pi_0\right\}, R'_{\mathbf{0}}, R'_{\mathbf{id}}, B', V'\right)$ be two pre-models. Let $\Delta_W$ be the set of paths of $\mathfrak{M}$ and $\Delta_{W'}$ the set of paths of $\mathfrak{M}'$. $\mathcal{Z} \subseteq \Delta_W \times \Delta_{W'}$ is a p-bisimulation between $\mathfrak{M}$ and $\mathfrak{M}'$ if the following conditions are met:

1. if $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_m$, then $n = m$.
2. if $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, then $w_i \dots w_n \mathcal{Z} w'_i \dots w'_n$ for every $i \leq n$.
3. if $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, then $w_0 \dots w_n$ and $w'_0 \dots w'_n$ satisfy the same atomic propositions.
4. if $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, then $w_0 \dots w_n$ satisfies $\mathfrak{b}$ iff $w'_0 \dots w'_n$ satisfies $\mathfrak{b}$.
5. if $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, $u_0 \dots u_m \in S_\alpha$ and $u_0 = w_0$, then there is $u'_0 \dots u'_m$ s.t. $u'_0 = w'_0$, $u'_0 \dots u'_m \in S'_\alpha$ and $u_0 \dots u_m \mathcal{Z} u'_0 \dots u'_m$.
6. if $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, $u'_0 \dots u'_m \in S'_\alpha$ and $u'_0 = w'_0$, then there is $u_0 \dots u_m$ s.t. $u_0 = w_0$, $u_0 \dots u_m \in S_\alpha$ and $u_0 \dots u_m \mathcal{Z} u'_0 \dots u'_m$.

Here 'p' is for 'path'.

PROPOSITION 7.2
Let $\mathfrak{M}$ and $\mathfrak{M}'$ be two pre-models and $\mathcal{Z}$ a p-bisimulation between them. If $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, then they satisfy the same formulas.

PROOF. We put an induction on $\phi$. We consider only a few key cases.

**Case** $\phi = \mathrm{X}\psi$. $\mathfrak{M}, w_0 \dots w_n \Vdash \mathrm{X}\psi \Leftrightarrow \mathfrak{M}, w_1 \dots w_n \Vdash \psi \Leftrightarrow \mathfrak{M}', w'_1 \dots w'_n \Vdash \psi \Leftrightarrow \mathfrak{M}', w'_0 \dots w'_n \Vdash \mathrm{X}\psi$.

**Case** $\phi = \psi \mathrm{U} \chi$. Assume $\mathfrak{M}, w_0 \dots w_n \Vdash \psi \mathrm{U} \chi$. Then there is an $i \leq n$ s.t. $\mathfrak{M}, w_i \dots w_n \Vdash \chi$ and $\mathfrak{M}, w_j \dots w_n \Vdash \psi$ for every $j < i$. By the inductive hypothesis, $\mathfrak{M}', w'_i \dots w'_n \Vdash \chi$ and $\mathfrak{M}', w'_j \dots w'_n \Vdash \psi$ for every $j < i$. Then $\mathfrak{M}', w'_0 \dots w'_n \Vdash \psi \mathrm{U} \chi$. The other direction is similar.

**Case** $\phi = \langle\!|\alpha|\!\rangle \psi$. Assume $\mathfrak{M}, w_0 \dots w_n \Vdash \langle\!|\alpha|\!\rangle \psi$. Then there is $u_0 \dots u_m$ in $S_\alpha$ starting at $w_0$ s.t. $\mathfrak{M}, u_0 \dots u_m \Vdash \psi$. By the definition of p-bisimulations, there is $u'_0 \dots u'_m$ s.t. $u'_0 = w'_0$, $u'_0 \dots u'_m \in S'_\alpha$ and $u_0 \dots u_m \mathcal{Z} u'_0 \dots u'_m$. By the inductive hypothesis, $\mathfrak{M}'$, $u'_0 \dots u'_m \Vdash \psi$. Then $\mathfrak{M}', w'_0 \dots w'_n \Vdash \langle\!|\alpha|\!\rangle \psi$. The other direction is similar. □

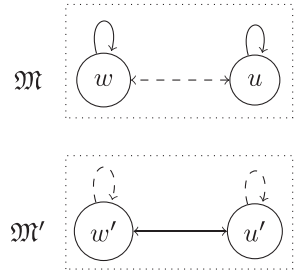


FIGURE 3. This figure shows that the respect requirement for c-bisimulations to imply equivalence cannot be replaced by the forth and back conditions on bad transitions. Solid arrows indicate fine transitions, dashed arrows bad transitions. All other information of the two models are identical. $W \times W'$ is a c-bisimulation meeting the forth and back conditions for bad transitions. However, $\mathfrak{M}, wu \Vdash \mathfrak{b}$ and not $\mathfrak{M}', w'u' \Vdash \mathfrak{b}$.

DEFINITION 7.3 (C-bisimulations)
Let $\mathfrak{M} = \left(W, \left\{R_a \,\middle|\, a \in \Pi_0\right\}, R_{\mathbf{0}}, R_{\mathbf{id}}, B, V\right)$ and $\mathfrak{M}' = \left(W', \left\{R'_a \,|\, a \in \Pi_0\right\}, R'_{\mathbf{0}}, R'_{\mathbf{id}}, B', V'\right)$ be two pre-models. $Z \subseteq W \times W'$ is a c-bisimulation between $\mathfrak{M}$ and $\mathfrak{M}'$ if the following conditions are met:

1. if $wZw'$, then they satisfy the same atomic propositions.
2. if $wZw'$ and $R_awu$, then there is $u'$ s.t. $R'_aw'u'$ and $uZu'$.
3. if $wZw'$ and $R'_aw'u'$, then there is $u$ s.t. $R_awu$ and $uZu'$.

Here 'c' is for "classical". We say that a c-bisimulation $Z$ *respects* $\mathfrak{b}$ if what follows is the case: if $wZw'$ and $uZu'$, then $(w, u) \in B$ iff $\left(w', u'\right) \in B'$. Each c-bisimulation respecting $\mathfrak{b}$ induces a p-bisimulation, as shown by the following proposition.

PROPOSITION 7.4
Let $\mathfrak{M}$ and $\mathfrak{M}'$ be two pre-models and $Z$ a c-bisimulation between them which respects $\mathfrak{b}$. Let $\Delta_W$ and $\Delta_{W'}$ be as above. Define $\mathcal{Z} \subseteq \Delta_W \times \Delta_{W'}$ as follows: $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$ iff $w_i Z w'_i$ for every $i \leq n$. Then $\mathcal{Z}$ is a p-bisimulation between $\mathfrak{M}$ and $\mathfrak{M}'$.

PROOF. We only verify that the fifth condition in Definition 7.1 is satisfied. We put an induction on $\alpha$. The **case** $\alpha = a$ is trivial.

**Case** $\alpha = \beta; \gamma$. Assume $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, $u_0 \dots u_m \in S_{\beta;\gamma}$ and $u_0 = w_0$. Then there is a $k \leq m$ s.t. $u_0 \dots u_k \in S_\beta$ and $u_k \dots u_m \in S_\gamma$. By the inductive hypothesis, there is $u'_0 \dots u'_k$ s.t. $u'_0 = w'_0$, $u'_0 \dots u'_k \in S'_\beta$ and $u_0 \dots u_k \mathcal{Z} u'_0 \dots u'_k$. By the definition of $\mathcal{Z}$, we know $u_k \mathcal{Z} u'_k$. By the inductive hypothesis again, there is $u'_k \dots u'_m$ s.t. $u'_k \dots u'_m \in S'_\gamma$ and $u_k \dots u_m \mathcal{Z} u'_k \dots u'_m$. Then $u'_0 \dots u'_m \in S'_{\beta;\gamma}$. By the definition of $\mathcal{Z}$, $u_0 \dots u_m \mathcal{Z} u'_0 \dots u'_m$.

**Case** $\alpha = \beta \cup \gamma$. Assume $w_0 \dots w_n \mathcal{Z} w'_0 \dots w'_n$, $u_0 \dots u_m \in S_{\beta\cup\gamma}$ and $u_0 = w_0$. Then $u_0 \dots u_m \in S_\beta$ or $u_0 \dots u_m \in S_\gamma$. Assume the former. By the inductive hypothesis, there is $u'_0 \dots u'_m$ s.t. $u'_0 = w'_0$, $u'_0 \dots u'_m \in S'_\beta$ and $u_0 \dots u_m \mathcal{Z} u'_0 \dots u'_m$. Then $u'_0 \dots u'_m \in S'_{\beta\cup\gamma}$. The arguments for the latter case is similar.

**Case** $\alpha = \beta^*$. The arguments for this case are the mix of the arguments for the cases $\alpha = \beta \cup \gamma$ and $\alpha = \beta; \gamma$. □

The *respect condition* for c-bisimulations is a strong requirement. Note that it cannot be replaced by the usual forth and back conditions on $B$: if $wZw'$ and $(w, u) \in B$, then there is a $u'$ s.t. $(w', u') \in B'$ and $uZu'$; if $wZw'$ and $(w', u') \in B'$, then there is a $u$ s.t. $(w, u) \in B$ and $uZu'$. Figure 3 provides a counter-example for this.

PROPOSITION 7.5
$\phi$ is satisfiable in a pre-model iff it is satisfiable in a model.

PROOF. The direction from right to left is trivial. To show the other direction, it suffices to show that for every pre-model $\mathfrak{M}$, there is a model $\mathfrak{M}'$ and a surjective function $Z$ from $\mathfrak{M}'$ to $\mathfrak{M}$ s.t. $Z$ is a c-bisimulation respecting $\mathfrak{b}$. This can be done by the so-called *copy method* presented by [5] to handle intersection of modalities. Ju and Hu [10] provide an improvement for this method. Here we sketch the arguments where the focus is that the copy method actually has room for handling bad transitions. For missing details, we refer to [10].

Let $\Pi_0 = \{a_1, \ldots, a_n\}$. Let $\mathfrak{M} = (W, \{R_a \mid a \in \Pi_0\}, R_{\mathbf{0}}, R_{\mathbf{id}}, B, V)$ be a pre-model. In *some* way, we can define $2^n$ structures $\mathfrak{M}^1 = (W^1, \{R^1_a \mid a \in \Pi_0\}, R^1_{\mathbf{0}}, R^1_{\mathbf{id}}, V^1)$, $\ldots$, $\mathfrak{M}^{2^n} = (W^{2^n}, \{R^{2^n}_a \mid a \in \Pi_0\}, R^{2^n}_{\mathbf{0}}, R^{2^n}_{\mathbf{id}}, V^{2^n})$ where $W^1, \ldots W^{2^n}$ are pairwise disjoint copies of $W$. In *some* way, we can merge these structures into a structure $\mathfrak{M}' = (W', \{R'_a \mid a \in \Pi_0\}, R'_{\mathbf{0}}, R'_{\mathbf{id}}, V')$ where atomic actions are pairwise disjoint and the *natural* function $Z$ from $W'$ to $W$ is a c-bisimulation. Define a set $B'$ on $W'$ in the following way: $(s, t) \in B'$ iff $(Z(s), Z(t)) \in B$. Then $B'$ is normatively serial and $Z$ respects $\mathfrak{b}$. □

## 8 Connections and future work

*Theoretical aspects*

The process logic PL presented by [7] strictly subsumes TDDL in expressivity if we ignore the propositional constant $\mathfrak{b}$. Harel, Kozen and Parikh [7] show that PL is decidable and complete. However, PL has an unnatural temporal operator $f$, called the first-moment operator, which is not definable by other operators. TDDL is a natural merge of PDL and CTL$^*$ and we think that it is still meaningful to find a decision procedure and a complete axiomatization for it.

*Another way to formalize refraining*

In order to handle to do something else, we put some constraints on actions: in syntax, there are only finitely many atomic actions and there is a special action **0**; in semantics, atomic actions are pairwise disjoint. These constraints let $\Pi_{\mathsf{PDL}}$ express to do something else. This is an implicit way. There is a different way to handle to do something else, i.e. explicitly introducing an action constructor for it.

Actions can be defined as before except that there is a new generation rule now: $\widetilde{\alpha}$ is an action if $\alpha$ is. To do $\widetilde{\alpha}$ means to do something else than $\alpha$. Models and paths are defined as before. Let $\mathcal{T}$ denote the set of paths. Definition 3.2 specifies an opposite function $\widetilde{\ }$ on the set of sets of computation sequences. Define it on the set of sets of paths in a similar way: for every set $\Delta$ of paths, let $\widetilde{\Delta} = \{\tau \in \mathcal{T} \mid \tau \not\sqsubseteq \sigma \ \&\ \sigma \not\sqsubseteq \tau \text{ for any } \sigma \in \Delta\}$. Actions are interpreted as sets of paths as before except that $\widetilde{\alpha}$ is interpreted as $\widetilde{S_\alpha}$ where $S_\alpha$ is the interpretation of $\alpha$. This way of dealing with to do something else follows a similar idea with the previous way. Many results and discussions made before, especially Proposition 3.5, can be easily transplanted here. Detailed comparison of the two ways is needed.

*The CTL version*

CTL is a restricted version of CTL* that is widely studied and applied. In CTL, temporal operators have to be immediately preceded by path quantifiers. CTL has only state formulas and the temporal formulas of CTL* are not well formed in it. In a similar way of how CTL* and PDL are mixed in this work, we can get a natural mix of PDL and CTL. By introducing to this mixed logic a propositional constant saying that this is a bad state, we can get a deontic logic. This logic might have better computational properties than TDDL. It deserves a close look.

*Betterness among actions*

We in this work make a black and white distinction among transitions: bad and fine ones. It is attractive to introduce a more fine-grained *betterness* relation among transitions and make connections with preference-based deontic logics such as [18].

*Legal relations*

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 PDL is given in [20]. 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.

## Acknowledgements

F. Ju was supported by the National Social Science Foundation of China (No. 12CZX053), the Fundamental Research Funds for the Central Universities (No. SKZZY201304) and China Scholarship Council. We would like to thank the anonymous referees for their useful comments and suggestions.

## References

[1] A. R. Anderson. Some nasty problems in the formal logic of ethics. *Noûs*, **1**, 345–360, 1967.
[2] J. Broersen. Action negation and alternative reductions for dynamic deontic logics. *Journal of Applied Logic*, **2**, 153–168, 2004.
[3] E. Emerson and J. Halpern. “Sometimes” and “not never” revisited: on branching versus linear time temporal logic. *Journal of the ACM*, **33**, 151–178, 1986.
[4] M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, **18**, 194–211, 1979.
[5] G. Gargov and S. Passy. A note on boolean modal logic. In *Mathematical Logic*, P. P. Petkov ed., pp. 311–321. 1990.
[6] P. T. Geach. Whatever happened to deontic logic? *Philosophia*, **11**, 1–12, 1982.
[7] D. Harel, D. Kozen and R. Parikh. Process logic: expressiveness, decidability, completeness. *Journal of Computer and System Sciences*, **25**, 144–170, 1982.
[8] R. Hilpinen. Deontic logic. In *The Blackwell Guide to Philosophical Logic*, L. Goble ed., pp. 159–182. Blackwell Publishing, 2001.

[9] J. Hopcroft, R. Motwani and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Pearson, 2006.
[10] F. Ju and X. Hu. On axiomatization of boolean modalities. *Studies in Logic*, **8**, 23–36, 2015.
[11] S. Kanger. New foundations for ethical theory. In *Deontic Logic: Introductory and Systematic Readings*, R. Hilpinen ed., pp. 36–58. Netherlands: Springer, 1971.
[12] P. McNamara. Deontic logic. In *The Stanford Encyclopedia of Philosophy*, E. N. Zalta ed., winter 2014 edn. 2014.
[13] J.-J. C. Meyer. A different approach to deontic logic: deontic logic viewed as a variant of dynamic logic. *Notre Dame Journal of Formal Logic*, **29**, 109–136, 1988.
[14] H. Nishimura. Descriptively complete process logic. *Acta Informatica*, **14**, 359–369, 1980.
[15] V. Pratt. Process logic. In *Proceedings of the 6th ACM Symposium on Principles of Programming Languages*, pp. 93–100. New York: ACM Press, 1979.
[16] R. Pucella and V. Weissman. Reasoning about dynamic policies. In *Proceedings of the 7th International Conference of Foundations of Software Science and Computation Structures*, I. Walukiewicz ed., pp. 453–467. Berlin: Springer-Verlag, 2004.
[17] X. Sun and H. Dong. Deontic logic based on a decidable PDL with action negation. 2015. Manuscript.
[18] J. van Benthem, D. Grossi and F. Liu. Deontics = betterness + priority. In *Proceedings of the 10th International Conference of Deontic Logic in Computer Science*, G. Governatori and G. Sartor, eds, pp. 50–65. Berlin: Springer-Verlag, 2010.
[19] R. van der Meyden. The dynamic logic of permission. *Journal of Logic and Computation*, **6**, 465–479, 1996.
[20] J. van Eijck and F. Ju. Modelling legal relations. 2017. Under submission.
[21] M. Vardi and P. Wolper. Yet another process logic. In *Logics of Programs*, E. Clarke and D. Kozen, eds, pp. 501–512. Vol. 164 of *Lecture Notes in Computer Science*, Berlin: Springer, 1984.
[22] H. Wansing. On the negation of action types: constructive concurrent PDL. In *Proceedings of the Twelfth International Congress of Logic, Methodology and Philosophy of Science*, P. Hájek, L. Valdes-Villanueva and D. Westerstahl, eds, pp. 207–225. London: King's College Publications, 2005.

Received 1 December 2016