Contextual Permission
A Solution to the Free Choice Paradox

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

Abstract

In this paper, we give a solution to the Free Choice Paradox. This is done in two stages. First, we have a close look at the logical interpretation of the natural language statements that lead to the paradox. This leads to making the important distinction of permitting an action in isolation or permitting it in combination with some or any other action, i.e. in a certain context. This distinction is made formal by the introduction of a new operator on actions, which forces them to be performed in isolation.
With this distinction made clear it is possible to give a "new", stronger definition for the permission operator, which solves the Free Choice Paradox and which does not lead to any new inconsistencies or paradoxes.

To appear in: Proceedings, Workshop on Deontic Logic in Computer Science (DEON'94), 6–8 January 1994.

1 Introduction

The paradox of free choice permission is described at several places [Hil, Kam74] and has defied several attempts to be solved. It can be exemplified as follows:

[*] 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

[‡] Free University of Amsterdam, Faculty of Mathematics and Computer Science, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands, e-mail:roelw@cs.vu.nl

(1) P(talk to the president) $\longrightarrow$ P(talk to the president $\cup$ shoot the president)

If it is permitted to talk to the president then it is also permitted to talk to the president or shoot him. The problem lies in the fact that the combined action of "talking or shooting the president" is permitted, although the choice of how this action should be performed is not free.

Basically, there are two approaches to resolving the paradox. The first is by limiting the choice. That is, the disjunction of the actions does not mean a free choice between the actions. The choice can be limited by some context. A good example of this type of solution can be found in [MW92, Mey92] . However, although this approach does solve the paradox, it also introduces some new problems. More important is that it will still allow for the following implication to be true:

(2) P(shoot a gun & aim in the air) $\longrightarrow$ P(shoot a gun)

Although this statement is not mentioned as an example of the free choice paradox it is basically of the same form. It can even be written in the format of the original paradox as follows:

(3) P(shoot a gun & aim in the air) $\longrightarrow$ P((shoot a gun & aim in the air) $\cup$ (shoot a gun & not aim in the air))

It may be clear that any solution to the free choice paradox should also block the above implication. Therefore, any method that works only on the basis of the choice operator ($\cup$) will not be sufficient to solve the paradox completely because (2) does not contain a choice operator but can still be seen as an instance of the free choice paradox as shown through (3).

The second approach to resolving the paradox (and the one that we will take in this paper) is by invalidating the implication. If the implication is no longer valid then the paradox does not occur. Invalidating the implication can only be done by giving a different semantics for the permission. Of course, this might also lead to some other difficulties. However, we will see that these problems can be solved through a proper representation of deontic sentences in logic.

Before we give a new definition for permission we will, in the next section, first discuss the exact meaning of the combination of deontic operators with actions. This will lead to some new definitions for actions which will be introduced in section 3. In section 4, 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 5, we discuss a possible refinement to the theory based on the previous sections. Finally, in section 6, we draw some conclusions and point out some future research.

2 Analysis of natural language

Before we start to analyze the Free Choice Paradox it is important to describe the exact meaning of the combination of the deontic operators with actions, as they occur in natural language. We will look at the simple action "*open the window*" and define its meaning when it is combined with each of the deontic operators. It will appear that the main point of interest is whether the action is supposed to be performed in isolation, in combination with any other action or in combination with a restricted set of other actions. The underlying maxim for the meaning of all the combinations of actions with deontic operators is that a state of violation (of some constraint) should be avoided. E.g. if an action is permitted then it is certain that the performance of that action will not lead to a violation state if performed in isolation.

Obligation

The sentence "It is obliged to open the window" means that, at least the window has to be opened. This action should be performed. It might be in parallel with any other action, but that does not matter. In this sense the action in this sentence can therefore be taken to be "open". For more details see [Dig92] .

Prohibition

The sentence "It is forbidden to open the window" also uses the action in an open sense. The window should not be opened, no matter which other action is performed in parallel. I.e. if the window is not opened but the heater is turned of, we will still not arive in a violation state.
Notice that in reality a prohibition is usually not that strict. Usually it is

possible to find some exceptions to the prohibition. E.g. "It is forbidden to open the window, except when the window is replaced". For the moment we will not consider these exceptions and assume that they are specified specifically as done above.
One will, however, not use the sentence "It is forbidden to open the window", if there are *normal* actions such that if they are performed together with the opening of the window will lift the prohibition.
E.g. if it is allowed to open the window when at the same time the door is closed, then one would not normally say that it is forbidden to open the window.

Permission

The sentence "It is permitted to open the window" does NOT use the action in an open sense. The permission to open the window does not seem to imply that this can be done together with any other action. E.g. the above permission does not mean that it is permitted to open the window and turn on the heater. The permission is ONLY for opening the window.

Without having done any linguistic research it seems natural that the meaning of the action together with the obligation and prohibition is different from the meaning of that action together with a permission. This is due to the character of the obligation and prohibition. These two operators can be seen as limiting the wanted actions. This should always be done in as wide a sense as possible, so that it is never possible to approve an unwanted action.
The permission can be seen as making actions wanted. For the same reason as given for the other two operators the permission should be given in the most strict sense, so that it is never possible to perform an unwanted action (that is permitted).

The above observations lead to the conclusion that the (logical) interpretation of the actions is different when combined with different deontic operators. In the next section we will show how these differences can be accommodated in the deontic logic as described in [Mey88, WMW89, DM90, Dig92] .

3 Interpretation for actions in logic

3.1 Introduction

In [Mey88, DM90, Dig92] the standard interpretation of an action was given in an open sense, i.e. if an action is specified it means that that actions is performed, p[ossibly in combination with other actions. The reason for this is twofold. First it is more natural. Usually there is no 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 the other actions.
The second reason is that it facilitates the definition of the negation of an action, which is needed in this approach to model the obligation. See [DM90] for more explanations on this topic.

We will keep this "standard" interpretation of an action, but we will add an operator to indicate that an action is performed in isolation. This operator is (quite obviously) called the "*only*" operator.
The intuitive meaning of $only(\alpha)$ is that α is performed and all other actions are not performed.

In the next section we give the formal semantic definition of the $only$ operator. This definition is based on the semantics of the actions as given in the appendix. This semantics is a simplified version of the one used in [Mey88]. The simplification consists in the restriction to actions, and not considering sequences of actions. This extension can be easily made, but would obscure the points that we try to make in this paper. At this point we will only introduce a few definitions from the appendix that are necessary for the following sections. The definitions that relate the actions to states can be found in the appendix.

We start with a set of **events** $\mathcal{A}$, with typical elements a, b, c.... There is one special event δ, which is not an element of $\mathcal{A}$, and which stands for failure.(comparable to deadlock in process algebra). The set of events $\mathcal{A}$ will form the basis for the semantics of the actions in our language *Act*.
The atomic **actions** in these languages are denoted by underlined versions of events from $\mathcal{A}$. Thus if $a \in \mathcal{A}$ then $\underline{a}$ is an atomic action in *Act*.
The set of all action expressions can now be determined by the following

BNF for its elements (α):

$$\alpha ::= \underline{a} | \alpha_1 \underline{\cup} \alpha_2 | \alpha_1 \& \alpha_2 | \overline{\alpha} | \mathbf{any} | \mathbf{fail}$$

The meaning of $\alpha_1 \underline{\cup} \alpha_2$ is a choice between α_1 and α_2. $\alpha_1 \& \alpha_2$ stands for the parallel execution of α_1 and α_2. The expression $\overline{\alpha}$ stands for the non-performance of the action α. The **any** action is a universal or "don't care which" action. Finally the **fail** action is the action that always fails (deadlock). This action does not lead to a next state.
In the following section we make use of the following concepts of actions:

Definition 1

1. atomic actions $\underline{a}$ and their negations $\overline{\underline{a}}$ are called action literals.
2. an action term is a conjunction (using the & operator) of one or more action literals.

The semantics of actions are given by sets of what we call *synchronicity* sets (or s-sets for short). A synchronicity set denotes a set of events that are executed simultaneously. They are the basic building blocks of the semantics.

Definition 2

1. The set $\{\delta\}$ is a synchronicity set.
2. Every non-empty 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 s-sets will be denoted by $\wp^+(\mathcal{A})$, where the '+' indicates that the s-sets are non-empty.

Definition 3 The domain $\mathcal{D}$ for our model for actions from *Act* is the collection of sets consisting of s-sets.

The semantics of actions in *Act* are defined as follows:

Definition 4 The semantic function $[\![]\!] \in \mathit{Act} \rightarrow \mathcal{D}$ is given by:

$$
\begin{array}{l}
[\![\underline{a}]\!] = \{S \in \wp^+(\mathcal{A}) | a \in S\} \\
[\![\alpha_1 \underline{\cup} \alpha_2]\!] = [\![\alpha_1]\!] \Cup [\![\alpha_2]\!] \\
[\![\alpha_1 \& \alpha_2]\!] = [\![\alpha_1]\!] \Cap [\![\alpha_2]\!] \\
[\![\overline{\alpha}]\!] = [\![\alpha]\!]^{\sim} \\
[\![\textbf{fail}]\!] = \{[\delta]\} \\
[\![\textbf{any}]\!] = \wp^+(\mathcal{A})
\end{array}
$$

In the above definition $\Cup$, $\Cap$ and $\sim$ are operators on the semantic domain $\mathcal{D}$, and roughly correspond to disjunction, conjunction and complement. The precise definition of these operators can be found in the appendix.

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

Definition 5 We define $\alpha_1 =_{\mathcal{D}} \alpha_2$ iff [[α_1]]= [[α_2]].
We define $\alpha_1 > \alpha_2$ iff [[α_1]]$\subseteq$ [[α_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 *involves* α_2.

3.2 Only

We will first introduce the $only$ operator in the syntax:

Definition 6 Let $\alpha \in \mathit{Act}$ and α does not contain the $only$ operator, then $only(\alpha) \in \mathit{Act}$.

We explicitly exclude the possibility to nest the $only$ operator, because it makes no sense to apply the operator twice.

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

Definition 7 An action is in disjunctive normal form if it is a finite disjunction (using the $\underline{\cup}$ operator) of one or more action terms not containing δ.

Definition 8 The action α is in strict disjunctive normal form if it is in disjunctive normal form and for every subexpression of α of the form $\alpha_1 \underline{\cup} \alpha_2$ holds that

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

The semantics of the $only$ operator is only defined on actions that are in strict disjunctive normal format. 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" then something like:"John only opens the window or closes the door". That is, the $only$ operator works more natural on atomic actions (and their conjunctions) then on disjunctions of actions. 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 difficulty is reflected in the definition of the $only$ operator.

Fortunately, in the following we can use the property:

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

Proof: Follows direct from the semantics and the definitions. □
E.g. [[$\underline{a} \underline{\cup} (\underline{a} \& \underline{b})$]]= [[$\underline{a}$]].
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.
It should be noted that there is no unique α^* for each α.

Now, we define the semantics of the $only$ operator in two steps:

Definition 9

1. Let $\underline{a}$ be an atomic action then

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

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

2. Let α and β be action terms then

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

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

   1. α is an action literal or an action term then $[\![only(\alpha)]\!]$ is defined as above.
   2. $\alpha = \alpha_1 \cup \alpha_2$ then $[\![only(\alpha)]\!] = [\![only(\alpha_1)]\!] \cup [\![only(\alpha_2)]\!] \cup [\![only((\alpha_1 \& \alpha_2)^*)]\!]$

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 actions. This is due to the fact that these negated actions 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.
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:

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

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

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

Because the $only$ operator does not ditribute 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 1

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

Proof: Easy by induction.□
This property means that $only(\alpha^*)$ leads to less possible states than α.

Proposition 2 *Let $\alpha \neq_{\mathcal{D}} \beta$ and let α and β both be action terms not containing negation 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 permission operator and the Free Choice Paradox.

4 Permission

We will now take a closer look at the definition of the deontic operators (especially the permission) and discuss their relation with the Free Choice Paradox. The definition of the deontic operators, as it is standard for this kind of approach (cf. [Mey88, WMW89, MW92, 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 the obligation) can be found at several places in the literature [Mey88, MW92, 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 predicate $Violation$ is true). Remember that the meaning of α is that α is performed, possibly in parallel with other actions. 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 α might be permitted, although 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, it also means that $P(\alpha)$ says that the actor (the person that initiates the action) is permitted to choose how to perform α. It depends on the choice of the actor whether or not a violation state will arise. It is not surprising that this notion of permission gives rise to the Free Choice Paradox.

At this point it should also be noted that the introduction of the $only$ operator does NOT by itself resolve the paradox. It can easily be seen that $P(only(a)) \rightarrow P(only(a \underline{\cup} b))$.

In order to avoid this paradox, we use another definition for permission, which is motivated by the analysis of section 2.

Definition 10

$$P(\alpha) = [\alpha]\neg Violation$$

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:

$$[\alpha]\phi \not\longrightarrow [\alpha \underline{\cup} \beta]\phi$$

and therefore also

$$[\alpha]\neg Violation \not\longrightarrow [\alpha \underline{\cup} \beta]\neg Violation$$

Thus:

$$P(\alpha) \not\longrightarrow P(\alpha \underline{\cup} \beta)$$

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

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

and thus

$$P(\alpha \& \beta) \not\longrightarrow P(\alpha)$$

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.

4.1 Permission to fail

The new, strong definition for the permission implies that **fail** is also permitted, which can be seen from the following:

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

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).

4.2 Permission for conjunction

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

Proposition 3

$$P(\alpha) \longrightarrow P(\alpha \& \beta)$$

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

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

Given our interpretation of $P(\alpha)$ 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:

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

Given our strong interpretation of the permission, using the prudencial assumption of Grice about the maxims of communication, what we actually mean by:

It is permitted to buy a gun

is more appropriately represented as:

$$P(only(buy\ a\ gun))$$

This permission does still imply permissions like

$$\begin{array}{l}
P(only(buy\ a\ gun)\ \&\ shoot\ the\ president) \\
P(only(buy\ a\ gun\ \&\ shoot\ the\ president))\ \ or \\
P(only(buy\ a\ gun)\ \&\ only(shoot\ the\ president))
\end{array}$$

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

So, in general, we do have:

$$P(\alpha) \longrightarrow P(\alpha \& \beta)$$

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

$$P(only(\alpha)) \longrightarrow P(only(\alpha) \& only(\beta))$$

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).

4.3 Complementarity of permission and prohibition

The third issue that is raised by the definition is that the permission is no longer complementary to the prohibition. I.e. $P(\alpha) \neq \neg F(\alpha)$. With the new definition for the permission there can be actions α such that $\neg P(\alpha) \wedge \neg F(\alpha)$.
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 4 *Let* $\alpha \neq_{\mathcal{D}}$ **fail** *then:*

$$P(\alpha) \longrightarrow \neg F(\alpha)$$

or equivalently

$$F(\alpha) \longrightarrow \neg P(\alpha)$$

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

The fact that permission and prohibition are no longer complementary in our system also influences the following implication that holds in all the deontic systems that are defined in terms of dynamic logic like is done in [Mey88].

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

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

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

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

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

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

4.4 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 actions 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 can only be applied to actions in strict disjunctive normal form, which is a

big restriction on its use. Also the operator does not distribute over any of the other connectors of actions, which also limits its use. However, in practice, the $only$ operator is mainly applied to atomic actions and their conjunctions. Therefore, these restrictions on the use of the $only$ operator do not have very severe consequences.

The second point is, in practice, maybe more serious. Using the traditional definitions, $P(\alpha)$ means that α is permitted, possibly in combination with other actions. It is not necessary to explicitly state all the combinations that are allowed. Using $P(only(\alpha))$ is much stricter in the sense that it only allows 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 $P(\alpha)$.
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).
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.

5 Contexts

In this section, we will introduce contexts for actions. This is a bit less strict then the $only$ operator. The intuitive idea is that with an action α we specify a set of events $\mathcal{C}$ relative to which the semantics of the action 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 some semantics for them.

5.1 Contextual actions

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

Definition 11 Let $\alpha \in Act$ then $at(\alpha)$ is the set of events such that the underlined version of that event occurs in α.

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

The set of contextual actions $Act_{\mathcal{C}}$ can now be defined as follows:

Definition 12 Let $\mathcal{C} \subseteq \mathcal{A}$. $Act_{\mathcal{C}}$ is the minimally closed set such that:

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

The set $\mathcal{C}$ is called the context of the action.
Notation: Because the action α plays the central role in the action expressions, we will write the context of an action not as an operator on the action but as a subscript. I.e. we write $\alpha_{\mathcal{C}}$ for $\mathcal{C}(\alpha)$.
If the context $\mathcal{C}$ of an action α is $\mathcal{A}$ then we write α as a shorthand for $\alpha_{\mathcal{A}}$.

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

Definition 13 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 actions is now given in the following definition.

Definition 14 The semantic function $[\![]\!] \in Act_{\mathcal{C}} \rightarrow \mathcal{D}$ is given by:

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

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 "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 actions have the standard interpretation.

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

Proposition 5 *Let $\alpha_{\mathcal{C}} \in Act_{\mathcal{C}}$ such that it does not contain any negation and $\mathcal{C} = at(\alpha)$ then*

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

Proof: Easy by induction.□

The above proposition can be read as "$only(\alpha)$ is α done in the context of its own atomic actions".

In the above proposition we excluded actions that contain negations. That the equality does not hold for these actions can be seen from the following simple example:

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

while

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

After this introduction of contexts for actions, in the next section we will show how they can be related to the deontic operators.

5.2 Safe contexts

The intuitive meaning of a (deontically) safe context for α is that α is performed possibly together with some other actions that have no influence on the effects of α.
I.e. $(open\ the\ window)_{\mathcal{C}}$ where $\mathcal{C}$ is a safe context may 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. More formally:

Definition 15 Let Φ be a set of propositions closed under deduction and containing $\neg Violation$, then we call Φ a **deontic propositional context**.

The formal definition of a safe context can be given as follows:

Definition 16 Let Φ be a deontic propositional context and let $\alpha_{\mathcal{C}} \in Act_{\mathcal{C}}$ then $\mathcal{C}$ is a **safe** context for α in w with respect to Φ iff for all $\phi \in \Phi$

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

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

Proposition 6 *If C is* **safe** *for α with respect to a deontic propositional context Φ then*

$$P(only(\alpha)) \longrightarrow P(\alpha_{\mathcal{C}})$$
$$F(only(\alpha)) \longrightarrow F(\alpha_{\mathcal{C}})$$

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

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

Therefore we have that:

$$P(only(fire-gun)) \rightarrow P(fire-gun_{\{fire-gun,\ chew-gum\}})$$

but **not**

$$P(only(fire-gun)) \rightarrow P(fire-gun_{\{fire-gun,\ aim-at-president\}})$$

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

Proposition 7 *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 actions.
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 8 *Let C be a* **safe** *context for α then*

$$P(\alpha_{\mathcal{C}}) \wedge F(\beta) \longrightarrow \alpha_{\mathcal{C}} \& \beta =_{\mathcal{D}}$$ **fail**

Which means that if $\alpha_{\mathcal{C}}$ is 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 α is weakened to a permission to perform α only in some (safe) context of other actions, the above implication still holds.
The consequence is that we do have the following (if $\mathcal{C}$ is a safe context):

$$P(chew\ gum_{\mathcal{C}}) \longrightarrow chew\ gum_{\mathcal{C}} \& shoot\ the\ president =_{\mathcal{D}} \mathbf{fail}$$

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. the combination $\underline{a}\&\underline{b}$ might include the effects from $only(\underline{a})$ while it does not include the effects of $only(\underline{b})$. This happens, for instance, if $\underline{a}$ is permitted in isolation and also in combination with $\underline{b}$, but $\underline{b}$ is forbidden in isolation. More concrete this can be seen 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.

6 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 $P(\alpha) \rightarrow P(\alpha \cup \beta)$ for actions 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

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

omits something, viz. the admissibility of β. If we add this information, the inference is blocked:

$$(P(\alpha \cup \beta) \wedge \neg P(\beta)) \leftrightarrow ((P(\alpha) \vee P(\beta)) \wedge \neg P(\beta)) \rightarrow P(\alpha)$$

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. This is the idea that led to our stringent form of permission.

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. 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 actions. 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 any action 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 surprising 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 workgroup Logic and Law for their input. We mention esspecially Hans Weigand for pointing out a severe fallacy in the paper. Also we are greatly indebted to the anonymous referees for their stimulating remarks.

References

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

[Bro86] M. Broy. A theory for nondeterminism, parallelism, communication and concurrency. *Theoretical Computer Science*, (45):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.

[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.

[Hil] R. Hilpinen. Conditionals in Possible Worlds in G. Fløstad, editor, *Contemporary Philosophy, a New Survey*, Vol.1,pages 299-335, Reidel.

[Kam74] H. Kamp Free Choice Permission. *Aristotelian Society Proceedings N.S.* (74):57-74, 1974.

[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*, (29):109–136, 1988.

[Mey90] J.-J.Ch. Meyer. Using programming concepts in deontic reasoning. In R. Bartsch, J. van Benthem, and P. van EmdeBoas, editors, *Semantics and Contextual Expressions*. Foris, Dordrecht, 1990.

[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, ILLC, University of Amsterdam, 1992.

[MW92] J.-J.Ch. Meyer and R.J. Wieringa. Actors, Actions and Initiative in Normative System Specification. In *Annals of Mathematics and Artificial Intelligence*,Vol 7, pages 289-346, 1993.

[WMW89] R. Wieringa, J.-J.Ch. Meyer, and H. Weigand. Specifying dynamic and deontic integrity constraints in knowledge bases. *Data & Knowledge Engineering*, 4(4), 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.

A

In this appendix we give the formal semantics of actions as we use it in this paper. It is mainly a simplified (restricted to actions) version of the semantics given in [Mey88].

A.1 Formal definition of the semantics of actions

We start with a set of **events** $\mathcal{A}$, with typical elements $a, b, c....$ There is one special event δ, which is not an element of $\mathcal{A}$, and which stands for failure.(comparable to deadlock in process algebra). The set of events $\mathcal{A}$ will form the basis for the semantics of the actions in our language *Act.*
The atomic **actions** in these languages are denoted by underlined versions of events from $\mathcal{A}$. Thus if $a \in \mathcal{A}$ then $\underline{a}$ is an atomic action in *Act.* The set of all action expressions can now be determined by the following BNF for its elements (α):

$$\alpha :: - \ \underline{a} | \alpha_1 \underline{\cup} \alpha_2 | \alpha_1 \& \alpha_2 | \overline{\alpha} | \mathbf{any} | \mathbf{fail}$$

The meaning of $\alpha_1 \underline{\cup} \alpha_2$ is a choice between α_1 and α_2. $\alpha_1 \& \alpha_2$ stands for the parallel execution of α_1 and α_2. The expression $\overline{\alpha}$ stands for the negation of the transaction α. The **any** action is a universal or "don't care which" action. Finally the **fail** action is the action that always fails (deadlock). After this action nothing can be done anymore and the system stops.
It is important to note the difference between the syntactical atomic actions $\underline{a}$ and the semantical events a. The meaning of an atomic action $\underline{a}$ involves the execution of the corresponding semantical event a possibly together with other events! The meaning of $\underline{a}$ only specifies the performance of the corresponding semantical a, but one is free to perform any other set of events simultaneous with a!

The semantics of the actions consists of two parts. First we interpret the actions in terms of (sets of) events. This part is called the *uniform* semantics in the literature on the semantics of concurrency (e.g. [dBKM+86]). The second part of the semantics is the interpretation of the denotations of actions as state transforming functions. This yields

the non-uniform semantics. We will now start with the uniform part of the semantics.

The semantics of actions are given by sets of what we call *synchronicity* sets (or s-sets for short). A synchronicity set denotes a set of events that are executed simultaneously. They are the basic building blocks of the semantics.
The following definitions formally describe these sets and the ways they can be combined to form the semantic counterpart of actions.

Definition 17

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

Notation: We use $S, S_1, \ldots, S', \ldots$ do denote s-sets. 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 s-sets will be denoted by $\wp^+(\mathcal{A})$, where the '+' indicates that the s-sets are non-empty.

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.

Because the language of actions contains a choice operator, which introduces non-determinism, we have to consider sets of s-sequences as the semantics of a transaction. Each of the s-sequences in these sets stands for a possible choice. This non-determinism is also introduced by using the open specification of an action. That is, indicating that $\underline{a}$ is performed has as semantics that the event a is performed alone or simultaneous with any other set of events.
Notation: We use $T, T_1, \ldots, T', \ldots$ to denote sets of s-sets.

Having introduced the basic elements, we can now proceed to give the formal definition of the semantic domain of the actions with its operators.

Definition 18 The domain $\mathcal{D}$ for our model for actions from *Act* is the collection of sets T consisting of s-sets.

To give the denotation for all actions in *Act* we define the semantical counterparts of the syntactical operators $\underline{\cup}$, & and negation. Before we give these definitions, we define a handy operator on sets of s-sets:

Definition 19 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 $\cap$, which is almost the same as the normal set-intersection.

Definition 20 For $T, T' \in \mathcal{D}$:

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

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

Definition 21 For $T, T' \in \mathcal{D}$:

$$T \,\underline{\cup}\, T' = (T \cup T')^{\delta}$$

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.

Definition 22 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 $T \in \mathcal{D}$

$$T^{\sim} = \bigcap \{S^{\sim} | S \in T\}$$

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.

With these definitions, we can now define the semantics of actions in *Act*.

Definition 23 The semantic function $[\![]\!] \in Act \rightarrow \mathcal{D}$ is given by:

$$\begin{array}{l} [\![\underline{a}]\!] = \{S \in \wp^+(\mathcal{A}) | a \in S\} \\ [\![\alpha_1 \underline{\cup} \alpha_2]\!] = [\![\alpha_1]\!] \bigcup [\![\alpha_2]\!] \\ [\![\alpha_1 \& \alpha_2]\!] = [\![\alpha_1]\!] \bigcap [\![\alpha_2]\!] \\ [\![\overline{\alpha}]\!] = [\![\alpha]\!]^{\sim} \\ [\![\textbf{fail}]\!] = \{[\delta]\} \\ [\![\textbf{any}]\!] = \wp^+(\mathcal{A}) \end{array}$$

From the definitions above, it follows that the semantics of the negation of the atomic action $\underline{a}$ consists of all the s-sets that do not contain the event a. Thus the meaning of $\overline{\underline{a}}$ is the choice of doing any set of events (simultaneously) as long as they do not include a.

In the following definition we define the (in)equality of actions in terms of their semantics.

Definition 24 We define $\alpha_1 =_{\mathcal{D}} \alpha_2$ iff [[α_1]]= [[α_2]]. We define $\alpha_1 > \alpha_2$ iff [[α_1]]$\subseteq$ [[α_2]].

If $\alpha_1 > \alpha_2$ then we say that α_1 *involves* α_2.

Proposition 9

1. $(\mathcal{D}, \underline{\cup}, \&, \neg, \textbf{fail})$ *is a boolean algebra.*

2. $\mathcal{D}$ *satisfies the following property concerning the special actions:*

$$\overline{\textbf{fail}} =_{\mathcal{D}} \textbf{any}$$

Proof: Clear by the proposition.□.

A.2 Actions and worlds

In this section we will relate the actions of *Act* to worlds in which they are performed. Intuitively, the execution of an action in a world yields a collection of worlds which one might reach by performing the action.

First we introduce a function ϕ. For a given set of worlds W this function is an element of $\wp^+(\mathcal{A}) \cup [\delta] \rightarrow (W \rightarrow \wp_1(W))$. Here $\wp_1(W)$ stands for the sets of elements of W that consists of at most one element. The function determines the behaviour of the s-sets, i.e. the sets of events that are executed simultaneously. We thus assume that each s-set has a deterministic behaviour. The function ϕ is left unspecified except for its behaviour on the s-set $[\delta]$. The behaviour of the function ϕ on other s-sets depends on the events, which are in their turn dependent on the application. A way to specify the behaviour of ϕ in terms of the changes that are made by performing the set of events in an s-set is given in [Dig89]. Taking the function ϕ as a basis, we then define a function F that determines the behaviour of sets of s-sets.

Definition 25

1. $\phi([\delta])(w) = \emptyset$

2. The function $F(T) \in \wp^+(W) \rightarrow \wp^+(W)$ is defined inductively by:

$F(S)(W_0) = \bigcup_{w \in W_0} \{\phi(S)(w)\}$ for $S \in \wp^+(\mathcal{A}) \cup [\delta]$ and $W_0 \subset W$
$F(T)(W_0) = \bigcup_{S \in T} F(S)(W_0)$

The semantics of actions in a certain world can now be given by the function $[\![.]\!]_F$ which is an element of $Act \rightarrow (W \rightarrow \wp^+(W))$ and which is defined as follows:

Definition 26

$$[\![\alpha]\!]_F(w) = F([\![\alpha]\!])(\{w\})$$