F. Dignum, J.-J.Ch. Meyer, and R. Wieringa. Free choice and contextually permitted actions. *Studia Logica*, 57(1):193--220, 1996.

Free Choice and Contextually Permitted Actions

F.Dignum * J.-J.Ch.Meyer † R.J.Wieringa ‡

Abstract

We present a solution to the paradox of free choice permission by introducing strong and weak permission in a deontic logic of action. It is shown how counterintuitive consequences of strong permission can be avoided by limiting the contexts in which an action can be performed. This is done by introducing the *only* operator, which allows us to say that only α is performed (and nothing else), and by introducing contextual interpretation of action terms.

1 Introduction

In the standard system of deontic logic [Aqv84, Wri51], it is a theorem that

$$P(p) \rightarrow P(p \vee q)$$

which means that if p is permitted then $p \vee q$ is also permitted. As a consequence, we have that

$$P(\text{Talk to the president}) \rightarrow P(\text{Talk to the president} \vee \text{shoot the president}),$$

which is counterintuitive. In the literature, this is called the *paradox of free choice permission*. One way to resolve this paradox is to simply define *two* permission operators P_w and P_s, denoting weak and strong permissibility, that satisfy

$$P_w(p \vee q) \equiv P_w p \vee P_w q$$
$$P_s(p \vee q) \equiv P_s p \wedge P_s q.$$

This is done by e.g. Von Wright ([Wri68, page 22]). Although this avoids the paradox of free choice permission, it introduces another problem, for it now holds that $P_s p \rightarrow P_s(p \wedge q)$:

$$P_s p \equiv P_s((p \wedge q) \vee (p \wedge \neg q)) \equiv P_s(p \wedge q) \wedge P_s(p \wedge \neg q).$$

Kamp [Kam73] proposes to introduce a "focus" operator F, that keeps track of the disjuncts that are permitted in disjunctive permission. Thus Kamp allows expressions like $P_K(Fp \vee Fq)$ and $P_K F(p \vee q)$, where the permission operator P_K resembles Von Wright's P_s. The former expression entails $P_K Fp$, the latter does not. As Hilpinen [Hil81] remarks, F can be regarded

*Eindhoven University of Technology, Dept. of Mathematics and Computer Science, P.O.box 513, 5600 MB Eindhoven, The Netherlands, tel.+31-40-474426, fax. +31-40-436685, e-mail: dignum@win.tue.nl

†Utrecht University, Dept. of Computer Science, P.O.box 80085, 3508 TB Utrecht, The Netherlands, e-mail: jj@cs.ruu.nl; this author gratefully acknowledges the hospitality of Linköping University during revision of this paper.

‡Free University of Amsterdam, Faculty of Mathematics and Computer Science, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands, e-mail:roelw@cs.vu.nl This research of J.-J.Ch.Meyer and R.J.Wieringa is partially supported by ESPRIT BRWG project No.8319 'ModelAge'.

as an operator that selects the class of possible ways that are permitted in order to establish the result given as its argument.

Castañeda [Cas81] argues that the paradox of free choice permission, along with many other paradoxes of deontic logic, is due to the use of ordinary logical connectives inside the P operator. He proposes to distinguish *practitions* from *assertions* and to apply deontic operators to practitions rather than assertions. Interpreting this as a distinction between *actions* and *states*, McCarty [McC83], Khosla and Maibaum [KM87] and Meyer [Mey88] independently produced formalizations of deontic action logic, in which actions are distinguished from states and deontic operators apply to actions, not states. The action logic used by these authors is some variant of dynamic logic [Har79, KT90]. Meyer [Mey87, Mey88] showed how this approach can avoid many of the paradoxes of standard deontic logic. However, the paradox of free choice permission still remains, except that it now has the form

$$P(\alpha) \rightarrow P(\alpha + \beta),$$

where α and β denote actions and $+$ is a choice operator. It was suggested by Meyer and Wieringa [Mey92, WM93] that the paradox could be avoided by distinguishing *active* from *passive* choice, denoted by $\oplus$ and $+$, respectively. The action expression $\alpha + \beta$ means intuitively that α or β occurs, but it is not stated which, whereas $\alpha \oplus \beta$ means that an action occurs in which a choice between α and β is made. An action is said to be permitted in this approach if there is a way of executing it which does not lead to a violation. (This is a kind of Anderson reduction, explained in more detail below.) Given this intuitive meaning, the validities

$$P(\alpha) \vee P(\beta) \equiv P(\alpha + \beta) \text{ and}$$
$$P(\alpha) \wedge P(\beta) \equiv P(\alpha \oplus \beta)$$

are not counterintuitive. However, in this approach we also have the validity

$$P(\text{Shoot a gun \& Aim in the air}) \longrightarrow P(\text{Shoot a gun}),$$

where & is a *synchronous execution* operator, which says that its two arguments denote synchronously occurring actions. The reason is that if there is a way of executing α jointly with β that does not lead to a violation, then there is also a way of executing α that does not lead to a violation. We will refer to this problem as the *paradox of context-sensitive permission*. It can even be shown that the following holds:

$$P(\alpha\&\beta) \longrightarrow P((\alpha\&\beta)+(\alpha\&\overline{\beta})),$$

where $\overline{\beta}$ expresses the *negation* of an action expressed by β. Roughly, this is any event in which β does *not* occur; details on this follow later.

In this paper, we propose a resolution of the paradox of free choice permission which avoids the paradox of context-sensitive permission. The idea is simply to return to Von Wright's idea mentioned at the beginning, and distinguish a strong and a weak permission operator P_s and P_w. If α is weakly permitted, there is at least one way of doing it that does not lead to a violation. If it is strongly permitted, then no ways of doing it lead to a violation. Similar operators have been introduced by McCarty [McC83, McC86] and Segerberg [Seg82]. We will show that, by introducing contexts in which actions take place, the problem with strong permission mentioned at the beginning of this paper can be avoided.

Before we give a new definition for permission we will, in the next section, first discuss the exact meaning of action expressions in our formalism. We will introduce a new

operator on action expressions that will make it possible to specify that an action is performed in isolation. At the end of section 2 we will show how the action expressions are incorporated into a logical language that can be used to describe deontic properties. In section 3, we will discuss the definition for the permission operator and show how the Free Choice Paradox can be resolved while avoiding some other problems. In section 4, we discuss a possible refinement of the theory based on the previous sections, by introducing contexts of actions. Finally, in section 5, we draw some conclusions and point out some topics for future research.

2 Interpretation for actions in logic

2.1 Introduction

In [Mey88, DM90, Dig92] the standard interpretation of an action expression was given in an open sense, i.e. if an action expression is used it means informally that the action denoted by that action expression occurs, possibly in combination with other actions. The reason for this is twofold. First it is more natural. Usually we do not have complete knowledge of the world. In that case it is easier to be able to specify the action of which one is certain that it occurs, without having to say anything about other actions. The second reason is that it facilitates the definition of the parallel execution as well as the negation of an action, which is needed in this approach to model obligation. See [DM90] for more explanation on this topic.

We will keep this "standard" interpretation of an action expression, but we will add an operator to indicate that an action can only be performed in isolation. This operator is (quite obviously) called the "*only*" operator. The intuitive meaning of the action term $only(\alpha)$ is that the action denoted by α is performed and no other action is performed. Actually, $only(\alpha)$ denotes how the action denoted by α is usually viewed in computer science.

We now first define the formal language(s) that we shall use. First we give a definition of *action expressions*, which we shall typically denote α, possibly with subscripts. To this end we assume a set At of *atomic action expressions* that are typically denoted by $\underline{a}, \underline{b}, \ldots$. Finally, we assume special action expressions **any** and **fail** denoting "don't care what happens" and "failure", respectively.

Now we define:

Definition 1 The set $OAct$ of action expressions not containing the operator *only* is given as the smallest set closed under:

(i). $At \cup \{\mathbf{any}, \mathbf{fail}\} \subseteq OAct$

(ii). $\alpha_1, \alpha_2 \in OAct \Longrightarrow \alpha_1 + \alpha_2 \in OAct$

(iii). $\alpha_1, \alpha_2 \in OAct \Longrightarrow \alpha_1 \& \alpha_2 \in OAct$

(iv). $\alpha \in OAct \Longrightarrow \overline{\alpha} \in OAct$

The set Act of action expressions is given as the smallest set closed under:

(i). $OAct \cup \{only(\alpha) | \alpha \in OAct\} \subseteq Act$

(ii). $\alpha_1, \alpha_2 \in Act \Longrightarrow \alpha_1 + \alpha_2 \in Act$

(iii). $\alpha_1, \alpha_2 \in Act \Longrightarrow \alpha_1 \& \alpha_2 \in Act$

(iv). $\alpha \in Act \Longrightarrow \overline{\alpha} \in Act$

We further use the following terminology: atomic action expressions $\underline{a} \in Act$, as well as their negations $\overline{\underline{a}}$ ($\underline{a} \in At$) are called *action literals*. An action expression that is the conjunction (using the &-connective) of one or more action literals is called an *action term*. So, for example, $\underline{a}\&\overline{\underline{b}}\&\underline{c}$, for $\underline{a}, \underline{b}, \underline{c} \in At$, is an action term.

Note: We build the set of action expressions in two steps in order to prevent the nesting of the *only* operator.

In the rest of this section we will give the semantics of action expressions from Act that do not contain an occurrence of the *only* operator. The semantics of $only(\alpha)$ will be treated in the next section on the basis of the semantics developed in this section.
The semantics of action expressions is based on (sets of) so-called *events*. (In fact, we shall use even *sets of* sets of events, but we shall see this shortly.) The semantics that we shall give here is a simplified version of the one used in [Mey88]. The simplification is based on the fact that we do not consider sequences of actions in this paper. The extension to sequences of actions can be easily made, but would obscure the points that we try to make in this paper.

With every atomic action expression $\underline{a} \in Act$, we associate an event a in a given class $\mathcal{A}$ of events, with typical elements $a, b, c....$ Events are the semantical entities on which we shall base our interpretation of action expressions. They have an intensional flavour: they denote what happens in the world in an abstract way; only the event itself is recorded here, not yet its result. This is usually called a *uniform* semantics in the literature on the semantics of concurrency (e.g. [dBKM+86]). (Later on, we shall also introduce a more extensional (or non-uniform) semantics in the sense of events as causing changes (or transitions) of states of the world.) We further assume a special event δ, which is not an element of $\mathcal{A}$, called failure (comparable to deadlock in process algebra ([BW90]). The relation between an action expression $\underline{a} \in Act$ and the associated event $a \in \mathcal{A}$ is more involved than just interpreting $\underline{a}$ as a. We shall interpret atomic action expressions $\underline{a} \in Act$ in a more sophisticated way, which we call "open": the meaning of an atomic action expression $\underline{a} \in Act$ will be the event $a \in \mathcal{A}$ corresponding with it, in combination with any other subset of the events in $\mathcal{A}$. Thus $\underline{a}$ expresses (is a representation of the fact) that the (performance / occurrence of) event a is guaranteed, but also other events may happen / occur simultaneously. The latter implements the "open interpretation of an action expression" that we mentioned above. Of course, this will introduce a lot of "nondeterminism", since we have a great number of ways to let "other events happen / occur simultaneously". (Another source of nondeterminism is the fact that we have a choice operator for action expressions, so nondeterminism is already built into the language.)

So, to define the semantics formally, we need the notion of a *synchronicity set* (or *s-set* for short). A synchronicity set will denote a set of events that occur simultaneously. They are the basic building blocks of the semantics. Moreover, in order to treat the nondetermistic features of our language of actions (and the open interpretation) we shall consider *sets of* s-sets.

Definition 2

1. The set $\{\delta\}$ is a synchronicity set.
2. Every non-empty finite subset of $\mathcal{A}$ is an s-set.

Notation: In concrete cases we write the sets with square brackets, in order to distinguish them easily from other sets that we will use. So, the s-set consisting of δ is written as $[\delta]$ and

the s-set consisting of the events a and b is written as $\left[\begin{array}{c} a \\ b \end{array}\right]$. The powerset of non-empty finite subsets of $\mathcal{A}$ will be denoted by $\wp^+(\mathcal{A})$.

The above definition prevents the simultaneous execution of the special event δ with other events, because it is not in $\mathcal{A}$. This is necessary, because it is not possible to perform an event and at the same time have a deadlock.

Definition 3 The domain $\mathcal{D}$ for our model for action expressions from Act is the collection of sets of s-sets. I.e. $\mathcal{D}=\wp(\wp^+(\mathcal{A}) \cup \{[\delta]\})$.

Because the language of action expressions contains a choice operator, which introduces non-determinism, we have to consider sets of s-sets as the semantics of an action expression. Each of the s-sets in these sets stands for a possible choice. As we stated before, this non-determinism is also introduced by using the open specification of an action.

Notation: We use $T, T_1, \ldots, T', \ldots$ to denote sets of s-sets.

To give the denotation for all action expressions in Act we define the semantical counterparts of the syntactical operators $+$, $\&$ and negation. Before we give these definitions, we define a convenient operator on sets of s-sets:

Definition 4 Let T be a set of s-sets then

$$T^\delta = \begin{cases} T \backslash \{[\delta]\} & \text{if } \exists S \in T : S \neq [\delta] \\ \{[\delta]\} & \text{otherwise} \end{cases}$$

The operator T^δ is closely related to what is called "failure removal" in [dBKM+86]. The idea is that failure is avoided when possible, i.e. when there is a non-failing alternative. In [Bro86], this is called *angelic* nondeterminism.

With this function defined, we can now give the semantical operators on $\mathcal{D}$.
For the parallel operator $\&$ we use a set-intersection $\barwedge$, which is almost the same as the normal set-intersection.

Definition 5 For $T, T' \in \mathcal{D}$:

$$T \barwedge T' = \begin{cases} T \cap T' & \text{if } T \cap T' \neq \emptyset \\ \{[\delta]\} & \text{otherwise} \end{cases}$$

The semantical counterpart of the choice operator is defined as follows:

Definition 6 For $T, T' \in \mathcal{D}$:

$$T \boxdot T' = (T \cup T')^\delta$$

The above definition states that the choice between two sets of s-sets is the union of those two sets minus $[\delta]$, unless the union does not contain anything else.

The last definition defines the semantic counterpart of the negation (non-performance) of an action expression.

Definition 7 The definition of "$\sim$" is given as follows:

1. For an s-set S,
$$S^{\sim} = \begin{cases} \wp^+(\mathcal{A})\backslash\{S\} & \text{if } S \neq [\delta] \\ \wp^+(\mathcal{A}) & \text{if } S = [\delta] \end{cases}$$
2. For a non-empty set $T \in \mathcal{D}$
$$T^{\sim} = \bigcap_{S \in T} S^{\sim}$$

The idea of these definitions is the following: For an s-set ($S \neq [\delta]$) the negation just yields the set-theoretic complement of $\{S\}$ with respect to $\wp^+(\mathcal{A})$. For the negation of a set of s-sets T we take the intersection of the sets(!) of the negations of all the s-sets contained in T.

The semantics of action expressions in Act, apart from $only(\alpha)$, which is treated in the next section, are defined as follows:

Definition 8 The semantic function $[\![]\!] \in Act \rightarrow \mathcal{D}$ is given by:
$$[\![\underline{a}]\!] = \{S \in \wp^+(\mathcal{A}) | a \in S\}$$
$$[\![\alpha_1 + \alpha_2]\!] = [\![\alpha_1]\!] \cup [\![\alpha_2]\!]$$
$$[\![\alpha_1 \& \alpha_2]\!] = [\![\alpha_1]\!] \cap [\![\alpha_2]\!]$$
$$[\![\overline{\alpha}]\!] = [\![\alpha]\!]^{\sim}$$
$$[\![\mathbf{fail}]\!] = \{[\delta]\}$$
$$[\![\mathbf{any}]\!] = \wp^+(\mathcal{A})$$

The first clause of the above definition expresses that the meaning of the action expression $\underline{a}$ is exactly as we have described informally before: it is the set of s-sets that contain the event a, representing a choice between all (simultaneous) performances of sets of events which at least contain the event a, so that the performance of a is guaranteed but also other events may happen simultaneously. Here we recognize the open interpretation of action expressions as discussed earlier.
The meaning of the action expression **fail** is comparable to a deadlock. The only event that can be performed is δ. The action expression **any** is the complement of **fail**. It stands for a choice of any possible combination of events.

Finally we define the equality of action expressions and some kind of implication between action expressions in terms of their semantics.

Definition 9 We define $\alpha_1 =_{\mathcal{D}} \alpha_2$ iff $[\![\alpha_1]\!] = [\![\alpha_2]\!]$.
We define $\alpha_1 > \alpha_2$ iff $[\![\alpha_1]\!] \subseteq [\![\alpha_2]\!]$.

If $\alpha_1 =_{\mathcal{D}} \alpha_2$ then we say that α_1 and α_2 are intensionally equivalent.
If $\alpha_1 > \alpha_2$ then we say that α_1 intensionally *involves* α_2.

Defining the domain and the operators as we did, renders this into a boolean algebra, with **fail** as bottom element.

Proposition 1

1. $(\mathcal{D}, +, \&, ^{-}, \mathbf{fail})$ *is a boolean algebra.*
2. $\mathcal{D}$ *satisfies the following property concerning the special actions:*
$$\overline{\mathbf{fail}} =_{\mathcal{D}} \mathbf{any}$$

Proof: Clear by the proposition.□.

2.2 Only

In the semantics for $only(\alpha)$ we require that α be of a very strict format, which we will call strict disjunctive normal form. The definition of this format is given in two steps:

Definition 10 An action expression is in disjunctive normal form if

1. it is a finite disjunction (using the + operator) of one or more action terms and it does not contain any occurrence of **fail**, or
2. it consists of the action expression **fail**.

Definition 11 The action expression α is in strict disjunctive normal form if it is in disjunctive normal form and for every subexpression of α of the form $\alpha_1 + \alpha_2$ the following hold:

- neither $[\![\alpha_1]\!] \subseteq [\![\alpha_2]\!]$ nor $[\![\alpha_2]\!] \subseteq [\![\alpha_1]\!]$.
- neither $[\![\alpha_1]\!] = \{[\delta]\}$ nor $[\![\alpha_2]\!] = \{[\delta]\}$.

Note: The only action expression which is in strict disjunctive normal form and that contains an occurrence of **any** is the action expression **any** by itself!
The semantics of the *only* operator is only defined on action expressions that are in strict disjunctive normal form. The reason is the interaction between the choice operator and the *only* operator. Intuitively it is clear that it makes more sense to say "John only opens the window" than something like:"John only opens the window or closes the door". That is, the *only* operator works more naturally on atomic action expressions (and their conjunctions) than on disjunctions of action expressions. This is due to the fact that the *only* operator restricts the actions that are taking place, while the choice operator does the reverse. This is reflected in the formalization of $only(\underline{a} + \underline{b})$, which is informally defined to mean $only(\underline{a})$ or $only(\underline{b})$ or $only(\underline{a}\&\underline{b})$ (see definition 12).

Fortunately, in the following, we can use the property:

Theorem 1 For any action expression α there exists an action expression α^* such that α^* is in strict disjunctive normal form and $[\![\alpha]\!] = [\![\alpha^*]\!]$.

Proof: Follows direct from the semantics and the definitions. □
E.g. $[\![\underline{a} + (\underline{a}\&\underline{b})]\!] = [\![\underline{a}]\!]$. The righthand-side stands for "any action that includes a". Clearly, this includes the actions that contain a and b. Therefore it is clear that the equivalence above holds.
In the following we use α^* to denote the strict disjunctive normal form of α.
It should be noted that there is no unique α^* for each α.

Now, we define the semantics of the *only* operator in three steps:

Definition 12

1. Let $\underline{a}$ be an atomic action expression then
   - $[\![only(\underline{a})]\!] = \{[a]\}$
   - $[\![only(\overline{\underline{a}})]\!] = [\![\overline{\underline{a}}]\!]$
2. Let α and β be action terms then

– $[\![only(\alpha\&\beta)]\!] = \{S|S = S_1 \cup S_2 \; and \; S_1 \in [\![only(\alpha)]\!] \; and \; S_2 \in [\![only(\beta)]\!]\}$

3. Let α be an action expression in strict disjunctive normal form.

   – if α is an action literal or an action term then $[\![only(\alpha)]\!]$ is defined as above.
   – if $\alpha = \alpha_1 + \alpha_2$ then $[\![only(\alpha)]\!] = [\![only(\alpha_1)]\!] \cup [\![only(\alpha_2)]\!] \cup [\![only((\alpha_1\&\alpha_2)^*)]\!]$
   – $[\![only(\mathbf{any})]\!] = [\![\mathbf{any}]\!]$

Note: In the definition above it is stated that $[\![only(\overline{\underline{a}})]\!] = [\![\overline{\underline{a}}]\!]$ which means that the *only* operator has no effect on negated action expressions. This is due to the fact that these negated action expressions are already minimal in the following sense: It is not possible to leave out any synchronicity set from the semantics and still keep a meaningful definition of the negation.
Example: Let the set of events $\mathcal{A}$ be $\{a, b\}$ then

$$[\![only(\underline{a}\&\underline{b})]\!] = \{S|S = S_1 \cup S_2 \; and \; S_1 \in \{[a]\} \; and \; S_2 \in \{[b]\}\} = \{\begin{bmatrix} a \\ b \end{bmatrix}\}$$
$$[\![only(\underline{a} + \underline{b})]\!] = [\![only(\underline{a})]\!] \cup [\![only(\underline{b})]\!] \cup [\![only((\underline{a}\&\underline{b}))]\!] =$$
$$\{[a], [b], \begin{bmatrix} a \\ b \end{bmatrix}\}$$

Notation: In the rest of this paper, whenever we write $only(\alpha)$, we assume that α is in strict disjunctive normal form.

It is easy to see from the definition that the *only* operator does not distribute over the other operators. That is:

$$only(\alpha\&\beta) \neq_{\mathcal{D}} only(\alpha)\&only(\beta)$$
$$only(\alpha + \beta) \neq_{\mathcal{D}} only(\alpha) + only(\beta)$$
$$\overline{only(\alpha)} \neq_{\mathcal{D}} only(\overline{\alpha})$$

This can be seen with the following very small example where we take the set of all events $\mathcal{A}$ to be $\{a, b\}$.

$$[\![only(\underline{a}\&\underline{b})]\!] = \{\begin{bmatrix} a \\ b \end{bmatrix}\} \neq$$
$$\{[\delta]\} = \{[a]\} \bullet \{[b]\} = [\![only(\underline{a})\&only(\underline{b})]\!]$$
$$[\![only(\underline{a} + \underline{b})]\!] = \{[a], [b], \begin{bmatrix} a \\ b \end{bmatrix}\} \neq$$
$$\{[a], [b]\} = \{[a]\} \cup \{[b]\} = [\![only(\underline{a}) \cup only(\underline{b})]\!]$$
$$[\![only(\overline{\underline{a}})]\!] = [\![\overline{\underline{a}}]\!] = \{[b]\} \neq$$
$$\{[b], \begin{bmatrix} a \\ b \end{bmatrix}\} = \{[a]\}^{\sim} = [\![only(\underline{a})]\!]^{\sim} = [\![\overline{only(\underline{a})}]\!]$$

Because the *only* operator does not distribute over any of the other operators there are not many properties that can be proven for this operator. The only properties that are important to mention here are the following:

Proposition 2

$$[\![only(\alpha^*)]\!] \subseteq [\![\alpha]\!]$$

Proof: Easy by induction.□
This property means that $only(\alpha^*)$ does not lead to more possible states than α.

Proposition 3 *Let $\alpha \neq_{\mathcal{D}} \beta$ and let α and β both be action terms not containing any negative literal then*

$$only(\alpha)\&only(\beta) =_{\mathcal{D}} \textbf{fail}$$

Proof: Easy by induction.□
This proposition states that $only(\alpha)$ and $only(\beta)$, in general, cannot be performed in combination.

With the *only* operator defined, in the next section we will discuss the logical language in which the action expressions are used.

2.3 A logic about actions

We now proceed with the definition of a set of *formulas* with which we can describe the behaviour of (interpreted) actions. This language is a variant of *dynamic logic* ([Har79]), and was first used for this purpose in [Mey88].

We assume a fixed set *Prop* of atomic propositions, including a special element $Violation$, denoting (a state of) violation. The set *Form* of formulae is then the smallest set closed under:

(1). $Prop \subseteq Form$

(2). $\phi_1, \phi_2 \in Form \Longrightarrow \phi_1 \wedge \phi_2 \in Form$

(3). $\phi \in Form \Longrightarrow \neg\phi \in Form$

(4). $\alpha \in Act, \phi \in Form \Longrightarrow [\alpha]\phi \in Form$

Note: Other propositional connectives such as $\vee$ and $\rightarrow$ are assumed to be introduced as the usual abbreviations; moreover, $< \alpha > \phi$ is introduced as an abbreviation for $\neg[\alpha]\neg\phi$. Furthermore, it is convenient to use the abbreviation **false** for $\phi \wedge \neg\phi$, and **true** for $\neg$**false**. The informal meaning of $[\alpha]\phi$ is "doing α necessarily leads to a state where ϕ obtains; $< \alpha > \phi$ means that doing α possibly leads to a state where ϕ obtains.

Moreover, and very importantly for the purpose of this paper, deontic operators are introduced. If α is any action expression, then $F(\alpha)$, $O(\alpha)$ and $P(\alpha)$ are formulas with the senses: It is forbidden to perform the action named by α, it is obliged to perform the action named by α and it is permitted to perform the action named by α. The formulas are introduced in the set *Form* as abbreviations as follows:

- $F(\alpha)$ abbreviates $[\alpha]Violation$
- $O(\alpha)$ abbreviates $F(\overline{\alpha})$
- $P(\alpha)$ abbreviates $\neg F(\alpha)$

For an extensive account of the adequacy of these abbreviations we refer to [Mey88, Mey87, WWMD91]; here it suffices to note that e.g. the claim that it is forbidden to do the action denoted by α is equated with the dynamic logic expression $[\alpha]Violation$, stating that performing the action denoted by α leads to a violation state (i.e., a state in which a violation obtains).

Interpretation of formulas in *Form* is given by means of the notion of a Kripke-model $\mathcal{M} = (\Sigma, \pi, \{R_\alpha | \alpha \in Act\})$, where Σ is a set of states (worlds), π is a truth assignment function to the atomic propositions relative to a state: π is a function $\Sigma \to (Prop \to \{tt, ff\})$, where tt amd ff denote truth and falsehood, respectively. Thus, for $p \in Prop$, $\pi(\sigma)(p) = tt$ means that the atomic proposition p is true in state σ. The accessibility relations R_α ($\alpha \in Act$) specify how actions can change states. For this we need an extensional, state-based semantics of action expressions α.

As we have seen before, our basic semantic entities are events in A. So we start with postulating what effects they have in terms of state transformations (we do this relative to a set Σ of states): we have a function $eff_\Sigma : \mathcal{A} \to (\Sigma \to \Sigma)$, such that $eff_\Sigma(a)$ is a function from states to states. (Thus we assume events to be deterministic. If one wants, this can be easily modified to nondeterministic events, but for simplicity we will not do this here.)

Definition 13 An s-set $A = [a_1, ..., a_n] \subseteq \mathcal{A}$ is called *compatible* (on Σ) w.r.t. state σ if

$$(eff_\Sigma(a_1) \circ ... \circ eff_\Sigma(a_n))(\sigma) = (eff_\Sigma(a_{i_1}) \circ ... \circ eff_\Sigma(a_{i_n}))(\sigma)$$

for all permutations $(i_1, ..., i_n)$ of $(1, ..., n)$. (Here $\circ$ denotes function composition.)

Now we define the effect of an s-set $A = [a_1, ..., a_n]$ by:

- $eff_\Sigma(A)(\sigma) = \{(eff_\Sigma(a_1) \circ ... \circ eff_\Sigma(a_n))(\sigma)\}$ if A is compatible on Σ w.r.t. σ,
- $eff_\Sigma(A)(\sigma) = \emptyset$, otherwise

The function eff_Σ is lifted further to sets of s-sets in the usual way:

$$eff_\Sigma(T)(\sigma) = \bigcup_{A \in T} eff_\Sigma(A)(\sigma)$$

for $T \subseteq \wp^+(\mathcal{A})$ and $\sigma \in \Sigma$. Moreover, we define $eff_\Sigma([\delta])(\sigma) = \emptyset$,
Now we are ready to give the extensional semantics of an action expression:

Definition 14

$$[\![\alpha]\!]_\Sigma = \lambda\sigma \cdot eff_\Sigma([\![\alpha]\!])(\sigma)$$

(where λ denotes the usual lambda or function abstraction).

Finally we are able to define the accessibility relations $R_\alpha \subseteq \Sigma \times \Sigma$:

Definition 15

$$R_\alpha(\sigma, \sigma') \Longleftrightarrow_{def} \sigma' \in [\![\alpha]\!]_\Sigma(\sigma)$$

Proposition 4

1. $R_{\alpha_1 + \alpha_2} = R_{\alpha_1} \cup R_{\alpha_2}$
2. $R_{\alpha_1 \& \alpha_2} = R_{\alpha_1} \cap R_{\alpha_2}$
3. $R_{\overline{\overline{\alpha}}} = R_\alpha$
4. $R_{\overline{\alpha_1 + \alpha_2}} = R_{\overline{\alpha_1} \& \overline{\alpha_2}}$

5. $R_{\overline{\alpha_1 \& \alpha_2}} = R_{\overline{\alpha_1} + \overline{\alpha_2}}$

Proof: Follows directly from the definition of $[\![\alpha]\!]_\Sigma$. □
Now we have properly defined our Kripke-models, we can give the semantic interpretation of formulas in *Form*:

Definition 16 Given $\mathcal{M} = (\Sigma, \pi, \{R_\alpha | \alpha \in Act\})$ as above, and $\sigma \in \Sigma$, we define:

- $(\mathcal{M}, \sigma) \models p \iff \pi(\sigma)(p) = tt$ (for $p \in Prop$)
- $(\mathcal{M}, \sigma) \models \phi_1 \wedge \phi_2 \iff (\mathcal{M}, \sigma) \models \phi_1$ and $(\mathcal{M}, \sigma) \models \phi_2$
- $(\mathcal{M}, \sigma) \models \neg\phi \iff$ not $(\mathcal{M}, \sigma) \models \phi$
- $(\mathcal{M}, \sigma) \models [\alpha]\phi \iff \forall\sigma'[R_\alpha(\sigma, \sigma') \Rightarrow (\mathcal{M}, \sigma') \models \phi]$
- ϕ is *valid* w.r.t. model $\mathcal{M} = (\Sigma, \pi, \{R_\alpha | \alpha \in Act\})$, notation $\mathcal{M} \models \phi$, if $(\mathcal{M}, \sigma) \models \phi$ for all $\sigma \in \Sigma$.
- ϕ is *valid*, notation $\models \phi$, if ϕ is valid w.r.t. all models $\mathcal{M}$ of the form considered above.

Theorem 2 *The following formulas and properties hold:*

1. $\models \phi$ *(all instances* ϕ *of propositional tautologies)*
2. $\models [\alpha](\phi_1 \to \phi_2) \to ([\alpha]\phi_1 \to [\alpha]\phi_2)$
3. $\models \phi$ *and* $\models \phi \to \psi \Longrightarrow \models \psi$
4. $\models \phi \Longrightarrow \models [\alpha]\phi$
5. $\models [\mathbf{fail}]\phi$
6. $\models [\alpha_1 + \alpha_2]\phi \leftrightarrow [\alpha_1]\phi \wedge [\alpha_2]\phi$
7. $\models ([\alpha_1]\phi \vee [\alpha_2]\phi) \to [\alpha_1 \& \alpha_2]\phi$
8. $\models ([\overline{\alpha_1}]\phi \vee [\overline{\alpha_2}]\phi) \to [\overline{\alpha_1 + \alpha_2}]\phi$
9. $\models [\overline{\alpha_1 \& \alpha_2}]\phi \leftrightarrow [\overline{\alpha_1}]\phi \wedge [\overline{\alpha_2}]\phi$
10. $\models [\overline{\overline{\alpha}}]\phi \leftrightarrow [\alpha]\phi$
11. $\models F(\alpha_1 + \alpha_2) \leftrightarrow F(\alpha_1) \wedge F(\alpha_2)$
12. $\models (F(\alpha_1) \vee F(\alpha_2)) \to F(\alpha_1 \& \alpha_2)$
13. $\models (O(\alpha_1) \vee O(\alpha_2)) \to O(\alpha_1 + \alpha_2)$
14. $\models O(\alpha_1 \& \alpha_2) \leftrightarrow O(\alpha_1) \wedge O(\alpha_2)$
15. $\models P(\alpha_1 + \alpha_2) \leftrightarrow P(\alpha_1) \vee P(\alpha_2)$
16. $\models P(\alpha_1 \& \alpha_2) \to (P(\alpha_1) \wedge P(\alpha_2))$

Proof: 1 - 4 follow directly from the fact that we have provided a Kripke-style modal semantics. 5. is immediate. 6 - 10 follow from Proposition 4. 11 - 16 follow from the validity of 1 -10 (cf. [Mey88]).□

3 Permission

We will now take a closer look at the definition of the deontic operators (especially permission) and discuss their relation with the Free Choice Paradox. The definition of the deontic operators, as is standard for this kind of approach and as given in the previous section (cf. [Mey88, WMW89, WM93, Mey92]), is as follows:

$$\begin{aligned} O(\alpha) &= [\overline{\alpha}]Violation \\ F(\alpha) &= [\alpha]Violation \\ P(\alpha) &= \neg F(\alpha) \\ &= \neg[\alpha]Violation \end{aligned}$$

We will only discuss the definition of the permission operator at this place. A discussion of the other operators (especially obligation) can be found at several places in the literature [Mey88, WM93, Mey92, WMW89, Dig92].
The definition above states that α is permitted if there is a way to do α such that it does not lead to a violation state (a state in which the special proposition $Violation$ is true). Remember that α denotes an action possibly performed in parallel with other events. So the specification of α contains an implicit choice about the context in which α is performed. Of course, this definition is quite weak. An action is permitted if there is a way to do it in a desirable way. This means that an action might be permitted, even if there is only one way that does not lead to a violation state (e.g. when α is done only together with β), while all other ways lead to a violation. If we assume that each action is initiated by some actor, then $P(\alpha)$ says that there is one way in which the actor can perform the action named α such that it does not lead to a violation state. However, what we would like $P(\alpha)$ to mean is that the actor is permitted to choose how to perform α. In the current formalization of permission the actor does not have a free choice to perform the action named α. We will therefore introduce a second operator (P_s), which formalizes the intuition that the actor is free to choose any way of performing the action denoted by α.
At this point it should also be noted that the use of the *only* operator to delimit the choice of how to perform the action within the context of a permission does NOT by itself resolve the paradox. It can easily be seen that $\models P(only(a)) \rightarrow P(only(a+b))$.

In order to avoid the Free Choice paradox, we will adopt another definition for permission (while retaining the definitions for obligation and prohibition), which resembles the strong permission referred to in section 1.

Definition 17

$$\begin{aligned} P_s(\alpha) &= [\alpha]\neg Violation \\ O(\alpha) &= [\overline{\alpha}]Violation \\ F(\alpha) &= [\alpha]Violation \end{aligned}$$

Which means that α is permitted if α does never lead to a violation state.

This definition solves the Free Choice Paradox, because of the following observation:

$$\not\models [\alpha]\phi \longrightarrow [\alpha+\beta]\phi$$

and in particular

$$\not\models [\alpha]\neg Violation \longrightarrow [\alpha+\beta]\neg Violation$$

Thus:

$$\not\models P_s(\alpha) \longrightarrow P_s(\alpha+\beta)$$

In the same way the definition also solves the paradox of context-sensitive permission:

$$\not\models [\alpha \& \beta]\phi \longrightarrow [\alpha]\phi$$

and thus

$$\not\models P_s(\alpha \& \beta) \longrightarrow P_s(\alpha)$$

Although this definition of the permission operator solves the Free Choice Paradox it also raises three new issues. We will now discuss each of these three issues.

3.1 Permission to fail

The new, strong definition for the permission operator implies that **fail** is also permitted. Actually Segerberg [Seg82] defines a strong permission starting with the permission to fail as the central property. That **fail** is permitted with the new definition can easily be seen from the following:

$$\begin{array}{l} for\ all\ \phi : \models [\mathbf{fail}]\phi \Rightarrow \\ \models [\mathbf{fail}]\neg Violation \Rightarrow \\ \models P_s(\mathbf{fail}) \end{array}$$

So it follows directly from the definition that **fail** is permitted. Although at first sight this might look a bit strange it does not pose any problems for the theory. The only consequence is that, at any moment, it is permitted to terminate all activities (get into a deadlock). However, this does not mean that it should be done and in fact it is also forbidden to do so. (This can be easily checked from the definition.)

3.2 Permission for conjunction

The second issue raised by the strong definition of the permission is the following proposition:

Proposition 5

$$\models P_s(\alpha) \longrightarrow P_s(\alpha \& \beta)$$

Proof: Follows directly from the fact that $\models [\alpha]\phi \longrightarrow [\alpha \& \beta]\phi$ and Theorem 2. □
So, if α is permitted, it is (also) permitted in any combination with other actions. This leads to the following example:

$$P_s(\text{fire a gun}) \longrightarrow P_s(\text{fire a gun \& aim at the president})$$

Given our interpretation of $P_s(\alpha)$ as "no matter how you perform α a permitted state of the world results", this is natural.
The implication follows from the following line of reasoning:

$$\begin{array}{l} [\![fire\ a\ gun\ \&\ aim\ at\ the\ president]\!] \subseteq [\![fire\ a\ gun]\!] \Rightarrow \\ \forall\phi(\models [fire\ a\ gun]\phi \longrightarrow\ [fire\ a\ gun\ \&\ aim\ at\ the\ president]\phi) \Rightarrow \\ \models [fire\ a\ gun]\neg Violation \longrightarrow \\ [fire\ a\ gun\ \&\ aim\ at\ the\ president]\neg Violation \Rightarrow \\ \models P_s(fire\ a\ gun) \longrightarrow\ P_s(fire\ a\ gun\ \&\ aim\ at\ the\ president) \end{array}$$

Given our strong interpretation of the permission, using the prudential assumption of Grice [Gam91a, Gam91b] about the maxims of communication, what we actually mean by:

It is permitted to fire a gun

is more appropriately represented as:

$$P_s(only(fire\ a\ gun))$$

This permission does still imply permissions like

$$P_s(only(fire\ a\ gun)\ \&\ shoot\ the\ president)$$
$$P_s(only(fire\ a\ gun\ \&\ shoot\ the\ president))\ \ or$$
$$P_s(only(fire\ a\ gun)\ \&\ only(shoot\ the\ president))$$

But the big difference is that all these actions are equal to the action denoted by **fail**. That this is indeed true can be easily seen from the semantics of the different action expressions.

So, in general, we do have:

$$\models P_s(\alpha) \longrightarrow P_s(\alpha\&\beta)$$

but by choosing the appropriate logical interpretation for the action expressions, we force $\alpha\&\beta$ to be **fail**, thereby effectively blocking the performance of the combination of actions. More specifically we have:

$$\models P_s(only(\alpha)) \longrightarrow P_s(only(\alpha)\&only(\beta))$$

but $only(\alpha)\&only(\beta) =_{\mathcal{D}}$ **fail** if α and β are action terms that contain no negation.
Therefore, we can conclude that the problem that is seemingly created by the strong definition of the permission can be resolved by using an appropriate interpretation of the natural language sentences (using the *only* operator).

3.3 Complementarity of permission and prohibition

The third issue that is raised by the definition is that the strong permission is no longer complementary to the prohibition. I.e. $P_s(\alpha) \neq \neg F(\alpha)$. With the new definition for the strong permission there can be action expressions α such that $\neg P_s(\alpha) \wedge \neg F(\alpha)$.
In fact, this point is an advantage of this strong definition of the permission. Usually, it is not the case that every action is either permitted or forbidden. Many actions are neither permitted nor forbidden, although they might be permitted or forbidden in some combination with other actions.
Of course, we do still have the following proposition:

Proposition 6 *Let* $\alpha \neq_{\mathcal{D}}$ **fail** *then:*

$$\models P_s(\alpha) \longrightarrow \neg F(\alpha)$$

or equivalently

$$\models F(\alpha) \longrightarrow \neg P_s(\alpha)$$

Proof: By definition.□
I.e. if something is permitted then it is not forbidden, and if something is forbidden then it is not permitted.

The fact that strong permission and prohibition are not complementary in our system also influences the following implication that holds in all the deontic systems that are defined in terms of dynamic logic as is done in [Mey88].

$$\models O(\alpha) \wedge \neg P(\alpha) \longrightarrow F(\mathbf{any})$$

Which means that if an action is obliged and at the same time not permitted then it is forbidden to do anything.
In our system we do have that

$$\models O(\alpha) \wedge F(\alpha) \longrightarrow F(\mathbf{any})$$

But we leave open the possibility that the action named α is not permitted in general (i.e. $\neg P_s(\alpha)$), but is permitted in specific combinations. For instance, it might be permitted to perform the action named $\alpha\&\beta\&\gamma$. By performing this combination we fulfil the obligation while not doing anything forbidden.
Note that we do **NOT** have:

$$\models O(\alpha) \longrightarrow \neg F(\alpha) (= \neg O(\overline{\alpha}))$$

So, there is no *guarantee* that if the action named α is obliged it is not forbidden. It is still possible to have contradictory obligations.

3.4 Preliminary conclusions

Taking the above points into account, we can state that the Free Choice Paradox is solved by using this strong definition of the permission. The appropriate use of the *only* operator in the translation from natural language to logic avoids the problems that were raised with this definition.
Although the combination of the *only* operator for action expressions and the strong definition for the permission resolves the problems with the Free Choice paradox, it also raises some new issues.

The first point to be mentioned is that the *only* operator does not distribute over any of the other connectors of actions, which limits its use. However, in practice, the *only* operator is mainly applied to action terms. Therefore, this restriction on the use of the *only* operator does not have very severe consequences.

The second point is, in practice, maybe more serious. $P_s(\alpha)$ means that the action denoted by α is permitted, possibly in combination with other actions. It is not necessary to explicitly state all the combinations that are allowed. Using $P_s(only(\alpha))$ is much stricter in the sense that it only allows the action denoted by α to be performed by itself. If it can be performed in combination with other actions, then this has to be explicitly stated, unless it can be performed in combination with any other action, in which case we can use $P_s(\alpha)$.
The latter, however, almost never (if ever) occurs. It is almost always possible to find an action which should not be performed in combination with the permitted one (e.g. take any forbidden action, like committing murder).
This also bears a resemblance with problems of knowledge representation in artificial intelligence, like the so-called qualification problem, where one does not want to make explicit the numerous qualifications that (may) influence an action's normal properties (like its being permitted in this particular case). On the other hand it is also almost always possible to find actions that have no influence on the deontic status of the combined action. We would

like to permit the combination of the permitted action with these actions, without explicitly having to mention all of them.

To counter this problem we will propose another mechanism for the actions, in the next section, which is less restrictive, but still keeps the properties of the *only* operator with respect to the permission operator.

4 Contexts

In this section, we will introduce contexts for actions. The effect of using contexts will be less strict than that of using the *only* operator. The intuitive idea is that with an action expression α we specify a set of events $\mathcal{C}$ relative to which the semantics of the action expression is taken. In the standard case this set of events is the set $\mathcal{A}$, the set of all possible events (in the system). We will give a characterization of *safe* contexts, which are contexts within which an action can be performed without changing its effects.
We will now give the necessary definitions to introduce contextual actions in the syntax and give a semantics for them.

4.1 Contextual actions

We start with a definition that is needed to facilitate the other definitions.

Definition 18 If $\alpha \in Act$ then $at(\alpha)$ is the set of events such that the underlined version of that event occurs in α.

E.g. if $\alpha = \underline{a}\&(\overline{\underline{b} + \underline{c}})$ then $at(\alpha) = \{a, b, c\}$.

Now we can introduce the set of contextual action expressions $Act_{\mathcal{C}}$ with elements $\alpha_{\mathcal{C}}$ as follows:

Definition 19 Let $\mathcal{C} \subseteq \mathcal{A}$. $Act_{\mathcal{C}}$ is the minimal set satisfying:

1. If $\alpha \in Act$ and $at(\alpha) \subseteq \mathcal{C} \subseteq \mathcal{A}$ then $\alpha_{\mathcal{C}} \in Act_{\mathcal{C}}$.
2. If α, β $Act_{\mathcal{C}}$ then $\overline{\alpha}$, $\alpha\&\beta$ and $\alpha + \beta$ are elements of $Act_{\mathcal{C}}$.

The set $\mathcal{C}$ is called the context of the action.
If the context $\mathcal{C}$ of an action denoted by α is $\mathcal{A}$ then we write α as a shorthand for $\alpha_{\mathcal{A}}$.

To give the semantics of contextual action expressions we can use the same semantic operators as were used for the normal action expressions, except for the negation, which becomes context-dependent.

Definition 20 The definition of "$\sim_{\mathcal{C}}$" is given as follows:

1. For an s-set S,
$$S^{\sim_{\mathcal{C}}} = \begin{cases} \wp^+(\mathcal{C})\backslash\{S\} & \text{if } S \neq [\delta] \\ \wp^+(\mathcal{C}) & \text{if } S = [\delta] \end{cases}$$
2. For a non-empty set $T \in \mathcal{D}$
$$T^{\sim_{\mathcal{C}}} = \bigcap\{S^{\sim_{\mathcal{C}}} | S \in T\}$$

The semantics of the contextual action expressions is now given in the following definition.

Definition 21 The semantic function $[\![]\!] \in Act_{\mathcal{C}} \rightarrow \mathcal{D}$ is given by:

$$
\begin{array}{l}
[\![\underline{a}_{\mathcal{C}}]\!] = \{S \in \wp^{+}(\mathcal{C}) | a \in S\} \\
[\![(\alpha_1)_{\mathcal{C}_1} + (\alpha_2)_{\mathcal{C}_2}]\!] = [\![(\alpha_1)_{\mathcal{C}_1}]\!] \bullet [\![(\alpha_2)_{\mathcal{C}_2}]\!] \\
[\![(\alpha_1 + \alpha_2)_{\mathcal{C}}]\!] = [\![(\alpha_1)_{\mathcal{C}}]\!] \bullet [\![(\alpha_2)_{\mathcal{C}}]\!] \\
[\![(\alpha_1)_{\mathcal{C}_1} \& (\alpha_2)_{\mathcal{C}_2}]\!] = [\![(\alpha_1)_{\mathcal{C}_1}]\!] \circledcirc [\![(\alpha_2)_{\mathcal{C}_2}]\!] \\
[\![(\alpha_1 \& \alpha_2)_{\mathcal{C}}]\!] = [\![(\alpha_1)_{\mathcal{C}}]\!] \circledcirc [\![(\alpha_2)_{\mathcal{C}}]\!] \\
[\![\overline{\alpha}_{\mathcal{C}}]\!] = [\![\alpha]\!]^{\sim_{\mathcal{C}}} \\
[\![\overline{\alpha_{\mathcal{C}}}]\!] = [\![\alpha_{\mathcal{C}}]\!]^{\sim} \\
[\![only(\alpha)_{\mathcal{C}}]\!] = [\![only(\alpha)_{\mathcal{A}}]\!] \circledcirc \wp^{+}(\mathcal{C}) \\
[\![\mathbf{fail}_{\mathcal{C}}]\!] = \{[\delta]\} \\
[\![\mathbf{any}_{\mathcal{C}}]\!] = \wp^{+}(\mathcal{C})
\end{array}
$$

It can easily be seen that these definitions are the same as before, except that the semantics are not taken relative to $\mathcal{A}$ but relative to a context $\mathcal{C}$. This means that $\alpha_{\mathcal{C}}$ is to be read as "the action denoted by α, possibly performed together with any set of actions from $\mathcal{C}$". Of course, if the context $\mathcal{C}$ is equal to the set of all actions $\mathcal{A}$, then the action expressions have the standard interpretation.

The following proposition relates the notion of contexts with the *only* operator.

Proposition 7 *Let $\alpha_{\mathcal{C}} \in Act_{\mathcal{C}}$ be such that it does not contain any occurrence of the negation operator and $\mathcal{C} = at(\alpha)$ then*

$$\alpha_{\mathcal{C}} =_{\mathcal{D}} only(\alpha)$$

Proof: Easy by induction.□
The above proposition can be read as "the action denoted by $only(\alpha)$ is the action denoted by α done in the context of its own literals".

In the above proposition we excluded action expressions that contain negations. That the equality does not hold for these action expressions can be seen from the following simple example (where we assume that $\mathcal{A} = \{a, b\}$):

$$[\![only(\underline{a})]\!] = [\![\overline{a}]\!] = \{[b]\}$$

while

$$[\![\overline{\underline{a}}_a]\!] = \{[\delta]\}$$

The definitions for the extensional (state-transforming) semantics remain the same as before but are based on the new intensional semantics.
In the next section we will show how the contexts for actions can be related to the deontic operators.

4.2 Safe contexts

The intuitive meaning of a (deontically) safe context for α is that the action denoted by α is performed possibly together with some other actions that have no influence on the effects of the action denoted by α.
I.e. $(open\ the\ window)_{\mathcal{C}}$ where $\mathcal{C}$ is a safe context might include in its semantics opening the window while at the same time watching television, but not opening the window and at the same time turning on the heater. Before we give the definition of a safe context, it should be noted that contexts are given with respect to certain properties of the world. For instance,

in the example above we take into account the temperature, but not the use of electricity. We will take this into account in the definition by introducing a propositional context that indicates which aspects of the world are taken into account. Because here we are interested in deontic aspects, we take at least the atom $Violation$ into account into our contexts. More formally:

Definition 22 A propositional context Φ is a set of propositions.
If Φ is a set of propositions containing $\neg Violation$ and $Violation$, then we call Φ a **deontic propositional context**.

Note: propositional contexts are not consistent (by definition) and should thus not be considered as theories. They should be considered as reference sets possibly ranging over different possible worlds.
The formal definition of a (deontically) safe context can be given as follows:

Definition 23 Let Φ be a (deontic) propositional context and let $\alpha_{\mathcal{C}} \in Act_{\mathcal{C}}$ then $\mathcal{C}$ is a **(deontically) safe** context for α in σ with respect to Φ iff for all $\phi \in \Phi$

$$\sigma \models ([only(\alpha)]\phi \rightarrow [\alpha_{\mathcal{C}}]\phi)$$

Note: $only(\alpha)$ stands for $only(\alpha)_{\mathcal{A}}$.
If σ is clear from the (textual) context then we will just say that $\mathcal{C}$ is **(deontically) safe** for α.

Proposition 8 *If C is* **deontically safe** *for α then*

$$\sigma \models P_s(only(\alpha)) \longrightarrow P_s(\alpha_{\mathcal{C}})$$
$$\sigma \models F(only(\alpha)) \longrightarrow F(\alpha_{\mathcal{C}})$$

Proof: Follows direct from the definition.□
The following example shows how deontically safe contexts can be used.

> We assume that chewing gum does not affect the deontic effect of firing a gun. However, aiming at the president does affect the deontic effect of firing a gun. Therefore "chewing gum" is member of a deontically safe context of "firing a gun" while "aiming at the president" is not.

Therefore we have that:

$$\sigma \models P_s(only(firing-a-gun)) \rightarrow P_s(firing-a-gun_{\{firing-a-gun,\, chewing-gum\}})$$

but

$$\sigma \not\models P_s(only(firing-a-gun)) \rightarrow P_s(firing-a-gun_{\{firing-a-gun,\, aiming-at-president\}})$$

The introduction of ((deontically) safe) contexts makes it possible to restrict the interpretation of an action expression while still allowing the possibility to perform an action in combination with some other actions.

Proposition 9 *Let α and β be action terms without negations and let $at(\alpha) \cup at(\beta) \subseteq \mathcal{C}_i$ $i = 1, 2$*
then $\alpha_{\mathcal{C}_1} \& \beta_{\mathcal{C}_2} \neq_{\mathcal{D}}$ **fail**

Proof: From the premisses it follows that the s-set $at(\alpha) \cup at(\beta)$ is an element from $[\![\alpha_{\mathcal{C}_1} \& \beta_{\mathcal{C}_2}]\!]$ and therefore $\alpha_{\mathcal{C}_1} \& \beta_{\mathcal{C}_2} \neq_{\mathcal{D}}$ **fail**.
□

The above proposition shows that there are cases in which it is possible to perform a combination of actions in a certain context, while this would not be possible if the *only* operator was applied on one or both of the action expressions.
E.g.

$$[\![\underline{a}_{\{a,b\}} \& \underline{b}_{\{a,b\}}]\!] = \{ \begin{bmatrix} a \\ b \end{bmatrix} \}$$

while

$$[\![only(\underline{a}) \& \underline{b}]\!] = \{\delta\}$$

The following proposition can be easily proven:

Proposition 10 *Let C be a* **deontically safe** *context for α then*

$$\models P_s(\alpha_{\mathcal{C}}) \wedge F(\beta) \Longrightarrow \alpha_{\mathcal{C}} \& \beta =_{\mathcal{D}} \textbf{fail}$$

Which means that if the action denoted by $\alpha_{\mathcal{C}}$ is strongly permitted, it is not possible to perform it in combination with an action that is forbidden. Note that the above implication holds trivially if no context for α is specified. The above proposition states that even when the permission to perform the action denoted by α is weakened to a permission to perform the action denoted by α only in some (safe) context of other actions, the above implication still holds.
The consequence is that we do have the following (assuming that it is forbidden to shoot the president and that $\mathcal{C}$ is a deontically safe context):

$$P_s(chewing - gum_{\mathcal{C}}) \longrightarrow chewing - gum_{\mathcal{C}} \& shooting\ the\ president =_{\mathcal{D}} \textbf{fail}$$

A final point about safe contexts that should be noted is that if an event b is an element of a safe context for $\underline{a}$ it does not imply that a is an element of a safe context for $\underline{b}$. I.e. performing the action denoted by the combination $\underline{a} \& \underline{b}$ might include the effects from performing the action denoted by $only(\underline{a})$ while it does not include the effects of performing the action denoted by $only(\underline{b})$. This happens, for instance, if the action denoted by $\underline{a}$ is permitted in isolation and also in combination with the action denoted by $\underline{b}$, but the action denoted by $\underline{b}$ is forbidden in isolation. This can be seen more concretely in the following example:

> It is permitted to close the door. It is also permitted to close the door and at the same time open the window. However, it is forbidden to open the window without at the same time closing the door.

The above means that *open the window* is part of a safe context of *close the door*, but *close the door* is not part of a safe context of *open the window*. This is due to the fact that "closing the door" changes the effect of "opening the window". It is *forbidden* to open the window and not close the door, but *permitted* to do both together.

5 Conclusions

We started with the observation that the Free Choice paradox has two forms, each of which is the dual of the other. The first form is $P(\alpha) \rightarrow P(\alpha + \beta)$ for action expressions α and β, and is called the paradox of free choice permission. The second form is $P(\alpha \& \beta) \rightarrow P(\alpha)$, and we called this the paradox of context-sensitive permission.
We noted that any solution of the Free Choice paradox based on the concept of choice won't work for the paradox of context-sensitive permission. In this paper we showed that both forms of the paradox can be eliminated if we take the context in which an action is performed into account.

In a survey of paradoxes in deontic logic, al-Hibri [AlH78] rightly points out that the inference

$$P(\alpha) \rightarrow P(\alpha + \beta)$$

omits something, viz. the admissibility of β. If we add the information that β is not permitted then it follows that there is actually no free choice between α and β:

$$(P(\alpha \cup \beta) \wedge \neg P(\beta)) \Leftrightarrow ((P(\alpha) \vee P(\beta)) \wedge \neg P(\beta)) \Rightarrow P(\alpha)$$

However, the observation that we can derive what we want if we put in more information at the beginning, does not block the faulty inference itself. Her observation does point the way to our solution, though, for it makes clear that to make a choice, we must have permission to perform the choice in any way we want.

We have given a solution of the Free Choice Paradox by choosing a strong definition for the permission operator, that does not give the actor the permission to choose between ways to perform the action that lead to a violation state and ways that do not lead to such a state. This definition of the permission operator concurs with an intuitive idea that a state of violation should always be avoided by giving apropriate deontic constraints. In the same way as a prohibition indicates that performing a certain action always leads to a violation, the strong permission guarantees that if the action is performed (in isolation) it will never lead to a violation.
The combination of this operator with the *only* operator for actions avoids the problems that were raised earlier against this definition, by eliminating all possible contexts of the action. It was noted, however, that the *only* operator only has a limited use (for actions in strict disjunctive normal form) and excluding all contexts is usually too strict.

The introduction of explicit contexts gives the opportunity to combine the strong definition of the permission operator with an "open" specification of action expressions. This "open" specification is limited to possible combinations with actions that do not interfere with the specified action by defining **safe** contexts. It is not very difficult to find a safe context for an action. ($at(\alpha)$ is a safe context for an action denoted by α.) However, finding the biggest safe context for an action is not that simple. In practice, we are only interested to see whether a particular event is part of a safe context of an action or not. Therefore it is usually not necessary to explicitly construct the biggest safe context of an action. The idea is that there are some default biggest safe contexts for each action.

The proposed solution of the Free Choice Paradox shows a connection between free choice and the context-sensitivity of permission: An agent has a free choice between alternatives if all ways of doing both alternatives are permitted, i.e. if both alternatives can be done in all possible contexts.

Acknowledgements:

We would like to thank the working group Logic and Law for their input. We mention especially Hans Weigand for pointing out a severe fallacy in a previous version of the paper. Also we are greatly indebted to the anonymous referees for their stimulating remarks.

References

[Aqv84] L. Åqvist. Deontic logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic II*, pages 605–714. Reidel, 1984.

[AlH78] A. al-Hibri. Deontic Logic: A Comprehensive Appraisal and a New Proposal. University Press of America, 1978

[BW90] J.C.M. Baeten and W.P. Weijland. Process Algebra. Cambridge University Press, 1990.

[Bro86] M. Broy. A theory for nondeterminism, parallelism, communication and concurrency. *Theoretical Computer Science*, vol.45, pages 1–62, 1986.

[dBKM+86] J.W. de Bakker, J.N Kok, J.-J.Ch. Meyer, E.-R. Olderog, and J.I. Zucker. Contrasting themes in the semantics of imperative concurrency. In J.W. de Bakker, W.P. de Roever, and G. Rozenberg, editors, *Current Trends in Concurrency: Overviews and Tutorials*, pages 51–121. LCNS 224 Springer, Berlin, 1986.

[Cas81] H.-N. Castañeda. The Paradoxes of Deontic Logic: The simplest solution to all of them in one fell swoop. In R. Hilpinen, editor, *New Studies in Deontic Logic*. pages 37–85, Reidel, 1981.

[Dig89] F. Dignum. A language for modelling knowledge bases. Ph.d. thesis, Vrije Universiteit, Amsterdam, 1989.

[DM90] F. Dignum and J.-J.Ch. Meyer. Negations of transactions and their use in the specification of dynamic and deontic integrity constraints. In M. Kwiatkowska, M.W. Shields, and R.M. Thomas, editors, *Semantics for Concurrency, Leicester 1990*, pages 61–80, Berlin, 1990. Springer.

[Dig92] F. Dignum. Using transactions in integrity constraints. *Workshop on Applied Logic*, Amsterdam, 1992.

[Gam91a] L.T.F. Gamut. *Logic, Language and Meaning 1: Introduction to Logic*. University of Chicago Press, 1991. L.T.F. Gamut is a pseudonym for J.F.A.K. van Benthem, J. Groenendijk, D. de Jongh, M. Stokhof, and H. Verkuyl.

[Gam91b] L.T.F. Gamut. *Logic, Language and Meaning 2: Intensional Logic and Logical Grammar*. University of Chicago Press, 1991. L.T.F. Gamut is a pseudonym for J.F.A.K. van Benthem, J. Groenendijk, D. de Jongh, M. Stokhof, and H. Verkuyl.

[Har79] D. Harel. *First Order Dynamic Logic*. Springer, 1979. Lecture Notes in Computer Science 68.

[Hil81] R. Hilpinen. Conditionals in possible worlds. In G. Fløstad, editor, *Contemporary Philosophy, a New Survey*, vol.1, pages 299–335. Reidel, 1981.

[Kam73] H. Kamp. Free choice permission. *Aristotelian Society Proceedings N.S.*, vol.74, pages 57–74, 1973-1974.

[KM87] S. Khosla and T.S.E. Maibaum. The prescription and description of state based systems. In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Temporal Logic in Specification*, pages 243–294. Springer, 1987. Lecture Notes in Computer Science 398.

[KT90] D. Kozen and J. Tiuryn. Logics of programs. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 789–840. Elsevier Science Publishers, 1990.

[McC83] L.T. McCarty. Permissions and obligations. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 287–294, Karlsruhe, W. Germany, 1983. Kaufmann.

[McC86] L.T. McCarty. Permissions and obligations: An informal introduction. In A.A. Martino and F.S. Natali, editors, *Automated Analysis of Legal Texts*, pages 307–337. North-Holland, 1986. Edited versions of selected papers from the Second International Conference on "Logic, Informatics, Law," Florence, Italy, September 1985.

[Mey87] J.-J.Ch. Meyer. A simple solution to the 'deepest' paradox in deontic logic. *Logique et Analyse, Nouvelle Série*, vol.30, pages 81–90, 1987.

[Mey88] J.-J.Ch. Meyer. A different approach to deontic logic: Deontic logic viewed as a variant of dynamic logic. *Notre Dame Journal of Formal Logic*, vol.29, pages 109–136, 1988.

[Mey90] J.-J.Ch. Meyer. Using Programming Concepts in Deontic Reasoning. In R. Bartsch, J. van Benthem, and P. van Emde Boas, editors, *Semantics and Contextual Expression*, pages 117–145, Foris, Dordrecht, 1989.

[Mey92] J.-J.Ch. Meyer. Free Choice Permissions and Ross's Paradox: Internal vs. External Nondeterminism. In C.P.Dekker and M.Stockhof, editors, *Proceedings 8th. Amsterdam Collloquium*,pages 367–380, University of Amsterdam, 1992.

[Seg82] K. Segerberg. A deontic logic of action. *Studia Logica*, vol.41, pp.269–282, 1982.

[WMW89] R. Wieringa, J.-J.Ch. Meyer, and H. Weigand. Specifying dynamic and deontic integrity constraints in knowledge bases. *Data & Knowledge Engineering*, vol.4, pages 157–189, 1989.

[WWMD91] R. Wieringa, H. Weigand, J.-J.Ch. Meyer, and F. Dignum. The inheritance of dynamic and deontic integrity constraints. In *Annals of Mathematics and Artificial Intelligence 3*, pages 393–428. Baltzer A.G., 1991.

[WM93] R.J. Wieringa and J.-J.Ch. Meyer. Actors, Actions and Initiative in Normative System Specification. In *Annals of Mathematics and Artificial Intelligence*,Vol 7, pages 289–346, 1993.

[Wri51] G.H. von Wright. Deontic logic. *Mind*, vol.60, pages 1–15, 1951.

[Wri68] G.H. von Wright. An Essay in Deontic Logic and the General Theory of Action. *Acta Philosophica Fennica*, Fasc. 21, North-Holland, 1968.