Unravelling Nondeterminism:
On Having the Ability to Choose*

W. van der Hoek
B. van Linder
J.-J. Ch. Meyer†
Utrecht University
Department of Computer Science
P.O. Box 80.089, 3508 TB Utrecht
The Netherlands
Email: bernd@cs.ruu.nl

Abstract

We demonstrate ways to incorporate nondeterminism in a system designed to formalize the reasoning of agents concerning their abilities and the results of the actions that they may perform. We distinguish between two kinds of nondeterministic choice operators: one that expresses an internal choice, in which the agent decides what action to take, and one that expresses an external choice, which cannot be influenced by the agent. The presence of abilities in our system is the reason why the usual approaches towards nondeterminism cannot be used here. The semantics that we define for nondeterministic actions is based on the idea that composite actions are unravelled in the strings of atomic actions and tests that constitute them. The main notions used in defining this semantics are finite computation sequences and finite computation runs of actions. The results that we obtain meet our intuitions regarding events and abilities in the presence of nondeterminism.

1. Introduction

To investigate the reasoning of rational agents regarding correctness and feasibility of their plans we defined in [HLM93] a formal system, designed to deal with both the knowledge and abilities of agents, and the effects of the actions performed by the agent. Based on the concepts given in [Moo80] and [Moo84], the system of [HLM93] is firmly rooted in both epistemic logic (see [MH]) and dynamic logic (see [Har79], [Har84], [KT90], and [Gol92]). The actions that we considered in [HLM93] are deterministic: the event that consists of some agent performing some action has a unique outcome. In this paper we will consider a natural extension of the system of [HLM93], given by the introduction of nondeterministic

*This research is partially supported by ESPRIT III BRA project No.6156 'DRUMS II' and the Vrije Universiteit Amsterdam.
†This author is partially supported by the Katholieke Universiteit Nijmegen.

action constructors. These constructors combine two actions into a new one: the nondeterministic choice between the two actions. Although we introduce nondeterminism of composite actions, we will assume throughout this paper that the atomic actions still have a unique outcome. Rationale behind this assumption is our conviction that determinism is, together with unspecifiedness, one of the main characterizations of atomicity.

In this first attempt to formalize the different sorts of nondeterministic operators, we have chosen to deal with them differently and in separate ways. Nevertheless combining these operators certainly needs further consideration.

The contents of the rest of this paper is as follows. In section 2 we introduce some of our ideas concerning events and abilities that underlay the formal system of [HLM93]. Furthermore the syntax and semantics of the system of [HLM93], which will be used as a basis to build nondeterminism upon, are (re)introduced. In section 3 the intuition behind our approach towards nondeterminism is explained. Furthermore we compare our ideas with those of two other systems. In section 4 it is shown on the basis of two examples why the usual approaches towards nondeterminism are not suitable for our goals. In section 5 we will show how to incorporate internal nondeterminism into the system given in section 2. Section 6 contains the definitions used to incorporate external nondeterminism. Section 7 ends this paper with some conclusions and a guideline for further research.

2. Knowledge, events, and abilities: the basic definitions

In [HLM93] a formal system is given in which, in contrast with the usual approaches in dynamic logic, the abilities of agents are considered in their own rights. The usual approach in dynamic logic when abilities are considered is to identify the ability of an agent i for an action α with truth of the formula $<\mathrm{do}_i(\alpha)>\mathbf{tt}$ (cf. for instance [Tho93]); the latter formula intuitively captures the idea that the action α is possible for agent i. In our opinion this approach does no justice to the intuitive meaning of abilities.

2.1. Example (The shampoo example). Consider an agent planning to wash her hair. Since there is shampoo available, $<\mathrm{do}_i(\mathrm{wash\_hair})>\mathrm{hair\_clean}$ holds, which in the usual approaches towards abilities would suffice to conclude that the agent is capable of washing her hair. The moment the last drip of shampoo leaves the bottle, $<\mathrm{do}_i(\mathrm{wash\_hair})>\mathrm{hair\_clean}$ no longer holds, and hence the agent is no longer capable of washing her hair. But this is intuitively strange: the ability of the agent seems to be something that is a property of the agent rather than a context dependent notion that depends on the presence of shampoo.

In our opinion abilities are to be considered as an additional concept, defined independently from the diamond formulae $<\mathrm{do}_i(\alpha)>\varphi$. We use the formula $<\mathrm{do}_i(\alpha)>\varphi$ to indicate that all mundane, action specific, prerequisites of α are satisfied and that as a result of executing α, φ holds. In terms of example 2.1, if there is shampoo available, washing your hair leads to your hair being clean. The formula $\mathbf{A}_i\alpha$ denotes the fact that the action α is an element of the abilities of the agent i: the physical (mental, moral) condition of agent i is such that s/he is capable of α. In terms of the example 2.1, the agent is capable of

washing his/her hair (s/he has learned how to do it, s/he is not handicapped), regardless of whether there is shampoo available. As such the combination of $<\mathrm{do}_i(\alpha)>\varphi$ and $\mathbf{A}_i\alpha$ can be used to capture the idea of α being a *correct* ($<\mathrm{do}_i(\alpha)>\varphi$) and a *feasible* ($\mathbf{A}_i\alpha$) plan for agent i to achieve φ (see also sections 5.2 and 6.2).

Next the definitions of the system defined in [HLM93] are given. Although this deterministic system serves as a basis to build our treatment of nondeterminism upon, the way in which we treat nondeterminism is dependent only on the class of actions under consideration: the approach as we give it can be applied to any dynamic logic like system (as can be found in [Har84], [HR83], [KT90], and [Gol92]).

2.2. DEFINITION. The language $\mathcal{L}$ is based on a given set Π of propositional symbols, a finite set At of atomic actions and a finite set $\mathcal{A} = \{1, \ldots, n\}$ of agents. The language $\mathcal{L}$ is defined by the following BNF, where φ is a typical element of the set of formulae in $\mathcal{L}$, p is a typical element of the set of propositional symbols Π, α is a typical element of the set of actions Ac, a is a typical element of the set of atomic actions At, and i is a typical element of the set of agents $\mathcal{A}$.

$$\varphi ::= p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \mathbf{K}_i\varphi \mid\ <\mathrm{do}_i(\alpha)>\varphi \mid \mathbf{A}_i\alpha$$

where the class Ac of actions is given by:

$\alpha ::= a$	$\mid$	*atomic actions*
$\texttt{confirm}\ \varphi$	$\mid$	*confirmations*
$\alpha_1; \alpha_2$	$\mid$	*sequential composition*
$\texttt{if}\ \varphi\ \texttt{then}\ \alpha_1\ \texttt{else}\ \alpha_2\ \texttt{fi}$	$\mid$	*conditional composition*
$\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}$		*repetitive composition*

The constructs $\wedge$, $\rightarrow$, and $\leftrightarrow$ are defined in the usual way. Other additional constructs are introduced by definitional abbreviation:

$\mathbf{tt}$	is	$p \vee \neg p$
$\mathbf{ff}$	is	$\neg\mathbf{tt}$
$[\mathrm{do}_i(\alpha)]\varphi$	is	$\neg <\mathrm{do}_i(\alpha)> \neg\varphi$
$\texttt{skip}$	is	$\texttt{confirm}\ \mathbf{tt}$
$\texttt{fail}$	is	$\texttt{confirm}\ \mathbf{ff}$
α^0	is	$\texttt{skip}$
α^{n+1}	is	$\alpha; \alpha^n$

2.3. REMARK. To provide for optimal flexibility no demand whatsoever is made upon the use of parentheses in $\mathcal{L}$: whenever one thinks that using parentheses provides for more clarity one is encouraged to use them. Furthermore for reasons of practical convenience the sequential composition operator ; is assumed to be right associative, i.e., $\alpha_1; \alpha_2; \alpha_3$ should be read as $\alpha_1; (\alpha_2; \alpha_3)$.

2.4. REMARK. The knowledge of agents, formalized by the $\mathbf{K}_i$ operator, is mainly important for the Can-predicate and the Cannot-predicate. These predicates formalize the knowledge of agents regarding the (in)correctness and (in)feasibility of their plans (see also sections 5.2 and 6.2).

The semantics is based on the use of Kripke models.

2.5. DEFINITION. The class $\mathfrak{M}$ of Kripke models contains all tuples $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$ such that

(1) $\mathcal{S}$ is a set of possible worlds, or states.

(2) $\pi : \Pi \times \mathcal{S} \to \mathbf{bool}$ is a total function that assigns a truth value to propositional symbols in possible worlds.

(3) $\mathrm{R} : \mathcal{A} \to \wp(\mathcal{S} \times \mathcal{S})$ is a function that yields the epistemic accessibility relation for a given agent. Since we will assume **S5** to axiomatize the knowledge of agents, it is demanded that $\mathrm{R}(i)$ is an equivalence relation (cf. [HM85], [MH]).

(4) $\mathtt{r} : \mathcal{A} \times At \to \mathcal{S} \to \wp(\mathcal{S})$ is such that $\mathtt{r}(i,a)(s)$ yields the (possibly empty) state transition in s caused by the event $\mathrm{do}_i(a)$. This function is such that for all atomic actions a it holds that $\forall i \forall s[|\mathtt{r}(i,a)(s)| \leq 1]$, where $|V|$ denotes the number of elements of the set V.

(5) $\mathtt{c} : \mathcal{A} \times At \to \mathcal{S} \to \mathbf{bool}$ is the capability function such that $\mathtt{c}(i,a)(s)$ indicates whether the agent i is capable of performing the action a in the possible world s.

The models from $\mathfrak{M}$ are called *standard models*.

2.6. DEFINITION. Let $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$ be some standard model. The functions π, $\mathtt{r}$, and $\mathtt{c}$ are extended by simultaneous induction as follows.

$$
\begin{array}{ll}
\pi & : \ \mathcal{L} \times \mathcal{S} \to \mathbf{bool} \\
\pi(\neg\varphi, s) & = \neg\pi(\varphi, s) \\
\pi(\varphi \vee \psi, s) & = \pi(\varphi, s) \vee \pi(\psi, s) \\
\pi(\mathbf{K}_i\varphi, s) & = \mathbb{A}_{(s,s') \in \mathrm{R}(i)} \pi(\varphi, s') \\
\pi(<\mathrm{do}_i(\alpha)>\varphi, s) & = \mathbb{V}_{s' \in \mathtt{r}(i,\alpha)(s)} \pi(\varphi, s') \\
\pi(\mathbf{A}_i\alpha, s) & = \mathtt{c}(i,\alpha)(s) \\
 & \\
\mathtt{r} & : \ \mathcal{A} \times Ac \to \mathcal{S} \to \wp(\mathcal{S}) \\
\mathtt{r}(i, \mathtt{confirm}\ \varphi)(s) & = \{s\} \text{ if } \pi(\varphi, s) = \mathbf{1} \\
 & = \emptyset \ \text{ otherwise} \\
\mathtt{r}(i, \alpha_1; \alpha_2)(s) & = \mathtt{r}(i, \alpha_2)(\mathtt{r}(i, \alpha_1)(s)) \\
\mathtt{r}(i, \mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi})(s) & = \mathtt{r}(i, \alpha_1)(s) \text{ if } \pi(\varphi, s) = \mathbf{1} \\
 & = \mathtt{r}(i, \alpha_2)(s) \text{ otherwise} \\
\mathtt{r}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od})(s) & = \{s' \mid \exists k \in \mathbb{N} \exists s_0 \ldots \exists s_k \in \mathcal{S}[s_0 = s \wedge s_k = s' \wedge \\
 & \qquad \pi(\neg\varphi, s') = \mathbf{1}] \wedge \\
 & \qquad \forall j < k[s_{j+1} \in \mathtt{r}(i, \mathtt{confirm}\ \varphi; \alpha_1)(s_j)]\}
\end{array}
$$

where

$$
\begin{array}{ll}
\mathtt{r}(i,\alpha)(\mathcal{S}') & = \cup_{s'\in\mathcal{S}'}\mathtt{r}(i,\alpha)(s') \text{ for } \mathcal{S}'\subseteq\mathcal{S}. \\
 & \\
\mathtt{c} & : \ \mathcal{A}\times Ac\to\mathcal{S}\to\mathbf{bool} \\
\mathtt{c}(i,\mathtt{confirm}\,\varphi)(s) & = \pi(\varphi,s) \\
\mathtt{c}(i,\alpha_1;\alpha_2)(s) & = \mathtt{c}(i,\alpha_1)(s)\wedge\mathtt{c}(i,\alpha_2)(\mathtt{r}(i,\alpha_1)(s)) \\
\mathtt{c}(i,\mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi})(s) & = \mathtt{c}(i,\mathtt{confirm}\,\varphi;\alpha_1)(s)\vee \\
 & \quad\ \mathtt{c}(i,\mathtt{confirm}\,\neg\varphi;\alpha_2)(s) \\
\mathtt{c}(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od})(s) & = \mathbf{1} \text{ if } \exists k\in\mathbb{N}[\mathtt{c}(i,(\mathtt{confirm}\,\varphi;\alpha_1)^k; \\
 & \qquad\qquad\qquad\qquad\qquad\qquad \mathtt{confirm}\,\neg\varphi)(s)=1] \\
 & = \mathbf{0} \text{ otherwise}
\end{array}
$$

where

$$
\mathtt{c}(i,\alpha)(\mathcal{S}') = \mathbb{\wedge}_{s'\in\mathcal{S}'}\mathtt{c}(i,\alpha)(s') \text{ for } \mathcal{S}'\subseteq\mathcal{S}.
$$

2.7. REMARK. The motivation for the choices made in definition 2.6 regarding the abilities of agents is the following. An agent is capable of performing a sequential composition $\alpha_1;\alpha_2$ iff s/he is capable of α_1 and s/he is capable of performing α_2 after s/he has performed α_1. The definition of $\mathtt{c}(i,\mathtt{confirm}\,\varphi)(s)$ is based on the idea that a test for φ is actually an action that searches for confirmation of φ; if it is not possible to get confirmation for φ, the agent is not capable of testing φ. This is similar to the approach towards tests as taken in dynamic logic.
An agent is capable of performing a conditional composition, iff s/he is able to test the condition and thereafter s/he is capable of performing either the then-part or the else-part, dependent on whether the condition or the negation of the condition is confirmed.
Lastly, an agent is capable of performing a repetitive composition $\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}$ iff s/he is able to perform the action $(\mathtt{confirm}\,\varphi;\alpha_1)^k;\mathtt{confirm}\,\neg\varphi$ for some $k\in\mathbb{N}$. Note that the definitions for both the conditional and the repetitive composition are based on the defining equalities that hold in dynamic logic (see [Har84]).

2.8. REMARK. Singleton sets $\{s\}$ with $s\in\mathcal{S}$ are usually denoted by s.

2.9. DEFINITION. For all standard models $\mathcal{M}=<\mathcal{S},\pi,\mathrm{R},\mathtt{r},\mathtt{c}>$, for all possible worlds $s\in\mathcal{S}$, and for all formulae φ we define:

- $\mathcal{M},s\models\varphi$ iff $\pi(\varphi,s)=\mathbf{1}$.
- The formula φ is valid in $\mathcal{M}$, notation $\mathcal{M}\models\varphi$, iff $\mathcal{M},s\models\varphi$ for all $s\in\mathcal{S}$.

2.10. DEFINITION. For all formulae φ we define:

- The formula φ is satisfiable in $\mathfrak{M}$ if some Kripke model $\mathcal{M}\in\mathfrak{M}$ and $s\in\mathcal{S}$ exist such that $\mathcal{M},s\models\varphi$.
- The formula φ is valid in $\mathfrak{M}$, notation $\models_{\mathfrak{M}}\varphi$ or simply $\models\varphi$, iff $\mathcal{M}\models\varphi$ for all $\mathcal{M}\in\mathfrak{M}$.

2.11. DEFINITION. Let $\mathcal{M}=<\mathcal{S},\pi,\mathrm{R},\mathtt{r},\mathtt{c}>$ be some Kripke model and let α be some action.

- The event $\mathrm{do}_i(\alpha)$ is *deterministic* in some possible world s from $\mathcal{S}$ if and only if

  $$|\mathtt{r}(i,\alpha)(s)| \leq 1$$

- The event $\mathrm{do}_i(\alpha)$ is deterministic for $\mathcal{M}$ if and only if $\mathrm{do}_i(\alpha)$ is deterministic in all s from $\mathcal{S}$.

The framework given in this section is indeed a deterministic one:

2.12. LEMMA. *All events* $\mathrm{do}_i(\alpha)$*, where* $\alpha \in Ac$ *as given in definition 2.2, are deterministic for all* $\mathcal{M} \in \mathfrak{M}$*.*

PROOF OF LEMMA 2.12: See [HLM93].
⊠

3. Internal versus external nondeterminism

Intuitively the meaning of a nondeterministic choice between the actions α_1 and α_2 is given by: 'choose one of α_1 and α_2, and perform the action that is chosen.' As observed in [Hoa85], this intuitive description gives rise to two different implementations. Given the situation that the actor performing the actions is surrounded by, and strictly separated from, some external environment, it follows that in principle both the actor and the external environment could make the choice of what action to perform. Depending on who makes the choice, two different nondeterministic actions result.
The situation as we consider it in this paper is as sketched above: a group of agents, reasoning about their abilities and the effects of their actions, is surrounded by some unspecified external environment.

Following [Mey92], the action in which the nondeterministic choice is made by the agent itself is called *internal*; if the external environment makes the choice, the action is called *external*. The notation used in this paper is also conform [Mey92]: the internal nondeterministic choice between the actions α_1 and α_2 is denoted by $\alpha_1 \oplus \alpha_2$, the external nondeterministic choice is denoted by $\alpha_1 + \alpha_2$.

An important difference between the system defined in this paper, and the systems of [Mey92] and [WM91], in which also internal and external nondeterminism are considered, is given by the fact that in our system the agents themselves are reasoning with and about events and abilities. This implies that the semantic definitions of the internal and external nondeterministic choice should be such that they correctly formalize the reasoning of agents concerning their abilities and the effects of their actions when nondeterministic choices are involved. In the other systems reasoning is only done at a meta level, by the observers. This point is elaborated on in section 3.1.

Like in [Mey92], and unlike [WM91], the actual act of 'choosing' is not considered in this paper. Since we will introduce internal and external nondeterminism separately, and hence combinations of these actions do not occur, it seems fairly reasonable to ignore the 'choose'

action: it is simply decided on beforehand whether the agent or the external environment makes the choice for all nondeterministic actions.

As explained in section 2, the results of events are formalized using the diamond formula $< \mathrm{do}_i(\alpha) > \varphi$. In the nondeterministic framework that we are going to define, truth of $<\mathrm{do}_i(\alpha)>\varphi$ is intuitively taken to imply that at least one way of performing α is open to agent i such that φ follows as a result. The dual notion $[\mathrm{do}_i(\alpha)]\varphi$ implies that all possible ways open to agent i to perform α lead to φ. The abilities of the agents are, cf. section 2, formalized using the formula $\mathbf{A}_i\alpha$.

Now assume that some agent i is reasoning whether φ holds as a result of performing the internal nondeterministic choice $\alpha = \alpha_1 \oplus \alpha_2$. The agent could be reasoning as follows:

> ' Since I myself make the choice as of whether to perform α_1 or α_2, in order to conclude that φ holds as a result of performing α it suffices that for one of α_1 and α_2 a way of executing exists that leads to φ. This is also a necessary condition: if for neither of α_1 and α_2 a way of executing exists that leads to φ, it is not possible that the internal nondeterministic choice between them would lead to φ.'

For the reasoning of the agent concerning his/her abilities for an internal nondeterministic choice $\alpha = \alpha_1 \oplus \alpha_2$ an analogous line of reasoning can be given:

> ' Since I make the choice myself, in order to conclude that I am able to perform α it suffices that I am able to perform one of the actions constituting α. It is also necessary that I am capable of performing one of the actions, since otherwise it is not possible that I am able to perform α.'

The observation given above for the internal nondeterministic choice expresses the idea that the agent thinks of her/himself as showing an *angelic* (cf. [Bro86]) behaviour: whenever it is possible to perform an action in the correct, desired way, and the agent may choose how to perform the action, s/he will choose to perform the action in this correct way.

In case of an external nondeterministic choice $\alpha = \alpha_1 + \alpha_2$ the agent has no influence on which action is chosen and therefore s/he must be prepared to deal with either of the actions. In particular, the agent may not assume that the external environment will make the choice that is best for the agent, i.e., the agent should take into account that the external environment shows a *demonic* (also cf. [Bro86]) behaviour. Now suppose that agent i is reasoning whether φ holds as a result of performing $\alpha = \alpha_1 + \alpha_2$. The reasoning of the agent could proceed as follows:

> ' Since I have no influence whatsoever on which action is going to be chosen, it is possible that performing $\alpha_1 + \alpha_2$ leads to φ, if and only if for both of α_1 and α_2 a way of executing exists that leads to φ. After all I must be prepared to deal with either one of them.'

For the reasoning of the agent concerning his/her abilities for an external nondeterministic choice $\alpha = \alpha_1 + \alpha_2$ an analogous line of reasoning can be given:

> ' Since the external environment makes the choice between α_1 and α_2, it is possible that I am capable of performing $\alpha_1 + \alpha_2$ if and only if I am able to perform both of the actions.'

The semantics for the internal and the external nondeterministic choice, given in section 5 and section 6 respectively, are based on the observations given above.

3.1. REMARK. Note that the choice as of agents showing an angelic, and the external environment showing a demonic behaviour is a somewhat arbitrary one. In fact the definitions that we give are such that the opposite situation, or any of the four possible situations, can equally well be formalized.

3.1. A comparison with some other approaches

As far as we know no formal systems exist that treat abilities in a way similar to ours, let alone systems that consider abilities for nondeterministic actions. Nevertheless some systems exist that for events either distinguish the two sorts of nondeterminism as discussed at the beginning of section 3, or deal with the related notion of concurrency. We will briefly consider two such systems: one is the system of [Mey92], in which internal and external nondeterminism are considered for events, the other one is the system of [Pel87] that deals with concurrency.

The system of [Mey92] is based on the following two equivalences:

- $<\mathrm{do}_i(\alpha_1 \oplus \alpha_2)>\varphi \leftrightarrow (<\mathrm{do}_i(\alpha_1)>\varphi \wedge <\mathrm{do}_i(\alpha_2)>\varphi)$
- $<\mathrm{do}_i(\alpha_1 + \alpha_2)>\varphi \leftrightarrow (<\mathrm{do}_i(\alpha_1)>\varphi \vee <\mathrm{do}_i(\alpha_2)>\varphi)$

It turns out that Meyer's starting point is exactly the opposite of ours! This difference is caused by the fact that in Meyer's system an external observer is reasoning about the effect of the actions performed by some agent/actor. The external observer has the implicit assumption that the agent shows a demonic behaviour, and that the external environment shows an angelic one. Thus the approach of Meyer is not in contradiction with ours: a shift in perspective causes these two approaches to behave differently.

In [Pel87] Concurrent Dynamic Logic (CDL) is defined. In CDL the operator $\cap$ is used to combine two actions α_1 and α_2 into a new action, with intuitive meaning 'α_1 and α_2 in parallel'. The operator $\cap$ is such that the following is a valid formula:

- $<\mathrm{do}_i(\alpha_1 \cap \alpha_2)>\varphi \leftrightarrow (<\mathrm{do}_i(\alpha_1)>\varphi \wedge <\mathrm{do}_i(\alpha_2)>\varphi)$

Peleg's approach is based on the idea that concurrency and nondeterminism (the kind of nondeterminism that we would call *internal*) are dual notions. Hence it is obvious that Peleg's concurrency and our external nondeterminism (note that the latter is the dual of internal nondeterminism) are analogous notions. Also intuitively this correspondence

seems to be correct: although in case of an external nondeterministic choice $\alpha_1 + \alpha_2$ only one of α_1 and α_2 is performed, the agent has to be prepared to deal with execution of both, and hence reasons about the effect of this action as if both actions were actually executed.

Although abilities are not considered in CDL it seems that our treatment of the external nondeterministic choice is intuitively also a correct one for concurrency: an agent is able to perform the actions α_1 and α_2 in parallel iff both actions belong to the abilities of the agents. Hence our treatment of the external nondeterministic choice might as well be taken to be one for the concurrency operator, i.e., besides two sorts of nondeterminism, we also (implicitly) touch upon concurrency in this paper.

4. Why the obvious approach will not do

Both in [Pel87] and [Gol92] a semantics for events is given that deals with the concurrency operator $\cap$, which bears a close resemblance to our external nondeterministic choice $+$, and the nondeterministic choice operator $\cup$, which shows the same behaviour as our $\oplus$. One could be tempted to introduce nondeterminism/concurrency into the system of [HLM93] by using a Peleg/Goldblatt approach towards the semantics for events, and extending the semantics for abilities by adding the clause

$$\mathsf{c}(i, \alpha_1 \oplus \alpha_2)(s) = \mathbf{1} \Leftrightarrow \mathsf{c}(i, \alpha_1)(s) = \mathbf{1} \vee \mathsf{c}(i, \alpha_2)(s) = \mathbf{1}$$

and

$$\mathsf{c}(i, \alpha_1 + \alpha_2)(s) = \mathbf{1} \Leftrightarrow \mathsf{c}(i, \alpha_1)(s) = \mathbf{1} \wedge \mathsf{c}(i, \alpha_2)(s) = \mathbf{1}$$

to deal with nondeterminism/concurrency. However according to our intuition this semantics fails to satisfactory formalize the abilities of agents for the sequential composition. The following two examples make this point more clear.

4.1. Example (Internal nondeterminism and sequential composition).
Consider the Kripke model $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathsf{r}, \mathsf{c} >$ given by:

(1) $\mathcal{S} = \{s_0, s_1, s_2\}$.
(2) π is arbitrary.
(3) R is arbitrary.
(4) $\mathsf{r}(i, a_1)(s_0) = s_1$, $\mathsf{r}(i, a_2)(s_0) = s_2$.
(5) $\mathsf{c}(i, a_1)(s_0) = \mathbf{1} = \mathsf{c}(i, a_3)(s_1)$,
$\mathsf{c}(i, a_2)(s_0) = \mathbf{0} = \mathsf{c}(i, a_3)(s_2)$.

Intuitively agent i is capable of $\alpha = (a_1 \oplus a_2); a_3$ in s_0: s/he is capable of a_1, and since all possible executions of a_1 lead to states in which s/he is capable of a_3, it is clear that the agent is capable of $a_1; a_3$, i.e., in the terminology of section 1, the agent is capable of performing one of the actions constituting α. Should we use the semantics suggested above, in which the equivalence

$$[\mathrm{do}_i(\alpha_1 \oplus \alpha_2)]\varphi \leftrightarrow [\mathrm{do}_i(\alpha_1)]\varphi \wedge [\mathrm{do}_i(\alpha_2)]\varphi$$

defines the $\oplus$ for events, we would have to conclude that i is not capable of α:

(1) $\mathcal{M}, s_0 \models \mathbf{A}_i(a_1 \oplus a_2)$ since $\mathcal{M}, s_0 \models \mathbf{A}_i a_1$,
(2) $\mathcal{M}, s_0 \not\models [\mathrm{do}_i(a_1 \oplus a_2)]\mathbf{A}_i a_3$, since $\mathcal{M}, s_0 \not\models [\mathrm{do}_i(a_2)]\mathbf{A}_i a_3$,
(3) hence $\mathcal{M}, s_0 \not\models \mathbf{A}_i(a_1 \oplus a_2) \wedge [\mathrm{do}_i(a_1 \oplus a_2)]\mathbf{A}_i a_3$, and thus $\mathcal{M}, s_0 \not\models \mathbf{A}_i\alpha$.

In our opinion this is not the intuitively correct approach towards sequential composition.

4.2. Example (External nondeterminism and sequential composition).
Consider the Kripke model $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$ given by:

(1) $\mathcal{S} = \{s_0, s_1\}$.
(2) π is arbitrary.
(3) R is arbitrary.
(4) $\mathtt{r}(i, a_1)(s_0) = s_1$, $\mathtt{r}(i, a_2)(s_0) = \emptyset$.
(5) $\mathtt{c}(i, a_1)(s_0) = \mathbf{1} = \mathtt{c}(i, a_2)(s_0)$,
$\mathtt{c}(i, a_3)(s_1) = \mathbf{0}$.

Intuitively agent i is incapable of $\alpha = (a_1 + a_2); a_3$ in s_0: s/he is capable of a_1 and a_2 in s_0, but since the only possible execution of a_1 leads to a state in which the agent is incapable of a_3, s/he is incapable of $a_1; a_3$, i.e., in the terminology of section 1, the agent is incapable of performing one of the actions constituting α. Using a Peleg-like semantics for events would result in the equivalence

$$[\mathrm{do}_i(\alpha_1 + \alpha_2)]\varphi \leftrightarrow [\mathrm{do}_i(\alpha_1)]\varphi \vee [\mathrm{do}_i(\alpha_2)]\varphi$$

whereas a Goldblatt-like semantics would yield

$$[\mathrm{do}_i(\alpha_1 + \alpha_2)]\varphi \leftrightarrow (<\mathrm{do}_i(\alpha_1>\mathbf{tt} \rightarrow [\mathrm{do}_i(\alpha_2)]\varphi) \wedge (<\mathrm{do}_i(\alpha_2>\mathbf{tt} \rightarrow [\mathrm{do}_i(\alpha_1)]\varphi)$$

as a defining equivalence[1]. Regardless of which of these equivalences is used in the approach suggested above, we have to conclude that the agent is capable of α in s_0:

(1) $\mathcal{M}, s_0 \models \mathbf{A}_i(a_1 + a_2)$ since $\mathcal{M}, s_0 \models \mathbf{A}_i a_1$ and $\mathcal{M}, s_0 \models \mathbf{A}_i a_2$,
(2) $\mathcal{M}, s_0 \models [\mathrm{do}_i(a_2)]\mathbf{A}_i a_3$ and $\mathcal{M}, s_0 \not\models <\mathrm{do}_i(a_2)>\mathbf{tt}$, hence $\mathcal{M}, s_0 \models [\mathrm{do}_i(a_1 + a_2)]\mathbf{A}_i a_3$,
(3) hence $\mathcal{M}, s_0 \models \mathbf{A}_i(a_1 + a_2) \wedge [\mathrm{do}_i(a_1 + a_2)]\mathbf{A}_i a_3$, and thus $\mathcal{M}, s_0 \models \mathbf{A}_i(a_1 + a_2); a_3$.

Again the obvious approach is not an intuitively acceptable one.

An intuitively correct treatment of the situations sketched in the examples 4.1 and 4.2 is what we are striving for with our definition of the semantics for the nondeterministic

[1] To avoid confusion both equivalences are expressed using our notation.

choices. The idea behind the semantics as we give it, is that the meaning of a composite action is determined by the meaning of the sequences of (atomic) actions that constitute it: for instance, in example 4.1, the meaning of $\alpha = (a_1 \oplus a_2); a_3$ is determined by the meaning of the sequences $a_1; a_3$ and $a_2; a_3$.

5. Introducing internal nondeterminism

In this section we try and extend the definitions of section 2 such that internal nondeterminism is adequately dealt with. In particular example 4.1 should be treated in an intuitively acceptable way.

We start with extending the syntax in an obvious way.

5.1. Definition. The language $\mathcal{L}_I$ is an extension of the language $\mathcal{L}$. For the language $\mathcal{L}_I$ the class of actions Ac as given in definition 2.2 is extended by the following, thus obtaining the class Ac_I of actions.

$$\alpha ::= \alpha_1 \oplus \alpha_2$$

As touched upon at the end of section 4, the unraveling of composite actions into the sequences of (atomic) actions that constitute them will be the basis of the semantics that we define. This unraveling is done using *finite computation sequences*, as defined in [KT90]. A finite computation sequence of a given action α is a finite length string of atomic actions and tests, representing a possible sequence of atomic steps that may occur in a halting execution of some event $\mathrm{do}_i(\alpha)$.

5.2. Definition. Let the language $\mathcal{L}_I$ be as given in definition 5.1. The class of basic actions Ac_b^I based on $\mathcal{L}_I$ is given by the following BNF, where a is a typical element of At and φ is a typical formula from $\mathcal{L}_I$.

$$\alpha ::= a \mid \texttt{confirm}\ \varphi \mid \alpha_1; \alpha_2$$

If some action α is either an atomic action a or some confirmation $\texttt{confirm}\ \varphi$, then α is called *semi-atomic*.

5.3. Remark. Note that the class Ac_b^I is a subclass of Ac_I.

5.4. Lemma. *For all $\alpha \in Ac_b^I$ and for all $\mathcal{M} \in \mathfrak{M}$ it holds that $\mathrm{do}_i(\alpha)$ is deterministic for $\mathcal{M}$.*

Proof of lemma 5.4: By induction on the structure of α.
$\boxtimes$

NOTATION. To keep our notation compact we will sometimes use the following abbreviation:

$$\Pi_{j=1}^{k}(\varphi, \alpha_j) \stackrel{\text{def}}{=} (\texttt{confirm}\ \varphi; \alpha_1); \ldots; (\texttt{confirm}\ \varphi; \alpha_k); \texttt{confirm}\ \neg\varphi$$

where $k \in \mathbb{N}, k \geq 1$, φ is some formula, and the α_j's are actions.

5.5. DEFINITION. The function CS, yielding the finite computation sequences of a given action, is inductively defined as follows.

$$\begin{array}{ll}
CS & : \ Ac_I \to \wp(Ac_b) \\
CS(a) & = \{a\} \\
CS(\texttt{confirm}\ \varphi) & = \{\texttt{confirm}\ \varphi\} \\
CS(\alpha_1; \alpha_2) & = \{\alpha_1'; \alpha_2' \mid \alpha_1' \in CS(\alpha_1), \alpha_2' \in CS(\alpha_2)\} \\
CS(\texttt{if}\ \varphi\ \texttt{then}\ \alpha_1\ \texttt{else}\ \alpha_2\ \texttt{fi}) & = CS(\texttt{confirm}\ \varphi; \alpha_1) \cup CS(\texttt{confirm}\ \neg\varphi; \alpha_2) \\
CS(\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}) & = \cup_{k=1}^{\infty} Seq_k(\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}) \cup \{\texttt{confirm}\ \neg\varphi\} \\
CS(\alpha_1 \oplus \alpha_2) & = CS(\alpha_1) \cup CS(\alpha_2)
\end{array}$$

where for $k \geq 1$

$$Seq_k(\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}) = \{\Pi_{j=1}^{k}(\varphi, \alpha_j') \mid \alpha_j' \in CS(\alpha_1) \text{ for } j = 1, \ldots, k\}$$

5.6. REMARK. The definition of the finite computation sequences of action α is essentially identical to that of the *trace set* of α using linear time semantics (cf. [BBKM84]). Note furthermore that the set of finite computation sequences of an action α is determined by the syntactical shape of α only.

5.7. LEMMA. *For all actions $\alpha \in Ac_I$ we have: $CS(\alpha) \neq \emptyset$.*

PROOF OF LEMMA 5.7: By induction on the structure of α: the cases for atomic actions, confirmations, conditional composition, and repetitive composition are straightforward from definition 5.5; these cases do not use the induction hypothesis. In the other cases the induction hypothesis is used.
⊠

Having introduced the notion of finite computation sequences, the semantics fit to cope in an intuitively acceptable way with the internal nondeterministic choice can be defined. For events and abilities this semantics is a straightforward translation of the intuitive idea that $<\mathrm{do}_i(\alpha)>\varphi$ holds iff it is possible for the agent i to execute α in such a way that φ results, and $\mathbf{A}_i\alpha$ holds iff the agent i is capable of performing at least one of the atomic sequences that constitute α.

5.8. DEFINITION. The valuation π as given in definition 2.6 is modified as follows.

$$\begin{array}{ll}
\pi(<\mathrm{do}_i(\alpha)>\varphi, s) = \mathbf{1} & \Leftrightarrow \exists \alpha' \in CS(\alpha) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}] \\
\pi(\mathbf{A}_i\alpha, s) = \mathbf{1} & \Leftrightarrow \exists \alpha' \in CS(\alpha)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]
\end{array}$$

where

$$\begin{aligned}
&\mathtt{r}(i, \mathtt{confirm}\,\varphi)(s) && = \{s\} \text{ if } \pi(\varphi, s) = \mathbf{1} \\
&&& = \emptyset \text{ otherwise} \\
&\mathtt{r}(i, \alpha_1; \alpha_2)(s) = s' && \Leftrightarrow \exists t \in \mathcal{S}[\mathtt{r}(i, \alpha_1)(s) = t \,\&\, \mathtt{r}(i, \alpha_2)(t) = s'] \\
\\
&\mathtt{c}(i, \mathtt{confirm}\,\varphi)(s) && = \pi(\varphi, s) \\
&\mathtt{c}(i, \alpha_1; \alpha_2)(s) && = \mathtt{c}(i, \alpha_1)(s) \wedge \mathtt{c}(i, \alpha_2)(\mathtt{r}(i, \alpha_1)(s))
\end{aligned}$$

and

$$\mathtt{c}(i, \alpha)(\emptyset) = \mathbf{1}$$

5.9. Remark. Since events and abilities are separate concepts, it it important to note that whenever in an example in this paper two states are connected via an arrow this is an *events only* arrow, which has nothing to do with the actual abilities of the agent in question.

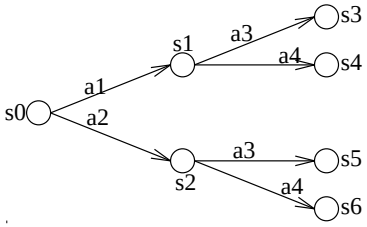
5.10. Examples. The following examples show how definitions 5.5 and 5.8 work out in practice.

- Consider the Kripke model $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$ as given in example 4.1. Again consider the ability of agent i for the action $\alpha = (a_1 \oplus a_2); a_3$:

  $\mathcal{M}, s_0 \models \mathbf{A}_i\alpha$
  $\Leftrightarrow \pi(\mathbf{A}_i\alpha, s_0) = \mathbf{1}$
  $\Leftrightarrow \exists \alpha' \in CS(\alpha)[\mathtt{c}(i, \alpha')(s_0) = \mathbf{1}]$
  $\Leftrightarrow \mathtt{c}(i, a_1; a_3)(s_0) = \mathbf{1}$ or $\mathtt{c}(i, a_2; a_3)(s_0) = \mathbf{1}$
  $\Leftrightarrow (\mathtt{c}(i, a_1)(s_0) = \mathbf{1}$ and $\mathtt{c}(i, a_3)(s_1) = \mathbf{1})$ or
  $(\mathtt{c}(i, a_2)(s_0) = \mathbf{1}$ and $\mathtt{c}(i, a_3)(s_2) = \mathbf{1})$
  $\Leftrightarrow$ true, since $\mathtt{c}(i, a_1)(s_0) = \mathbf{1}$ and $\mathtt{c}(i, a_3)(s_1) = \mathbf{1}$

  Since the agent is capable of performing one of the atomic sequences that constitute the action α, s/he is able to perform α. This is exactly the outcome as we intuitively expected it to be.

- Assume that $\mathcal{M}$ is the Kripke model for which the function $\mathtt{r}$ is for agent i as in the following picture.

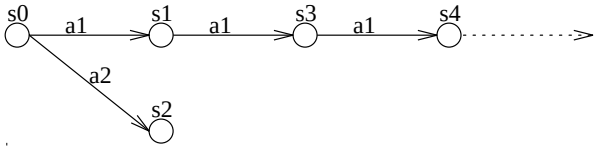


  Let action α be given by $\alpha \stackrel{\text{def}}{=} (a_1 \oplus a_2); (a_3 \oplus a_4)$. We have the following for $\mathcal{M}$:

$$\begin{aligned}
CS(\alpha) &= \{\alpha_1; \alpha_2 \mid \alpha_1 \in CS(a_1 \oplus a_2), \alpha_2 \in CS(a_3 \oplus a_4)\} \\
&= \{\alpha_1; \alpha_2 \mid \alpha_1 \in \{a_1, a_2\}, \alpha_2 \in \{a_3, a_4\}\} \\
&= \{a_1; a_3,\ a_1; a_4,\ a_2; a_3,\ a_2; a_4\}
\end{aligned}$$

- Let $\mathcal{M}$ be the Kripke model given by:

(1) $\mathcal{S} = \{s_j \mid j \in \mathbb{N}\}$
(2) $\pi(p, s_j) = \mathbf{0} \Leftrightarrow j = 2$,
$\pi(q, s_j) = \mathbf{1} \Leftrightarrow j = 2$.
(3) R is arbitrary.
(4) $\mathtt{r}$: see picture below for the action transitions for agent i.
(5) $\mathtt{c}(i, a_1)(s_j) = \mathtt{c}(i, a_2)(s_j) = \mathbf{1}$, for all $j \in \mathbb{N}$.



Let $\alpha = \mathtt{while}\ p\ \mathtt{do}\ a_1 \oplus a_2\ \mathtt{od}$. Note that the finite computation sequences for α are given by the following equation:

$$CS(\alpha) = \{\Pi_{j=1}^{k}(p, \alpha_j) \mid k \in \mathbb{N}, k \geq 1, \alpha'_j \in \{a_1, a_2\}\} \cup \{\mathtt{confirm}\ \neg p\}$$

5.11. Proposition. *The following is true for $\mathcal{M}$:*

(1) $\mathcal{M}, s_0 \models <\mathrm{do}_i(\alpha)> q$.
(2) $\mathcal{M}, s_0 \models \mathbf{A}_i\alpha$.

Proof of proposition 5.11:

Case 1:
$\mathcal{M}, s_0 \models <\mathrm{do}_i(\alpha)> q$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha)> q, s_0) = \mathbf{1}$
$\Leftrightarrow \exists \alpha' \in CS(\alpha) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(q, s') = \mathbf{1}]$
$\Leftrightarrow$ true, since $\alpha' \stackrel{\mathrm{def}}{=} (\mathtt{confirm}\ p; a_2); \mathtt{confirm}\ \neg p \in CS(\alpha)$, $\mathtt{r}(i, \alpha')(s) = s_2$, and $\pi(q, s_2) = \mathbf{1}$.

Case 2:
$\mathcal{M}, s_0 \models \mathbf{A}_i\alpha$
$\Leftrightarrow \pi(\mathbf{A}_i\alpha, s_0) = \mathbf{1}$
$\Leftrightarrow \exists \alpha' \in CS(\alpha)[\mathtt{c}(i, \alpha')(s_0) = \mathbf{1}]$
$\Leftrightarrow$ true, since $(\mathtt{confirm}\ p; a_2); \mathtt{confirm}\ \neg p \in CS(\alpha)$, and $\mathtt{c}(i, (\mathtt{confirm}\ p; a_2); \mathtt{confirm}\ \neg p)(s_0) = \mathbf{1}$.

$\boxtimes$

The following theorem shows that our semantics for the internal nondeterministic choice behaves as desired.

5.12. Theorem. *For all agents i, actions $\alpha_1, \alpha_2 \in Ac_I$, and for all formulae φ we have:*

- $\models <\mathrm{do}_i(\alpha_1 \oplus \alpha_2)> \varphi \leftrightarrow (<\mathrm{do}_i(\alpha_1)> \varphi \vee <\mathrm{do}_i(\alpha_2)> \varphi)$.
- $\models \mathbf{A}_i(\alpha_1 \oplus \alpha_2) \leftrightarrow (\mathbf{A}_i\alpha_1 \vee \mathbf{A}_i\alpha_2)$.

Proof of theorem 5.12:

First case:

$\mathcal{M}, s \models <\mathrm{do}_i(\alpha_1 \oplus \alpha_2)> \varphi$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1 \oplus \alpha_2)> \varphi, s) = \mathbf{1}$
$\Leftrightarrow \exists \alpha' \in CS(\alpha_1 \oplus \alpha_2) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \exists \alpha' \in CS(\alpha_1) \cup CS(\alpha_2) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \exists \alpha' \in CS(\alpha_1) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$ or
$\exists \alpha' \in CS(\alpha_2) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1)> \varphi, s) = \mathbf{1}$ or $\pi(<\mathrm{do}_i(\alpha_2)> \varphi, s) = \mathbf{1}$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1)> \varphi \vee <\mathrm{do}_i(\alpha_2)> \varphi, s) = \mathbf{1}$
$\Leftrightarrow \mathcal{M}, s \models <\mathrm{do}_i(\alpha_1)> \varphi \vee <\mathrm{do}_i(\alpha_2)> \varphi$

Second case:

$\mathcal{M}, s \models \mathbf{A}_i(\alpha_1 \oplus \alpha_2)$
$\Leftrightarrow \pi(\mathbf{A}_i(\alpha_1 \oplus \alpha_2), s) = \mathbf{1}$
$\Leftrightarrow \exists \alpha' \in CS(\alpha_1 \oplus \alpha_2)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$
$\Leftrightarrow \exists \alpha' \in CS(\alpha_1) \cup CS(\alpha_2)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$
$\Leftrightarrow \exists \alpha' \in CS(\alpha_1)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$ or $\exists \alpha' \in CS(\alpha_2)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$
$\Leftrightarrow \pi(\mathbf{A}_i \alpha_1, s) = \mathbf{1}$ or $\pi(\mathbf{A}_i \alpha_2, s) = \mathbf{1}$
$\Leftrightarrow \pi(\mathbf{A}_i \alpha_1 \vee \mathbf{A}_i \alpha_2, s) = \mathbf{1}$
$\Leftrightarrow \mathcal{M}, s \models \mathbf{A}_i \alpha_1 \vee \mathbf{A}_i \alpha_2$

$\boxtimes$

5.1. Composite actions revisited

In [HLM93] some validities in the language $\mathcal{L}$ of definition 2.2 on $\mathfrak{M}$ of definition 2.5 were proved, both for events and abilities. In this section these validities are reconsidered in the light of the semantics for the internal nondeterministic choice as introduced in the previous section. Theorem 5.13 deals with events, theorem 5.14 deals with abilities.

With regard to events, our new semantics behaves as the semantics given in [HLM93].

5.13. THEOREM. *For all agents i, actions $\alpha_1, \alpha_2 \in Ac_I$, and for all formulae φ and ψ we have:*

(1) $\models \varphi \leftrightarrow <\mathrm{do}_i(\mathtt{skip})> \varphi$.

(2) $\models \neg <\mathrm{do}_i(\mathtt{fail})> \mathbf{tt}$.

(3) $\models <\mathrm{do}_i(\mathtt{confirm}\ \varphi)> \psi \leftrightarrow (\varphi \wedge \psi)$.

(4) $\models <\mathrm{do}_i(\alpha_1; \alpha_2)> \varphi \leftrightarrow <\mathrm{do}_i(\alpha_1)> (<\mathrm{do}_i(\alpha_2)> \varphi)$.

(5) $\models <\mathrm{do}_i(\mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi})> \psi \leftrightarrow$
$<\mathrm{do}_i(\mathtt{confirm}\ \varphi; \alpha_1)> \psi \vee <\mathrm{do}_i(\mathtt{confirm}\ \neg\varphi; \alpha_2)> \psi$.

(6) $\models <\mathrm{do}_i(\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od})> \psi \leftrightarrow$
$<\mathrm{do}_i(\mathtt{confirm}\ \neg\varphi)> \psi \vee <\mathrm{do}_i(\mathtt{confirm}\ \varphi; \alpha_1)> <\mathrm{do}_i(\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od})> \psi$.

PROOF OF THEOREM 5.13: The first three cases are straightforward, since it holds that $CS(\alpha) = \alpha$, for $\alpha = \texttt{skip}, \texttt{fail}, \texttt{confirm}\ \varphi$. Also case 5 is easy to prove. Since the proof of case 6 is analogous to that of case 4, we will prove case 4 only; equivalences that are standard for first-order logic (see for instance [Gam91]) are assumed to be familiar and are therefore used throughout this proof without mentioning them.

Case 4:

$\mathcal{M}, s \models <\mathrm{do}_i(\alpha_1;\alpha_2)>\varphi$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1;\alpha_2)>\varphi, s) = \mathbf{1}$
$\Leftrightarrow \exists \alpha' \in CS(\alpha_1;\alpha_2)\exists s'[\mathtt{r}(i,\alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \exists \alpha'_1 \in CS(\alpha_1)\exists \alpha'_2 \in CS(\alpha_2)\exists s'\exists s''[\mathtt{r}(i,\alpha'_1)(s) = s'' \,\&\, \mathtt{r}(i,\alpha'_2)(s'') = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \exists \alpha'_1 \in CS(\alpha_1)\exists s''\exists \alpha'_2 \in CS(\alpha_2)\exists s'[\mathtt{r}(i,\alpha'_1)(s) = s'' \,\&\, \mathtt{r}(i,\alpha'_2)(s'') = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \exists \alpha'_1 \in CS(\alpha_1)\exists s''[\mathtt{r}(i,\alpha'_1)(s) = s'' \,\&\, \exists \alpha'_2 \in CS(\alpha_2)\exists s'[\mathtt{r}(i,\alpha'_2)(s'') = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]]$
$\Leftrightarrow \exists \alpha'_1 \in CS(\alpha_1)\exists s''[\mathtt{r}(i,\alpha'_1)(s) = s'' \,\&\, \pi(<\mathrm{do}_i(\alpha_2)>\varphi, s'') = \mathbf{1}]$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1)><\mathrm{do}_i(\alpha_2)>\varphi) = \mathbf{1}$
$\Leftrightarrow \mathcal{M}, s \models <\mathrm{do}_i(\alpha_1)><\mathrm{do}_i(\alpha_2)>\varphi$

⊠

It turns out that with regard to abilities the sequential composition and the repetitive composition behave differently for the (nondeterministic) semantics of section 5 than they did for the (deterministic) semantics of [HLM93]; for the other composite actions the same validities are found as in [HLM93].

5.14. THEOREM. *For all agents i, actions $\alpha_1, \alpha_2 \in Ac_I$, and for all formulae φ we have:*

(1) $\models \mathbf{A}_i\texttt{skip}$.

(2) $\models \neg\mathbf{A}_i\texttt{fail}$.

(3) $\models \varphi \leftrightarrow \mathbf{A}_i\texttt{confirm}\ \varphi$.

(4) $\models \mathbf{A}_i\texttt{if}\ \varphi\ \texttt{then}\ \alpha_1\ \texttt{else}\ \alpha_2\ \texttt{fi} \leftrightarrow \mathbf{A}_i(\texttt{confirm}\ \varphi;\alpha_1) \vee \mathbf{A}_i(\texttt{confirm}\ \neg\varphi;\alpha_2)$.

PROOF OF THEOREM 5.14: Straightforward and left to the reader.
⊠

5.15. THEOREM. *For all agents i, actions $\alpha_1, \alpha_2, \alpha_3 \in Ac_I$, and for all formulae φ we have:*

(1) $\not\models \mathbf{A}_i(\alpha_1;\alpha_2) \rightarrow \mathbf{A}_i\alpha_1 \wedge [\mathrm{do}_i(\alpha_1)]\mathbf{A}_i\alpha_2$.

(2) $\models \mathbf{A}_i\alpha_1 \wedge [\mathrm{do}_i(\alpha_1)]\mathbf{A}_i\alpha_2 \rightarrow \mathbf{A}_i(\alpha_1;\alpha_2)$.

(3) $\models \mathbf{A}_i((\alpha_1 \oplus \alpha_2);\alpha_3) \leftrightarrow (\mathbf{A}_i(\alpha_1;\alpha_3) \vee \mathbf{A}_i(\alpha_2;\alpha_3))$.

(4) $\models \mathbf{A}_i(\alpha_1;(\alpha_2 \oplus \alpha_3) \leftrightarrow (\mathbf{A}_i(\alpha_1;\alpha_2) \vee \mathbf{A}_i(\alpha_1;\alpha_3))$.

(5) $\models \mathbf{A}_i\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od} \leftrightarrow (\mathbf{A}_i\texttt{confirm}\ \neg\varphi \vee \mathbf{A}_i((\texttt{confirm}\ \varphi;\alpha_1);\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}))$.

PROOF OF THEOREM 5.15: Case 1 is proved by the first of the examples given in 5.10. The proofs of the cases 3, 4, and 5 are left to the reader. We show case 2.

$\mathcal{M}, s \models \mathbf{A}_i\alpha_1 \wedge [\mathrm{do}_i(\alpha_1)]\mathbf{A}_i\alpha_2$
$\Leftrightarrow \exists\alpha_1' \in CS(\alpha_1)[\mathsf{c}(i,\alpha_1')(s) = \mathbf{1}]\,\&$
$\forall\alpha_1' \in CS(\alpha_1)\forall s'[\mathsf{r}(i,\alpha_1')(s) = s' \Rightarrow \exists\alpha_2' \in CS(\alpha_2)[\mathsf{c}(i,\alpha_2')(s') = \mathbf{1}]]$

Now let α_1' be such that $\mathsf{c}(i,\alpha_1') = \mathbf{1}$. We distinguish two cases:

- $\mathsf{r}(i,\alpha_1')(s) = \emptyset$. Take an arbitrary $\alpha_2' \in CS(\alpha_2)$; this is possible since $CS(\alpha_2) \neq \emptyset$ (lemma 5.7). It now holds that $\alpha_1';\alpha_2' \in CS(\alpha_1;\alpha_2)$ and $\mathsf{c}(i,\alpha_1';\alpha_2')(s) = \mathbf{1}$, and hence $\mathcal{M}, s \models \mathbf{A}_i\alpha_1;\alpha_2$.
- $\mathsf{r}(i,\alpha_1')(s) = s'$, for some $s' \in \mathcal{S}$. It then follows that $\mathsf{c}(i,\alpha_2')(s') = \mathbf{1}$ for some $\alpha_2' \in CS(\alpha_2)$. Hence $\mathsf{c}(i,\alpha_1';\alpha_2')(s) = \mathbf{1}$ and since $\alpha_1';\alpha_2' \in CS(\alpha_1;\alpha_2)$ we conclude that $\mathcal{M}, s \models \mathbf{A}_i\alpha_1;\alpha_2$.

This suffices to conclude that case 2 holds.
⊠

In our opinion theorems 5.13 to 5.15 show that our semantics are intuitively acceptable. The first two cases of theorem 5.15 are associated with the properties observed in example 4.1 and in the first example of 5.10.

5.2. The Can-predicate and the Cannot-predicate reconsidered

To formalize the knowledge of agents concerning correctness and feasibility of their plans with respect to a given goal we used in [HLM93] a completely modified version of the Can-predicate, originally defined by Robert Moore ([Moo80], [Moo84]).
Intuitively the Can-predicate $\mathbf{Can}_i(\alpha,\varphi)$ expresses the fact that agent i knows that α is a correct and feasible plan to achieve the goal φ. As already indicated in section 2, correctness of α with respect to φ is formalized by $<\mathrm{do}_i(\alpha)>\varphi$ and feasibility of α for agent i is formalized by $\mathbf{A}_i\alpha$. Hence a straightforward definition of the Can-predicate is the following one, which actually is the one used in [HLM93].

$$\mathbf{Can}_i'(\alpha,\varphi) \equiv \mathbf{K}_i(<\mathrm{do}_i(\alpha)>\varphi \wedge \mathbf{A}_i\alpha) \qquad (\dagger)$$

For the deterministic case this definition turned out to be intuitively perfectly acceptable (see [HLM93]). However as soon as internal nondeterministic actions are involved some awkward situations come into being:

5.16. Example (The poor agent). Consider the situation of a poor agent i. The agent knows that making money by magic would make her/him rich, but of course s/he is incapable of doing so. This situation is formalized in the model $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathsf{r}, \mathsf{c} >$, where a is the act of making money by magic, and r stands for richness.

(1) $\mathcal{S} = \{s_0, s_1\}$,
(2) $\pi(r, s_0) = \mathbf{0}$, $\pi(r, s_1) = \mathbf{1}$,
(3) $\mathrm{R}(i) = \{(s_0, s_0), (s_1, s_1)\}$,

(4) $\mathtt{r}(i,a)(s_0) = s_1$,

(5) $\mathtt{c}(i,a)(s_0) = \mathbf{0}$.

In this model we find the following to hold:

(1) $\mathcal{M}, s_0 \models <\mathrm{do}_i(a)> r$ and hence $\mathcal{M}, s_0 \models <\mathrm{do}_i(\mathtt{skip} \oplus a)> r$.

(2) $\mathcal{M}, s_0 \models \mathbf{A}_i\mathtt{skip}$ and hence $\mathcal{M}, s_0 \models \mathbf{A}_i(\mathtt{skip} \oplus a)$.

(3) $\mathcal{M}, s_0 \models \mathbf{K}_i(<\mathrm{do}_i(\mathtt{skip} \oplus a)> r \wedge \mathbf{A}_i(\mathtt{skip} \oplus a))$, i.e.,
$\mathcal{M}, s_0 \models \mathbf{Can}'_i(\mathtt{skip} \oplus a, r)$.

Hence the agent concludes that s/he has a correct and feasible plan to become rich! Since this conclusion is highly undesired, definition (†) is to be rejected.

To avoid the kind of counterintuitive behaviour as expressed in example 5.16, the Can-predicate needs to be modified. A first possible attempt to modify the Can-predicate could be to demand that all possible ways of doing α lead to the truth of φ, thus ending up with the following definition:

$$\mathbf{Can}''_i(\alpha, \varphi) \equiv \mathbf{K}_i([\mathrm{do}_i(\alpha)]\varphi \wedge <\mathrm{do}_i(\alpha)>\varphi \wedge \mathbf{A}_i\alpha) \qquad (\ddagger)$$

However definition (‡) does not yield intuitively acceptable results either:

5.17. Example (The living agent). Consider the situation of an agent i that has a zest for living. The agent knows that breathing keeps him/her alive and also that if s/he holds his/her breath for half an hour, s/he would no longer be alive. The agent furthermore knows that s/he is capable of breathing and not able to hold his/her breath for half an hour. A possible formalization of this situation is given by the model $\mathcal{M} = <\mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c}>$, where b is the act of breathing, h is the act of holding your breath, and l stands for the agent being alive.

(1) $\mathcal{S} = \{s_0, s_1\}$,

(2) $\pi(l, s_0) = \mathbf{1}$, $\pi(l, s_1) = \mathbf{0}$,

(3) $\mathrm{R}(i) = \{(s_0, s_0), (s_1, s_1)\}$,

(4) $\mathtt{r}(i,b)(s_0) = s_0$, $\mathtt{r}(i,h)(s_0) = s_1$,

(5) $\mathtt{c}(i,b)(s_0) = \mathbf{1}$, $\mathtt{c}(i,h)(s_0) = \mathbf{0}$.

In this model we find the following to hold:

(1) $\mathcal{M}, s_0 \models <\mathrm{do}_i(b)> l$ and hence $\mathcal{M}, s_0 \models <\mathrm{do}_i(b \oplus h)> l$.

(2) $\mathcal{M}, s_0 \models \mathbf{A}_i b$ and hence $\mathcal{M}, s_0 \models \mathbf{A}_i(b \oplus h)$.

(3) $\mathcal{M}, s_0 \not\models [\mathrm{do}_i(h)]l$ and hence $\mathcal{M}, s_0 \not\models [\mathrm{do}_i(b \oplus h)]l$.

(4) $\mathcal{M}, s_0 \not\models \mathbf{K}_i([\mathrm{do}_i(b \oplus h)]l)$, and hence $\mathcal{M}, s_0 \not\models \mathbf{Can}''_i(b \oplus h, l)$.

Despite the fact that agent i determines which action is chosen and s/he knows that one of the actions b and h is correct and feasible, the agent does not know that $b \oplus h$ is a

correct and feasible plan to stay alive. This is a consequence of the fact that not all possible ways of doing $b \oplus h$ lead to states in which the agent is alive. But since the agent decides which action to perform, the demand that all possible ways of doing an (internal nondeterministic) action lead to the desired goal is a counterintuitive one in the first place. Therefore definition (‡) is also to be rejected.

In order to ensure that the Can-predicate and the Cannot-predicate behave in an intuitively acceptable way in the situations sketched in the examples above, we will for the internal nondeterministic case abandon the idea of defining the Can-predicate and the Cannot-predicate as syntactical abbreviations. Instead we will treat these predicates as independent ones that have their own semantics. It will not come as a surprise that the unravelling of actions again plays an important part in this semantics.

5.18. Definition. For a given model $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$ the semantics of the Can-predicate and the Cannot-predicate is defined as follows:

- $\mathcal{M}, s \models \mathbf{Can}_i(\alpha, \varphi)$ if and only if
  $\exists \alpha' \in CS(\alpha)[\mathcal{M}, s \models \mathbf{K}_i(<\mathrm{do}_i(\alpha')>\varphi \wedge \mathbf{A}_i\alpha')]$.
- $\mathcal{M}, s \models \mathbf{Cannot}_i(\alpha, \varphi)$ if and only if
  $\forall \alpha' \in CS(\alpha)[\mathcal{M}, s \models \mathbf{K}_i(\neg <\mathrm{do}_i(\alpha')>\varphi \vee \neg\mathbf{A}_i\alpha')]$.

The Can-predicate as given in definition 5.18 formalizes the idea that an agent knows that some action α is a correct and feasible plan for some goal φ if and only if for some $\alpha' \in CS(\alpha)$ s/he knows that α' is a correct and feasible plan for the goal. Defining the Can-predicate and the Cannot-predicate as in 5.18 resolves the counterintuitive situations that occurred in examples 5.16 and 5.17:

- in example 5.16, $\mathcal{M}, s_0 \not\models \mathbf{Can}_i(a \oplus \mathtt{skip}, r)$, and $\mathcal{M}, s_0 \models \mathbf{Cannot}_i(a \oplus \mathtt{skip}, r)$, i.e., the agent knows that $a \oplus \mathtt{skip}$ is either incorrect or unfeasible and therefore cannot be used to achieve r.
- in example 5.17, $\mathcal{M}, s_0 \models \mathbf{Can}_i(b \oplus h, l)$ and $\mathcal{M}, s_0 \models \mathbf{Cannot}_i(b \oplus h, \neg l)$, i.e, not only does the agent know that $b \oplus h$ is a correct and feasible plan to stay alive, s/he also knows that it is not a correct and feasible plan to end his/her life. This corresponds to the idea that the agent decides which of b and h is to be executed.

Furthermore two intuitively desirable equivalences hold when using definition 5.18:

5.19. Lemma. *For all agents i, all actions α_1 and α_2, and for all formulae φ the following holds:*

- $\models \mathbf{Can}_i(\alpha_1 \oplus \alpha_2, \varphi) \leftrightarrow \mathbf{Can}_i(\alpha_1, \varphi) \vee \mathbf{Can}_i(\alpha_2, \varphi)$.
- $\models \mathbf{Cannot}_i(\alpha_1 \oplus \alpha_2, \varphi) \leftrightarrow \mathbf{Cannot}_i(\alpha_1, \varphi) \wedge \mathbf{Cannot}_i(\alpha_2, \varphi)$.

Proof of lemma 5.19: The lemma easily follows from definition 5.18.
⊠

Lemma 5.19 clearly expresses the idea of agent i being in control: the agent knows that an internal nondeterministic choice between two actions is correct and feasible if and only if one of the actions to be chosen is. Since the choice is completely free to the agent, this is what one intuitively would expect.

6. Introducing external nondeterminism

Next we will try and incorporate external nondeterminism into the system given in section 2. As in the case of internal nondeterminism, we start with extending the language $\mathcal{L}$.

6.1. DEFINITION. The language $\mathcal{L}_E$ is an extension of the language $\mathcal{L}$. For the language $\mathcal{L}_E$ the class of actions Ac as given in definition 2.2 is extended by the following, thus obtaining the class Ac_E of actions.

$$\alpha ::= \alpha_1 + \alpha_2$$

The class of basic actions based on $\mathcal{L}_E$ is defined in the obvious way:

6.2. DEFINITION. The class of basic actions Ac_b^E based on $\mathcal{L}_E$ is given by the following BNF, where a is a typical element of At and φ is a typical formula from $\mathcal{L}_E$.

$$\alpha ::= a \mid \texttt{confirm}\,\varphi \mid \alpha_1;\alpha_2$$

Note that, just as was the case with Ac_b^I, for all actions $\alpha \in Ac_b^E$ it holds that all events $\mathrm{do}_i(\alpha)$ are deterministic for all $\mathcal{M} \in \mathfrak{M}$.

In the light of the remarks made in the previous sections, one could be tempted to modify definition 5.8 in the following obvious manner:

- $\pi(<\mathrm{do}_i(\alpha)>\varphi, s) = \mathbf{1} \Leftrightarrow \forall \alpha' \in CS(\alpha)\exists s'[\mathtt{r}(i,\alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$ $(*)$
- $\pi(\mathbf{A}_i\alpha, s) = \mathbf{1} \Leftrightarrow \forall \alpha' \in CS(\alpha)[\mathtt{c}(i,\alpha')(s) = \mathbf{1}]$ $(**)$

This approach however has some undesired and awkward consequences, as can be seen in the following example.

6.3. EXAMPLE. Consider the action $\alpha \stackrel{\mathrm{def}}{=} \texttt{if}\ \varphi\ \texttt{then skip else fail fi}$. Assume that the event $\mathrm{do}_i(\alpha)$ occurs in a state s in which φ holds. Given the intuitive meaning of α one would expect the event $\mathrm{do}_i(\alpha)$ to behave in s as the event $\mathrm{do}_i(\texttt{skip})$ would. However, when using the definition of $\pi(<\mathrm{do}_i(\alpha)>\psi, s)$ as given in $(*)$, this is not the case:

$$\begin{aligned}
&\mathcal{M}, s \models <\mathrm{do}_i(\alpha)>\mathbf{tt} \\
\Leftrightarrow\ &\pi(<\mathrm{do}_i(\alpha)>\mathbf{tt}, s) = \mathbf{1} \\
\Leftrightarrow\ &\forall \alpha' \in CS(\alpha)\exists s'[\mathtt{r}(i,\alpha')(s) = s' \,\&\, \pi(\mathbf{tt}, s') = \mathbf{1}]
\end{aligned}$$

$\Leftrightarrow \exists s'[\mathtt{r}(i, \mathtt{confirm}\ \varphi; \mathtt{skip})(s) = s' \,\&\, \pi(\mathbf{tt}, s') = \mathbf{1}]$ and
$\exists s''[\mathtt{r}(i, \mathtt{confirm}\ \neg\varphi; \mathtt{fail})(s) = s'' \,\&\, \pi(\mathbf{tt}, s'') = \mathbf{1}]$
$\Leftrightarrow$ false, since $\mathtt{r}(i, \mathtt{confirm}\ \neg\varphi; \mathtt{fail})(s) = \emptyset$

whereas

$\mathcal{M}, s \models <\mathrm{do}_i(\mathtt{skip})>\mathbf{tt}$
$\Leftrightarrow \pi(<\mathrm{do}_i(\mathtt{skip})>\mathbf{tt}, s) = \mathbf{1}$
$\Leftrightarrow \forall \alpha' \in CS(\mathtt{skip}) \exists s'[\mathtt{r}(i, \mathtt{skip})(s) = s' \,\&\, \pi(\mathbf{tt}, s') = \mathbf{1}]$
$\Leftrightarrow \pi(\mathbf{tt}, s) = \mathbf{1}$
$\Leftrightarrow$ true.

Example 6.3 clearly shows the problems that are associated with using the definition $(*)$ (an analogous problem occurs with the definition $(**)$): due to the fact that the set of finite computation sequences of an action α is determined by the syntax of α alone, this set contains some sequences that should be left out of consideration in certain states for certain agents. For instance, the action `if` φ `then skip else fail fi` should have `confirm` φ; `skip` as the only relevant computation sequence in states in which φ holds.

To correctly deal with external nondeterminism, the *relevant* computation sequences of a given action, for a given agent in a given state, need somehow be singled out. We will call these relevant sequences the *finite computation runs* of the action, given an agent and a state.

Before we give the definition of the finite computation runs some additional terminology is necessary.

6.4. DEFINITION. For all agents i and actions α we define:

- the event $\mathrm{do}_i(\alpha)$ is *voidly non-terminating* in a given state s if the event does not cause any state transition. For instance the event $\mathrm{do}_i(\mathtt{fail})$ is voidly non-terminating in all states s.
- the event $\mathrm{do}_i(\alpha)$ is *infinitely non-terminating* in a given state s if the event causes infinitely many state transitions. An example of an event that is infinitely non-terminating in all states s is $\mathrm{do}_i(\mathtt{while}\ \mathbf{tt}\ \mathtt{do\ skip\ od})$

Since α may contain external nondeterministic choices, for $< \mathrm{do}_i(\alpha) > \varphi$ to hold it is demanded that no way of executing α exists such that a non-terminating event results; for in case of a non-terminating event certainly no end state exists, which is demanded for $<\mathrm{do}_i(\alpha)>\varphi$ to be true. For $\mathbf{A}_i\alpha$ to hold it is necessary that no way of executing α exists such that an infinitely non-terminating event results: since people die and machines break down, agents cannot be expected to be able to perform actions that result in infinitely non-terminating events. Note that it is possible that agents are able to perform actions that would lead to voidly non-terminating events (for instance the action that consists of washing hair in the shampoo example of section 2); it is therefore possible that actions resulting in a state s for an agent i in voidly non-terminating events still are relevant. For these reasons the definition of finite computation runs is such that if some computation sequence of an action α results, for a given agent i in a given state s, in an infinitely

non-terminating event, the set of finite computation runs of α for the agent and the state is equal to the singleton set {`fail`}. If none of the finite computation sequences of action α results in an infinitely non-terminating event for i in s, the set of finite computation runs of α is defined inductively. In this inductive definition it is taken into account that, depending on the truth or falsity of the condition, only some of the finite computation sequences of a conditional or a repetitive composition are finite computation runs. Note in particular that the set of finite computation runs of an action is determined by the context of the action, i.e., the agent that executes the action and the state in which it is executed, and is no longer determined by syntax alone.

To check whether an action α, for a given agent i and a state s, can be executed in such a way that an infinitely non-terminating event results, the predicate *Term* is used. The definition of this predicate is a rather straightforward formalization of the idea that infinite events result in infinitely many state transitions. If we assume that execution of semi-atomic actions takes one execution cycle, then execution of an action that results in an infinitely non-terminating event takes more than k execution cycles, for all $k \in \mathbb{N}$. To successfully define the termination predicate according to this intuition we define for basic actions the notion of one action being the *prefix* of another action, and the notion of the *norm* of an action, representing the number of execution cycles it would take an agent to execute the action.
The prefix predicate is necessary to successfully deal with infinite while-actions; the need for this predicate is caused by the definition of the computation sequences for the while-actions. If we consider for instance the action $\alpha \stackrel{\text{def}}{=}$ `while` **tt** `do skip od` in a given state s for a given agent i, it is obvious that execution of α by i results in infinitely many state transitions from s to s. However, the set of finite computation sequences of α fails to express this: since all finite computation sequences of α are of the form α'; `confirm` **ff**, all of these computation sequences result in voidly non-terminating events!
The definition of *Term* as we give it, i.e., using the prefix predicate, can cope with the situation sketched above: for all natural numbers k a natural number $n \geq k$ exists such that for some finite computation sequence $\alpha' \in CS($`while` **tt** `do skip od`$)$ the prefix of α' of length n results in n state transitions from s to s, and hence the event $\mathrm{do}_i($`while` **tt** `do skip od`$)$ is infinitely non-terminating in all states s, for all agents i.

After this, hopefully explanatory, introduction the actual definitions of the various predicates can be given.

6.5. Definition. The relation $\mathit{Prefix} \subseteq Ac_b^E \times Ac_b^E$ is defined as follows, where $\alpha, \beta, \gamma \in Ac_b^E$.

(1) $\mathit{Prefix}(\alpha, \alpha)$.

(2) if $\mathit{Prefix}(\alpha, \beta)$ then $\mathit{Prefix}(\alpha, \beta; \gamma)$.

(3) $\mathit{Prefix}(\alpha; \beta, \alpha; \gamma)$ iff $\mathit{Prefix}(\beta, \gamma)$.

6.6. Definition. The function $|.| : Ac_b^E \to \mathbb{N}$, denoting the norm of an action, is defined by:

$|\alpha| \quad = 1$ if α is semi-atomic

$|\alpha_1;\alpha_2| = |\alpha_1| + |\alpha_2|$

6.7. DEFINITION. For all actions $\alpha, \alpha' \in Ac_b^E$, and for all $n \in \mathbb{N}$ we define $|\alpha|_n$ by:

$$|\alpha|_n = \alpha' \Leftrightarrow \mathit{Prefix}(\alpha', \alpha) \,\&\, |\alpha'| = n$$

6.8. DEFINITION. The function *Term*, indicating termination of a given action, in a given state and for a given agent, is defined as follows.

$\mathit{Term}(i, \alpha, s) = \mathbf{0}$ if
$\quad \forall k \exists n \geq k \exists \alpha' \exists \alpha''[\alpha'' = |\alpha'|_n\,,\ \alpha' \in \mathit{CS}(\alpha)$ and $\mathtt{r}(i, \alpha'')(s) \neq \emptyset]$
$\mathit{Term}(i, \alpha, s) = \mathbf{1}$ otherwise.

6.9. REMARK. Note that the definition of *Term* as given above is correct only since the nondeterminism that is considered in this paper is *bounded*, i.e., no infinite branching inside a nondeterministic choice is possible (cf. [AP86]). It is not hard to see that for bounded nondeterminism it indeed holds that $\mathit{Term}(i, \alpha, s) = \mathbf{0}$ implies that $\mathrm{do}_i(\alpha)$ is infinitely non-terminating in s. For unbounded nondeterminism this implication is in general not valid.

6.10. DEFINITION. The function $CS : Ac_E \to Ac_b^E$ is defined as in 5.5, where the clause $\mathit{CS}(\alpha_1 \oplus \alpha_2) = \mathit{CS}(\alpha_1) \cup \mathit{CS}(\alpha_2)$ is replaced by $\mathit{CS}(\alpha_1 + \alpha_2) = \mathit{CS}(\alpha_1) \cup \mathit{CS}(\alpha_2)$.

6.11. DEFINITION. Let $\mathcal{M}$ be some standard model. The functions CR, π, $\mathtt{r}$, and $\mathtt{c}$ are by simultaneous induction defined as follows:

The function CR, denoting the *finite computation runs*, is defined by:

$$\begin{array}{lcl}
CR & : & \mathcal{A} \times Ac_E \times \mathcal{S} \to \wp(Ac_b^E) \\
CR(i, \alpha, s) & = & \{\mathtt{fail}\} \text{ if } \mathit{Term}(i, \alpha, s) = \mathbf{0} \\
\text{else if } \mathit{Term}(i, \alpha, s) = \mathbf{1}\text{:} & & \\
CR(i, a, s) & = & \{a\} \\
CR(i, \mathtt{confirm}\ \varphi, s) & = & \{\mathtt{confirm}\ \varphi\} \\
CR(i, \alpha_1;\alpha_2, s) & = & \{\alpha_1';\alpha_2' \mid \alpha_1' \in CR(i, \alpha_1, s), \\
 & & \qquad \alpha_2' \in CR(i, \alpha_2, \mathtt{r}(i, \alpha_1')(s))\} \\
CR(i, \mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi}, s) & = & CR(i, \mathtt{confirm}\ \varphi;\alpha_1, s) \text{ if } \pi(\varphi, s) = \mathbf{1} \\
 & = & CR(i, \mathtt{confirm}\ \neg\varphi;\alpha_2, s) \text{ if } \pi(\varphi, s) = \mathbf{0} \\
CR(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s) & = & \{\mathtt{confirm}\ \neg\varphi\} \text{ if } \pi(\varphi, s) = \mathbf{0} \\
 & = & \{\alpha' \in \mathit{CS}(\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}) \mid \\
 & & \qquad (\alpha' = (\mathtt{confirm}\ \varphi;\gamma_1);\gamma),\ \gamma_1 \in CR(i, \alpha_1, s), \\
 & & \qquad\quad \gamma \in CR(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, \mathtt{r}(i, \gamma_1)(s))\} \\
 & & \qquad\qquad \text{if } \pi(\varphi, s) = \mathbf{1} \\
CR(i, \alpha_1 + \alpha_2, s) & = & CR(i, \alpha_1, s) \cup CR(i, \alpha_2, s) \\
\text{and} & & \\
CR(i, \alpha, \emptyset) & = & \mathit{CS}(\alpha)
\end{array}$$

the function π as given in definition 2.6 is for events and abilities modified as follows:

$$\begin{array}{ll}
\pi(<\mathrm{do}_i(\alpha)>\varphi, s) = \mathbf{1} & \Leftrightarrow \forall \alpha' \in CR(i,\alpha,s) \exists s' \in \mathcal{S}[\mathtt{r}(i,\alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}] \\
\pi(\mathbf{A}_i\alpha, s) = \mathbf{1} & \Leftrightarrow \forall \alpha' \in CR(i,\alpha,s)[\mathtt{c}(i,\alpha')(s) = \mathbf{1}]
\end{array}$$

where

$$\begin{array}{ll}
\mathtt{r}(i, \mathtt{confirm}\,\varphi)(s) & = \{s\} \text{ if } \pi(\varphi, s) = \mathbf{1} \\
 & = \emptyset \text{ otherwise} \\
\mathtt{r}(i, \alpha_1; \alpha_2)(s) = s' & \Leftrightarrow \exists t \in \mathcal{S}[\mathtt{r}(i,\alpha_1)(s) = t \,\&\, \mathtt{r}(i,\alpha_2)(t) = s']
\end{array}$$

and

$$\mathtt{r}(i,\alpha)(\emptyset) = \emptyset$$

$$\begin{array}{ll}
\mathtt{c}(i, \mathtt{confirm}\,\varphi)(s) & = \pi(\varphi, s) \\
\mathtt{c}(i, \alpha_1; \alpha_2)(s) & = \mathtt{c}(i,\alpha_1)(s) \wedge \mathtt{c}(i,\alpha_2)(\mathtt{r}(i,\alpha_1)(s))
\end{array}$$

and

$$\mathtt{c}(i,\alpha)(\emptyset) = \mathbf{1}$$

6.12. Remark. Note that well-definedness of $CR(i,\alpha,s)$ in definition 6.11 depends essentially on termination of the event $\mathrm{do}_i(\alpha)$ in the state s.

6.13. Remark. The definition of $CR(i,\alpha,\emptyset)$ may seem to be a somewhat arbitrary one at this point: replacing $CS(\alpha)$ by for instance Ac_b^E or by $\{\mathtt{skip}\}$ would not alter the behaviour of the π function. However the definition as it is given is such that some interesting and intuitively desirable relations exist between the set of finite computation runs and the set of finite computation sequences (see theorems 6.18 and 6.20, lemmas 6.15 and 6.17, and corollaries 6.19 and 6.21).

6.14. Remark. Note that in the definition of the finite computation runs the conditional and the repetitive composition are not considered to be some sort of degenerated (external) nondeterministic choice, this in contrast with the usual approach in dynamic logic (cf. [HR83], [Har84], [KT90], [Gol92]). In our opinion the approach that we take in definition 6.11 is an intuitively better one than the standard dynamic logic approach. Since compared to the nondeterministic choice, the conditional does not comprise any real choice since the choice as to perform the then part or the else part is completely determined by the value of the condition; this is an *imposed* choice that has nothing to do with either the agent or the external environment surrounding the agent making a choice. An analogous line of reasoning can be given for the repetitive composition. The treatment of the conditional and the repetitive composition in definition 6.11 is based on our conviction that these action constructors are essentially deterministic by nature: in a given state s it is for a given conditional $\mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi}$ completely fixed whether α_1 or α_2 is performed. This depends only on the truth value of the conditional φ, and is by no means nondeterministic. Also for the repetitive composition it can be determined on beforehand whether the action in the body of the while will be performed, since this depends only on the truth or falsity of the φ in $\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}$. This intuition is clearly visible in the definitions of the finite computation runs for the conditional and the repetitive

composition. The fact that in section 5 a correspondence does seem to exist between the conditional $\texttt{if}\ \varphi\ \texttt{then}\ \alpha_1\ \texttt{else}\ \alpha_2\ \texttt{fi}$ on one side, and the internal nondeterministic choice $(\texttt{confirm}\ \varphi; \alpha_1) \oplus (\texttt{confirm}\ \neg\varphi; \alpha_2)$ on the other side, is a consequence of the fact that the imposed choice in the conditional has the same behaviour as the angelic nondeterministic choice with regard to the results of events and abilities.

The following lemmas and theorems sum up some of the properties of *Term*, *CS* and *CR*.

6.15. LEMMA. *For all models* $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$, *for all* $s \in \mathcal{S}$, *for all agents* i *and for all actions* $\alpha \in Ac_E$ *we have:*

- $CS(i, \alpha, s) \neq \emptyset$.
- $CR(i, \alpha, s) \neq \emptyset$.

PROOF OF LEMMA 6.15: Both cases are proved by induction on the structure of α.
$\boxtimes$

6.16. THEOREM. *For all models* $\mathcal{M}$, *for all* s *and for all actions* $\alpha_1, \alpha_2 \in Ac_E$ *and formula* φ *we have:*

- *if* $Term(i, \alpha_1; \alpha_2, s) = \mathbf{1}$ *then* $Term(i, \alpha_1, s) = \mathbf{1}$ *and* $Term(i, \alpha_2, s') = \mathbf{1}$ *for all* $s' = \mathtt{r}(i, \alpha_1')(s)$ *with* $\alpha_1' \in CS(\alpha_1)$.
- *if* $Term(i, \texttt{if}\ \varphi\ \texttt{then}\ \alpha_1\ \texttt{else}\ \alpha_2\ \texttt{fi}, s) = \mathbf{1}$ *then* $Term(i, \texttt{confirm}\ \varphi; \alpha_1, s) = \mathbf{1}$ *and* $Term(i, \texttt{confirm}\ \neg\varphi; \alpha_2, s) = \mathbf{1}$.
- *if* $Term(i, \texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}, s) = \mathbf{1}$ *then for all* α'' *such that* $Prefix(\alpha'', \alpha')$ *with* $\alpha' \in CS(\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od})$, *if* $\mathtt{r}(i, \alpha'')(s) = s'$ *then* $Term(i, \texttt{confirm}\ \varphi; \alpha_1, s') = \mathbf{1}$.
- *if* $Term(i, \alpha_1 + \alpha_2, s) = \mathbf{1}$ *then* $Term(i, \alpha_1, s) = \mathbf{1}$ *and* $Term(i, \alpha_2, s) = \mathbf{1}$.

PROOF OF THEOREM 6.16: The cases for the conditional $\texttt{if}\ \varphi\ \texttt{then}\ \alpha_1\ \texttt{else}\ \alpha_2\ \texttt{fi}$ and the external nondeterministic choice $\alpha_1 + \alpha_2$ are easily proved. We will show the two other cases.

(1) Case 1: Sequential Composition.
Firstly we will show that if $Term(i, \alpha_1, s) = \mathbf{0}$ then $Term(i, \alpha_1; \alpha_2, s) = \mathbf{0}$, and secondly that if for some $s' = \mathtt{r}(i, \alpha_1')(s)$, with $\alpha_1' \in CS(\alpha_1)$, it holds that $Term(i, \alpha_2, s') = \mathbf{0}$ then $Term(i, \alpha_1; \alpha_2, s) = \mathbf{0}$. With this the contraposition of the implication given above is proved.

- Assume that $Term(i, \alpha_1, s) = \mathbf{0}$. We will show that $Term(i, \alpha_1; \alpha_2, s) = \mathbf{0}$ by showing that for all $k \in \mathbb{N}$, an $n \geq k$ and actions α', α'' exist, such that $\alpha'' = |\alpha'|_n$, $\alpha' \in CS(\alpha_1; \alpha_2)$ and $\mathtt{r}(i, \alpha'')(s) \neq \emptyset$. Let $k \in \mathbb{N}$. Since $Term(i, \alpha_1, s) = \mathbf{0}$ some $n \geq k$, β', β'' exist such that $\beta'' = |\beta'|_n$, $\beta' \in CS(\alpha_1)$ and $\mathtt{r}(i, \beta'')(s) \neq \emptyset$. Now take an arbitrary $\alpha_2' \in CS(\alpha_2)$; such an α_2' exists since $CS(\alpha_2) \neq \emptyset$. Now we have: $\beta'' = |\beta'; \alpha_2'|_n$, $n \geq k$, $\beta'; \alpha_2' \in CS(\alpha_1; \alpha_2)$ and $\mathtt{r}(i, \beta'')(s) \neq \emptyset$. Since k was chosen arbitrary in $\mathbb{N}$ this suffices to conclude that $Term(i, \alpha_1; \alpha_2, s) = \mathbf{0}$.

- Assume that $s' = \mathtt{r}(i, \alpha_1')(s)$, for $\alpha_1' \in CS(\alpha_1)$ is such that $Term(i, \alpha_2, s') = \mathbf{0}$. As in the previous case we will show that $Term(i, \alpha_1; \alpha_2, s) = \mathbf{0}$ by showing that for all $k \in \mathbb{N}$, an $n \geq k$ and actions α', α'' exist, such that $\alpha'' = |\alpha'|_n$, $\alpha' \in CS(\alpha_1; \alpha_2)$ and $\mathtt{r}(i, \alpha'')(s) \neq \emptyset$. Let $k \in \mathbb{N}$. Since $Term(i, \alpha_2, s') = \mathbf{0}$ some $n \geq k$, γ', γ'' exist such that $\gamma'' = |\gamma'|_n$, $\gamma' \in CS(\alpha_2)$ and $\mathtt{r}(i, \gamma'')(s) \neq \emptyset$. It now holds that $\alpha_1'; \gamma' \in CS(\alpha_1; \alpha_2)$, $|\alpha_1'; \gamma''| = n' \geq n$, and $\mathtt{r}(i, \alpha_1'; \gamma'')(s) \neq \emptyset$. Since k was chosen arbitrary in $\mathbb{N}$ it follows that $Term(i, \alpha_1; \alpha_2, s) = \mathbf{0}$.

This suffices to conclude that case 1 holds.

(2) Case 3: Repetitive Composition.
Assume that $Term(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s) = \mathbf{1}$. Let $k_0 \in \mathbb{N}$ be such that $\forall n \geq k_0 \forall \alpha' \forall \alpha''[\alpha'' = |\alpha'|_n \,\&\, \alpha' \in CS(\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}) \Rightarrow \mathtt{r}(i, \alpha'')(s) = \emptyset]$. Assume that s', α'' and α' are such that $Prefix(\alpha'', \alpha')$, $\alpha' \in CS(\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od})$, and $\mathtt{r}(i, \alpha'')(s) = s'$.
Assume towards a contradiction that $Term(i, \mathtt{confirm}\ \varphi; \alpha_1, s') = \mathbf{0}$. Let $m \geq k_0$ and γ', γ'' be such that $\gamma'' = |\gamma'|_m$, $\gamma' \in CS(\mathtt{confirm}\ \varphi; \alpha_1)$ and $\mathtt{r}(i, \gamma'')(s') \neq \emptyset$.
Note that, since $Term(i, \mathtt{confirm}\ \varphi; \alpha_1, s') = \mathbf{0}$, it holds that $\pi(\varphi, s') = \mathbf{1}$.
Hence α'', which is such that $\mathtt{r}(i, \alpha'')(s) = s'$, has any of the following three forms:

- $\alpha'' = \mathtt{confirm}\ \varphi$, or
- $\alpha'' = (\mathtt{confirm}\ \varphi; \alpha_1'); \ldots; (\mathtt{confirm}\ \varphi; \alpha_l')$, for $l \geq 1$, or
- $\alpha'' = (\mathtt{confirm}\ \varphi; \alpha_1'); \ldots; (\mathtt{confirm}\ \varphi; \alpha_l'); \mathtt{confirm}\ \varphi$, for $l \geq 1$.

Now define the action δ in any of the three cases given above as follows:

- $\delta = \gamma''$.
- $\delta = (\mathtt{confirm}\ \varphi; \alpha_1'); \ldots; (\mathtt{confirm}\ \varphi; \alpha_l'); \gamma''$.
- $\delta = (\mathtt{confirm}\ \varphi; \alpha_1'); \ldots; (\mathtt{confirm}\ \varphi; \alpha_l'); \gamma''$.

It holds that $\delta = |\delta; \mathtt{confirm}\ \neg\varphi|_x$ for some $x \geq m \geq k_0$, $(\delta; \mathtt{confirm}\ \neg\varphi) \in CS(\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od})$, and $\mathtt{r}(i, \delta)(s) \neq \emptyset$. Since $x \geq k_0$ this contradicts the definition of k_0 as given above. It follows that $Term(i, \mathtt{confirm}\ \varphi; \alpha_1, s') = \mathbf{1}$, which was to be shown.

⊠

6.17. LEMMA. *For all models $\mathcal{M}$, for all s and for all actions $\alpha_1 \in Ac_E$ we have:*

$$Term(i, \alpha, s) = \mathbf{1} \Rightarrow CR(i, \alpha, s) \subseteq CS(\alpha)$$

PROOF OF LEMMA 6.17: By induction on the structure of α, using theorem 6.16.
⊠

6.18. THEOREM. *For all models $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$, for all $s \in \mathcal{S}$ and for all actions $\alpha \in Ac_E$:*

$$Term(i, \alpha, s) = \mathbf{1} \Rightarrow \forall \alpha' \forall s'[\alpha' \in CS(\alpha) \text{ and } \mathtt{r}(i, \alpha')(s) = s' \text{ iff } \alpha' \in CR(i, \alpha, s) \text{ and } \mathtt{r}(i, \alpha')(s) = s']$$

PROOF OF THEOREM 6.18: We will show by induction on $\alpha \in Ac_E$ that, given that $\mathit{Term}(i, \alpha, s) = \mathbf{1}$, it holds that $\forall \alpha' \forall s' [\alpha' \in \mathit{CS}(\alpha)$ and $\mathtt{r}(i, \alpha')(s) = s' \Rightarrow \alpha' \in \mathit{CR}(i, \alpha, s)$ and $\mathtt{r}(i, \alpha')(s) = s']$; the reverse implication is trivial since $\mathit{CR}(i, \alpha, s) \subseteq \mathit{CS}(\alpha)$ holds by lemma 6.17.

(1) α is semi-atomic. This case is trivial since for semi-atomic actions α it holds that $\mathit{CR}(i, \alpha, s) = \mathit{CS}(\alpha)$.

(2) $\alpha = \alpha_1; \alpha_2$. Let $\alpha' \in \mathit{CS}(\alpha)$, and let s' be such that $\mathtt{r}(i, \alpha')(s) = s'$. By definition of $\mathit{CS}(\alpha_1; \alpha_2)$, $\alpha' = \alpha'_1; \alpha'_2$, for some $\alpha'_1 \in \mathit{CS}(\alpha_1)$ and $\alpha'_2 \in \mathit{CS}(\alpha_2)$. Let $\alpha' = \alpha'_1; \alpha'_2$. From $\mathtt{r}(i, \alpha'_1; \alpha'_2)(s) = s'$ it follows that a $t \in \mathcal{S}$ exists such that $\mathtt{r}(i, \alpha'_1)(s) = t$ and $\mathtt{r}(i, \alpha'_2)(t) = s'$. Since $\mathit{Term}(i, \alpha_1; \alpha_2, s) = \mathbf{1}$ it follows that $\mathit{Term}(i, \alpha_1, s) = \mathbf{1}$, and we conclude by induction hypothesis that $\alpha'_1 \in \mathit{CR}(i, \alpha_1, s)$ and $\mathtt{r}(i, \alpha'_1)(s) = t$. Furthermore, from $\mathit{Term}(i, \alpha_1; \alpha_2, s) = \mathbf{1}$ it follows that $\mathit{Term}(i, \alpha_2, t) = \mathbf{1}$, and we conclude by the induction hypothesis that $\alpha'_2 \in \mathit{CR}(i, \alpha_2, t)$ and $\mathtt{r}(i, \alpha'_2)(t) = s'$. From $\alpha'_1 \in \mathit{CR}(i, \alpha_1, s)$, $\mathtt{r}(i, \alpha'_1)(s) = t$, $\alpha'_2 \in \mathit{CR}(i, \alpha_2, t)$ and $\mathtt{r}(i, \alpha'_2)(t) = s'$ we conclude that $\alpha'_1; \alpha'_2 \in \mathit{CR}(i, \alpha_1; \alpha_2, s)$ and $\mathtt{r}(i, \alpha'_1; \alpha'_2)(s) = s'$, which was to be shown.

(3) $\alpha = \mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi}$. Let $\alpha' \in \mathit{CS}(\alpha)$, and let s' be such that $\mathtt{r}(i, \alpha')(s) = s'$. We distinguish two cases:

- $\pi(\varphi, s) = \mathbf{1}$: from $\mathtt{r}(i, \alpha')(s) = s'$ it follows that $\alpha' \in \mathit{CS}(\mathtt{confirm}\ \varphi; \alpha_1)$; for if $\alpha' \in \mathit{CS}(\mathtt{confirm}\ \neg\varphi; \alpha_2)$ then $\mathtt{r}(i, \alpha')(s) = \emptyset$. Hence $\alpha' = \mathtt{confirm}\ \varphi; \alpha'_1$ for some $\alpha'_1 \in \mathit{CS}(\alpha_1)$. Since $\pi(\varphi, s) = \mathbf{1}$ it follows that $\mathtt{r}(i, \alpha'_1)(s) = s'$. Furthermore from $\pi(\varphi, s) = \mathbf{1}$ and $\mathit{Term}(i, \mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi}, s) = \mathbf{1}$ it follows that $\mathit{Term}(i, \alpha_1, s)$. By induction hypothesis it follows that $\alpha'_1 \in \mathit{CR}(i, \alpha_1, s)$ and $\mathtt{r}(i, \alpha'_1)(s) = s'$.
  Then $\mathtt{confirm}\ \varphi; \alpha'_1 \in \mathit{CR}(i, \mathtt{confirm}\ \varphi; \alpha_1, s) = \mathit{CR}(i, \mathtt{if}\ \varphi\ \mathtt{then}\ \alpha_1\ \mathtt{else}\ \alpha_2\ \mathtt{fi}, s)$ (the identity holds since $\pi(\varphi, s) = \mathbf{1}$) and $\mathtt{r}(i, \mathtt{confirm}\ \varphi; \alpha'_1)(s) = s'$, which was to be shown.
- $\pi(\neg\varphi, s) = \mathbf{1}$: completely analogous to the case where $\pi(\varphi, s) = \mathbf{1}$.

(4) $\alpha = \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}$. Let $\alpha' \in \mathit{CS}(\alpha)$, and let s' be such that $\mathtt{r}(i, \alpha')(s) = s'$. We distinguish two cases:

- $\pi(\neg\varphi, s) = \mathbf{1}$: from $\mathtt{r}(i, \alpha')(s) = s'$ it follows that $\alpha' = \mathtt{confirm}\ \neg\varphi$, and $s = s'$. Also $\mathit{CR}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s) = \{\mathtt{confirm}\ \neg\varphi\}$, and hence $\mathtt{r}(i, \alpha')(s) = s$ and $\alpha' \in \mathit{CR}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s)$.
- $\pi(\varphi, s) = \mathbf{1}$: then $\alpha' = \Pi_{j=1}^{k}(\varphi, \alpha'_j)$, where $k \geq 1$ and $\alpha'_j \in \mathit{CS}(\alpha_1)$. Let $s_1 \ldots s_{k+1}$ be such that $s = s_1$, $s_{k+1} = s'$ and for all $j \leq k$, $\mathtt{r}(i, \mathtt{confirm}\ \varphi; \alpha'_j)(s_j) = s_{j+1}$. We will show by finite backwards induction that for all $1 \leq j \leq k$ it holds that $\Pi_{i=j}^{k}(\varphi, \alpha'_i) \in \mathit{CR}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_j)$.
  Basis: $j = k$.
  Since $\pi(\neg\varphi, s_{k+1}) = \mathbf{1}$ we have $\mathit{CR}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_{k+1}) = \{\mathtt{confirm}\ \neg\varphi\}$. From $\mathit{Term}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s) = \mathbf{1}$ it follows that $\mathit{Term}(i, \alpha_1, s_k) = \mathbf{1}$ and since $\mathtt{r}(i, \alpha'_k)(s_k) = s_{k+1}$ it follows by the outmost induction that $\alpha'_k \in \mathit{CR}(i, \alpha_1, s_k)$ and $\mathtt{r}(i, \alpha'_k)(s_k) = s_{k+1}$.
  Hence $(\mathtt{confirm}\ \varphi; \alpha'_k); \mathtt{confirm}\ \neg\varphi \in \mathit{CR}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_k)$.
  Induction step: $j + 1 \mapsto j$.
  Assume that $\Pi_{i=j+1}^{k}(\varphi, \alpha'_i) \in \mathit{CR}(i, \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_{j+1})$.

Now since $Term(i, \texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}, s) = \mathbf{1}$ by theorem 6.16 also $Term(i, \alpha_1, s_j) = \mathbf{1}$. Furthermore since $\alpha'_j \in CS(\alpha_1)$ and $\mathtt{r}(i, \alpha'_j)(s_j) = s_{j+1}$ it follows by the outmost induction that $\alpha'_j \in CR(i, \alpha_1, s_j)$ and $\mathtt{r}(i, \alpha'_j)(s_j) = s_{j+1}$. Since $\pi(\varphi, s_j) = \mathbf{1}$ it follows that $(\texttt{confirm}\ \varphi; \alpha'_j); \Pi_{i=j+1}^{k}(\varphi, \alpha'_i) \in CR(i, \texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}, s_j)$.
By induction it holds that $\Pi_{i=j}^{k}(\varphi, \alpha'_i) \in CR(i, \texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}, s_j)$ for all $1 \leq j \leq k$,, and hence in particular $\alpha' = \Pi_{i=1}^{k}(\varphi, \alpha'_i) \in CR(i, \texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}, s)$ and $\mathtt{r}(i, \alpha')(s) = s'$, which was to be shown.

(5) $\alpha = \alpha_1 + \alpha_2$. Let $\alpha' \in CS(\alpha)$, and let s' be such that $\mathtt{r}(i, \alpha')(s) = s'$. Now either $\alpha' \in CS(\alpha_1)$ and then from $Term(i, \alpha_1 + \alpha_2, s) = \mathbf{1} \Rightarrow Term(i, \alpha_1, s) = \mathbf{1}$ and $\mathtt{r}(i, \alpha')(s) = s'$ it follows by induction hypothesis that $\alpha' \in CR(i, \alpha_1, s) \subseteq CR(i, \alpha_1 + \alpha_2, s)$, or $\alpha' \in CS(\alpha_2)$ and then from $Term(i, \alpha_1 + \alpha_2, s) = \mathbf{1} \Rightarrow Term(i, \alpha_2, s) = \mathbf{1}$ and $\mathtt{r}(i, \alpha')(s) = s'$ it follows by induction hypothesis that $\alpha' \in CR(i, \alpha_2, s) \subseteq CR(i, \alpha_1 + \alpha_2, s)$. In both cases $\alpha' \in CR(i, \alpha_1 + \alpha_2, s)$ and $\mathtt{r}(i, \alpha')(s) = s'$, which was to be shown.

⊠

6.19. Corollary. *For all models* $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$, *for all* $s \in \mathcal{S}$ *and for all actions* $\alpha \in Ac_E$*:*

$$Term(i, \alpha, s) = \mathbf{1} \Rightarrow \forall \alpha' \forall s' [\alpha' \in CS(\alpha)\ \textit{and}\ \mathtt{r}(i, \alpha')(s) = s'\ \textit{and}\ \mathtt{c}(i, \alpha')(s) = \mathbf{1}\ \textit{iff}\ \alpha' \in CR(i, \alpha, s)\ \textit{and}\ \mathtt{r}(i, \alpha')(s) = s'\ \textit{and}\ \mathtt{c}(i, \alpha')(s) = \mathbf{1}]$$

Proof of corollary 6.19: Directly from theorem 6.18.
⊠

6.20. Theorem. *For all models* $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$, *for all* $s \in \mathcal{S}$ *and for all actions* $\alpha \in Ac_E$*:*

$$Term(i, \alpha, s) = \mathbf{1} \Rightarrow \forall \alpha' [\alpha' \in CS(\alpha)\ \textit{and}\ \mathtt{r}(i, \alpha')(s) = \emptyset\ \textit{and}\ \mathtt{c}(i, \alpha')(s) = \mathbf{1}\ \textit{iff}\ \alpha' \in CR(i, \alpha, s)\ \textit{and}\ \mathtt{r}(i, \alpha')(s) = \emptyset\ \textit{and}\ \mathtt{c}(i, \alpha')(s) = \mathbf{1}]$$

Proof of theorem 6.20: The proof of this theorem proceeds in a way similar to that of theorem 6.18. We will show the case where $\alpha = \texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}$, the other cases are left to the reader. As in the proof of theorem 6.18 we show only the left to right implication: the other one is trivial by lemma 6.17.

- $\alpha = \texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}$. Assume that $\alpha' \in CS(\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od})$, $\mathtt{r}(i, \alpha')(s) = \emptyset$, and $\mathtt{c}(i, \alpha')(s) = \mathbf{1}$. Obviously $\alpha' \neq \texttt{confirm}\ \neg\varphi$, since in that case $\mathtt{r}(i, \alpha')(s) = \emptyset$ and $\mathtt{c}(i, \alpha')(s) = \mathbf{1}$ would conflict. Hence $\alpha' = \Pi_{j=1}^{k}(\varphi, \alpha'_i)$, where $k \geq 1$ and $\alpha'_j \in CS(\alpha_1)$. Determine the greatest $m < k$ such that $\mathtt{r}(i, \texttt{confirm}\,\varphi; \alpha'_1); \ldots; (\texttt{confirm}\,\varphi; \alpha'_m))(s) = t$, for $t \in \mathcal{S}$, and $\mathtt{r}(i, (\texttt{confirm}\ \varphi; \alpha'_{m+1})(t) = \emptyset$. For such an m to exist it is necessary that $\mathtt{r}(i, (\texttt{confirm}\ \varphi; \alpha'_1); \ldots; (\texttt{confirm}\ \varphi; \alpha'_k))(s) = \emptyset$. But this is indeed the case: for would $\mathtt{r}(i, (\texttt{confirm}\ \varphi; \alpha'_1); \ldots; (\texttt{confirm}\ \varphi; \alpha'_k))(s) = u$, then $\pi(\varphi, u) = \mathbf{0}$ since $\mathtt{r}(i, \alpha')(s) = \emptyset$ whereas $\mathtt{c}(i, \alpha')(s) = \mathbf{1}$ forces $\pi(\varphi, u) = \mathbf{1}$.
Let $s_1 \ldots s_{m+1}$ be such that $s = s_1$, $s_{m+1} = t$, and $\mathtt{r}(i, \texttt{confirm}\ \varphi; \alpha'_n)(s_n) = s_{n+1}$ for

all $n \leq m$. Note that from $\mathtt{c}(i,\alpha')(s) = \mathbf{1}$ it follows that $\mathtt{c}(i,\varphi;\alpha'_n)(s_n) = \mathbf{1}$ for all $n \leq m+1$. We show by a finite backwards induction that for all $1 \leq n \leq m+1$ it holds that $\Pi^k_{j=n}(\varphi,\alpha'_j) \in CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_n)$.

Basis: $n = m+1$.
Note that $\pi(\varphi, s_{m+1}) = \mathbf{1}$. Now since $Term(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s) = \mathbf{1}$ it follows by theorem 6.16 that also $Term(i,\mathtt{confirm}\ \varphi;\alpha_1, s_{m+1}) = \mathbf{1}$, and since $\pi(\varphi, s_{m+1}) = \mathbf{1}$, also $Term(i,\alpha_1,s_{m+1}) = \mathbf{1}$.
From the fact that $\alpha'_{m+1} \in CS(\alpha_1)$, $Term(i,\alpha_1,s) = \mathbf{1}$, $\mathtt{r}(i,\alpha'_{m+1})(s_{m+1}) = \emptyset$, and $\mathtt{c}(i,\alpha'_{m+1})(s_{m+1}) = \mathbf{1}$ it follows by the outmost induction that $\alpha'_{m+1} \in CR(i,\alpha_1,s_{m+1})$, $\mathtt{r}(i,\alpha'_{m+1})(s_{m+1}) = \emptyset$, and $\mathtt{c}(i,\alpha'_{m+1})(s_{m+1}) = \mathbf{1}$.
Now $CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, \mathtt{r}(i,\alpha'_{m+1})(s_{m+1})) = CS(\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od})$, which implies that $\Pi^k_{j=m+2}(\varphi,\alpha'_j) \in CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, \mathtt{r}(i,\alpha'_{m+1})(s_{m+1}))$.
Thus $\Pi^k_{j=m+1}(\varphi,\alpha'_j) \in CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_{m+1})$.

Induction step: $n+1 \mapsto n$.
Assume that $\Pi^k_{j=n+1}(\varphi,\alpha'_j) \in CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_{n+1})$.
Now since $Term(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s) = \mathbf{1}$ also $Term(i,\alpha_1,s_n) = \mathbf{1}$. Furthermore since $\alpha'_n \in CS(\alpha_1)$ and $\mathtt{r}(i,\alpha'_n)(s_n) = s_{n+1}$ it follows by theorem 6.18 that $\alpha'_n \in CR(i,\alpha_1,s_n)$ and $\mathtt{r}(i,\alpha'_n)(s_n) = s_{n+1}$. Since $\pi(\varphi,s_n) = \mathbf{1}$ it follows that $\Pi^k_{j=n}(\varphi,\alpha'_j) \in CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_n)$.

By induction, $\Pi^k_{j=n}(\varphi,\alpha'_j) \in CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s_n)$, for all $1 \leq n \leq m+1$. Hence in particular $\alpha' = \Pi^k_{j=1}(\varphi,\alpha'_j) \in CR(i,\mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}, s)$. Since also $\mathtt{r}(i,\alpha')(s) = \emptyset$ and $\mathtt{c}(i,\alpha')(s) = \mathbf{1}$ we conclude that for $\alpha = \mathtt{while}\ \varphi\ \mathtt{do}\ \alpha_1\ \mathtt{od}$ the theorem is correct.

⊠

6.21. Corollary. *For all models* $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$, *for all* $s \in \mathcal{S}$ *and for all actions* $\alpha \in Ac_E$:

$$Term(i,\alpha,s) = \mathbf{1} \Rightarrow \forall\alpha'[\alpha' \in CS(\alpha)\ \textit{and}\ \mathtt{c}(i,\alpha')(s) = \mathbf{1}\ \textit{iff}\ \alpha' \in CR(i,\alpha,s)\ \textit{and}\ \mathtt{c}(i,\alpha')(s) = \mathbf{1}]$$

Proof of corollary 6.21: The corollary follows from corollary 6.18 and theorem 6.20.
⊠

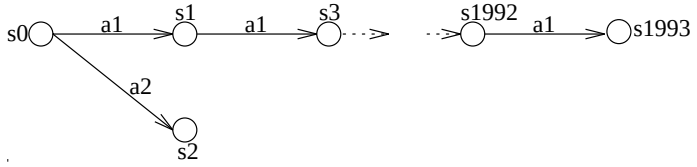
6.22. Examples. The following examples show how the definition of finite computation runs as we give it behaves in practice.

- Consider the Kripke model $\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$ as given in example 4.2. Consider the ability of agent i for the action $\alpha = (a_1 + a_2); a_3$:

  $\mathcal{M}, s_0 \models \mathbf{A}_i\alpha$
  $\Leftrightarrow \pi(\mathbf{A}_i\alpha, s_0) = \mathbf{1}$
  $\Leftrightarrow \forall\alpha' \in CR(i,\alpha,s_0)[\mathtt{c}(i,\alpha')(s_0) = \mathbf{1}]$
  $\Leftrightarrow \mathtt{c}(i,a_1;a_3)(s_0) = \mathbf{1}$ and $\mathtt{c}(i,a_2;a_3)(s_0) = \mathbf{1}$
  $\Leftrightarrow (\mathtt{c}(i,a_1)(s_0) = \mathbf{1}$ and $\mathtt{c}(i,a_3)(s_1) = \mathbf{1})$ and
  $(\mathtt{c}(i,a_2)(s_0) = \mathbf{1}$ and $\mathtt{c}(i,a_3)(\emptyset) = \mathbf{1})$
  $\Leftrightarrow$ false, since $\mathtt{c}(i,a_3)(s_1) = \mathbf{0}$.

Since the agent is not capable of performing all of the atomic sequences that constitute the action α, s/he is not able to perform α. This is exactly the outcome as we intuitively expected it to be

- Assume that $\mathcal{M}$ is the Kripke model for which the atomic actions are for agent i as given in the following picture.

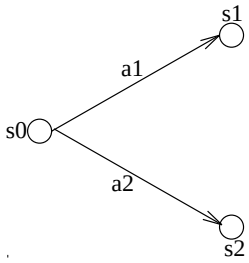


Assume furthermore that p holds in all states except s_2 and s_{1993}.

Let action α be given by $\alpha \stackrel{\text{def}}{=} \texttt{while}\ p\ \texttt{do}\ a_1 + a_2\ \texttt{od}$. To determine the computation runs of α for agent i in state s_0, first note that $\mathit{Term}(i, \alpha, s_0)$ holds, since for $k = 3986$ (note that $|(\texttt{confirm}\ p; a_1)^{1992}; \texttt{confirm}\ \neg p)| = 3985$) it holds that for all $n \geq k$ no action $\alpha' \in \mathit{CS}(\alpha)$ exists, such that $\texttt{r}(i, |\alpha'|_n)(s_0) \neq \emptyset$.
It now holds that $\mathit{CR}(i, \alpha, s_0) = \{\alpha', \alpha''\}$ where:

(1) $\alpha' = (\texttt{confirm}\ p; a_2); \texttt{confirm}\ \neg p$.
(2) $\alpha'' = (\texttt{confirm}\ p; a_1)^{1992}; \texttt{confirm}\ \neg p$.

- Consider the Kripke model $\mathcal{M}$ such that $\texttt{r}$ for agent i is as given in the following picture, and such that $\pi(p, s_j) = \mathbf{1}$ for $j = 0, 1, 2$ and $\texttt{c}(i, a_1)(s_j) = \texttt{c}(i, a_2)(s_j) = \mathbf{1}$ for $j = 0, 1, 2$.



Let $\alpha = \texttt{while}\ p\ \texttt{do}\ a_1 + a_2\ \texttt{od}$. Now we have:

6.23. Proposition. *In $\mathcal{M}$ it holds that*

$$\mathit{CR}(i, \alpha, s_0) = \{(\texttt{confirm}\ p; \alpha'_1); (\texttt{confirm}\ p; \alpha'_2); \beta \mid \alpha'_1, \alpha'_2 \in \{a_1, a_2\}, \beta \in \mathit{CS}(\alpha)\}$$

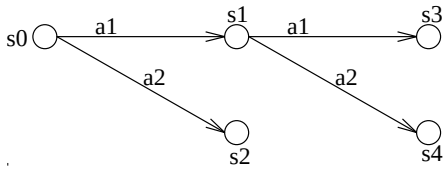
Proof of proposition 6.23: First note that $\mathit{Term}(i, \alpha, s_0) = \mathbf{1}$: for $k = 5$ it holds that for all $n \geq k$ no action $\alpha' \in \mathit{CS}(\alpha)$ exists such that $\texttt{r}(i, |\alpha'|_n)(s_0) \neq \emptyset$. Furthermore it holds that:

(1) $\mathit{CR}(i, \alpha, \texttt{r}(i, a_1)(s_1)) = \mathit{CR}(i, \alpha, \texttt{r}(i, a_2)(s_1)) = \mathit{CR}(i, \alpha, \texttt{r}(i, a_1)(s_2)) = \mathit{CR}(i, \alpha, \texttt{r}(i, a_2)(s_2)) = \mathit{CS}(\alpha)$, since $\texttt{r}(i, a_1)(s_1) = \ldots = \texttt{r}(i, a_2)(s_2) = \emptyset$.
(2) $\pi(p, s_1) = \pi(p, s_2) = \mathbf{1}$ and $\mathit{CR}(i, a_1 + a_2, s_1) = \mathit{CR}(i, a_1 + a_2, s_2) = \{a_1, a_2\}$. Hence $\mathit{CR}(i, \alpha, s_1) = \mathit{CR}(i, \alpha, s_2) = \{(\texttt{confirm}\ p; \alpha'_2); \alpha' \mid \alpha'_2 \in \{a_1, a_2\}, \alpha' \in \mathit{CS}(\alpha)\}$.
(3) $\pi(p, s_0) = \mathbf{1}$ and $\mathit{CR}(i, a_1 + a_2, s_0) = \{a_1, a_2\}$.

Hence $\mathit{CR}(i, \alpha, s_0) = \{(\texttt{confirm}\ p; \alpha'_1); (\texttt{confirm}\ p; \alpha'_2); \beta \mid \alpha_1, \alpha_2 \in \{a_1, a_2\}, \beta \in \mathit{CS}(\alpha)\}$.
⊠

As a consequence of proposition 6.23 we have that it holds that $\mathcal{M}, s_0 \models \mathbf{A}_i\alpha$ and $\mathcal{M}, s_0 \not\models <\mathrm{do}_i(\alpha)>\mathbf{tt}$.

- Consider the Kripke model $\mathcal{M}$ as given in the following picture and such that $\pi(p, s_0) = \pi(q, s_0) = \pi(p, s_1) = \mathbf{1}$ and $\pi(q, s_j) = \mathbf{0}$ for $j = 1, \ldots, 4$ and $\pi(p, s_j) = \mathbf{0}$ for $j = 2, 3, 4$.



Consider the ability of an agent i for the action α, given by $\alpha \stackrel{\text{def}}{=} \texttt{while}\ p\ \texttt{do}\ \texttt{confirm}\ q + (a_1; a_2)\ \texttt{od}$. In order to determine $CR(i, \alpha, s_0)$ we first check if $Term(i, \alpha, s_0)$ holds. Take an arbitrary $k \in \mathbb{N}$. Note that $\alpha' \stackrel{\text{def}}{=} (\texttt{confirm}\ p; \texttt{confirm}\ q)^{k+1}; \texttt{confirm}\ \neg p \in Seq_{k+1}(\alpha) \subseteq CS(\alpha)$. Now $|\alpha'|_{2k+2} = (\texttt{confirm}\, p; \texttt{confirm}\, q)^{k+1}$, and $\mathtt{r}(i, |\alpha'|_{2k+2})(s_0) = s_0$. Thus $Term(i, \alpha, s_0) = \mathbf{0}$, and $CR(i, \alpha, s_0) = \{\texttt{fail}\}$. Hence the agent is capable of performing α in s_0 iff s/he is able to perform `fail`, and we conclude that the agent is not capable of performing α.

The following theorem shows that our semantics for the external nondeterministic choice behaves as desired.

6.24. Theorem. *For all agents i, actions $\alpha_1, \alpha_2 \in Ac_E$, and for all formulae φ we have:*

- $\models <\mathrm{do}_i(\alpha_1 + \alpha_2)>\varphi \leftrightarrow (<\mathrm{do}_i(\alpha_1)>\varphi \wedge <\mathrm{do}_i(\alpha_2)>\varphi)$.
- $\models \mathbf{A}_i(\alpha_1 + \alpha_2) \leftrightarrow (\mathbf{A}_i\alpha_1 \wedge \mathbf{A}_i\alpha_2)$.

Proof of theorem 6.24:

First case:

$\mathcal{M}, s \models <\mathrm{do}_i(\alpha_1 + \alpha_2)>\varphi$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1 + \alpha_2)>\varphi, s) = \mathbf{1}$
$\Leftrightarrow \forall \alpha' \in CR(i, \alpha_1 + \alpha_2, s) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \forall \alpha' \in CR(i, \alpha_1, s) \cup CR(i, \alpha_2, s) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \forall \alpha' \in CR(i, \alpha_1, s) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$ and
$\forall \alpha' \in CR(i, \alpha_2, s) \exists s' \in \mathcal{S}[\mathtt{r}(i, \alpha')(s) = s' \,\&\, \pi(\varphi, s') = \mathbf{1}]$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1)>\varphi, s) = \mathbf{1}$ and $\pi(<\mathrm{do}_i(\alpha_2)>\varphi, s) = \mathbf{1}$
$\Leftrightarrow \pi(<\mathrm{do}_i(\alpha_1)>\varphi \wedge <\mathrm{do}_i(\alpha_2)>\varphi, s) = \mathbf{1}$
$\Leftrightarrow \mathcal{M}, s \models <\mathrm{do}_i(\alpha_1)>\varphi \wedge <\mathrm{do}_i(\alpha_2)>\varphi$

Second case:

$\mathcal{M}, s \models \mathbf{A}_i(\alpha_1 + \alpha_2)$
$\Leftrightarrow \pi(\mathbf{A}_i(\alpha_1 + \alpha_2), s) = \mathbf{1}$
$\Leftrightarrow \forall \alpha' \in CR(i, \alpha_1 + \alpha_2, s)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$
$\Leftrightarrow \forall \alpha' \in CR(i, \alpha_1, s) \cup CR(i, \alpha_2, s)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$
$\Leftrightarrow \forall \alpha' \in CR(i, \alpha_1, s)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$ and $\forall \alpha' \in CR(i, \alpha_2, s)[\mathtt{c}(i, \alpha')(s) = \mathbf{1}]$
$\Leftrightarrow \pi(\mathbf{A}_i\alpha_1, s) = \mathbf{1}$ and $\pi(\mathbf{A}_i\alpha_2, s) = \mathbf{1}$

$\Leftrightarrow \pi(\mathbf{A}_i\alpha_1 \wedge \mathbf{A}_i\alpha_2, s) = \mathbf{1}$
$\Leftrightarrow \mathcal{M}, s \models (\mathbf{A}_i\alpha_1 \wedge \mathbf{A}_i\alpha_2)$

$\boxtimes$

6.1. Composite actions revisited

As in section 5.1, we will reconsider the validities proved in [HLM93], this time in the light of the semantics given for the external nondeterministic choice.

It turns out that for events the same validities hold as found for the internal nondeterministic choice, i.e., theorem 5.13 also holds for the semantics given in definition 6.11. Also theorem 5.14 holds for the semantics for the external nondeterministic choice, this completely in accordance with our intuition.

As was the case with the internal nondeterministic choice, the behaviour of the sequential composition and the repetitive composition differs from that observed in [HLM93]. Furthermore, the behaviour of the sequential composition is for the semantics of definition 6.11 different from the behaviour for the semantics of section 5.

6.25. THEOREM. *For all agents i, actions $\alpha_1, \alpha_2, \alpha_3 \in Ac_I$, and for all formulae φ we have:*

(1) $\models \mathbf{A}_i(\alpha_1;\alpha_2) \rightarrow \mathbf{A}_i\alpha_1 \wedge [\mathrm{do}_i(\alpha_1)]\mathbf{A}_i\alpha_2$.
(2) $\not\models \mathbf{A}_i\alpha_1 \wedge [\mathrm{do}_i(\alpha_1)]\mathbf{A}_i\alpha_2 \rightarrow \mathbf{A}_i(\alpha_1;\alpha_2)$.
(3) $\models \mathbf{A}_i((\alpha_1 + \alpha_2);\alpha_3) \leftrightarrow (\mathbf{A}_i(\alpha_1;\alpha_3) \wedge \mathbf{A}_i(\alpha_2;\alpha_3))$.
(4) $\models \mathbf{A}_i(\alpha_1;(\alpha_2 + \alpha_3) \leftrightarrow (\mathbf{A}_i(\alpha_1;\alpha_2) \wedge \mathbf{A}_i(\alpha_1;\alpha_3))$.
(5) $\models \mathbf{A}_i\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od} \leftrightarrow (\mathbf{A}_i\texttt{confirm}\ \neg\varphi \vee \mathbf{A}_i((\texttt{confirm}\ \varphi;\alpha_1);\texttt{while}\ \varphi\ \texttt{do}\ \alpha_1\ \texttt{od}))$.

PROOF OF THEOREM 6.25: In the first case corollary 6.21 is used, the proof of the other cases is similar to the proof of theorem 5.15, and is therefore left to the reader.
$\boxtimes$

Note that the first two cases of theorem 6.25 are associated with the observations made in example 4.2. The third and the fourth case represent the intuitively desired behaviour of the sequential composition when an external nondeterministic choice is involved.

6.2. The Can-predicate and the Cannot-predicate reconsidered

The problems with the Can-predicate and the Cannot-predicate that occurred in the internal nondeterministic case do not occur with the external nondeterministic one. The semantics for the external nondeterministic choice as given in definition 6.11 is such that the Can-predicate and the Cannot-predicate express an intuitively acceptable behaviour when defined by the syntactical abbreviations that are also used in [HLM93].

6.26. DEFINITION. The Can-predicate and the Cannot-predicate are defined as follows:

- $\mathbf{Can}_i(\alpha, \varphi) \equiv \mathbf{K}_i(<\mathrm{do}_i(\alpha)>\varphi \wedge \mathbf{A}_i\alpha)$.
- $\mathbf{Cannot}_i(\alpha, \varphi) \equiv \mathbf{K}_i(\neg <\mathrm{do}_i(\alpha)>\varphi \vee \neg\mathbf{A}_i\alpha)$.

The following lemma formalizes the intuitively acceptable behaviour of the predicates.

6.27. LEMMA. *For all agents i, actions α_1, $\alpha_2 \in Ac_E$, and for all formula φ we have:*

- $\models \mathbf{Can}_i(\alpha_1 + \alpha_2, \varphi) \leftrightarrow \mathbf{Can}_i(\alpha_1, \varphi) \wedge \mathbf{Can}_i(\alpha_2, \varphi)$.
- $\models \mathbf{Cannot}_i(\alpha_1, \varphi) \vee \mathbf{Cannot}_i(\alpha_2, \varphi) \rightarrow \mathbf{Cannot}_i(\alpha_1 + \alpha_2, \varphi)$.
- $\not\models \mathbf{Cannot}_i(\alpha_1 + \alpha_2, \varphi) \rightarrow \mathbf{Cannot}_i(\alpha_1, \varphi) \vee \mathbf{Cannot}_i(\alpha_2, \varphi)$.

PROOF OF LEMMA 6.27: The proof of the first two cases is left to the reader. The third case is proved by the following model:
$\mathcal{M} =< \mathcal{S}, \pi, \mathrm{R}, \mathtt{r}, \mathtt{c} >$ where

(1) $\mathcal{S} = \{s_0, s_1, s_2\}$,
(2) $\pi(p, s_j) = \mathbf{1} \Leftrightarrow j = 2$,
(3) $\mathrm{R}(i) = (\{s_0, s_1\} \times \{s_0, s_1\}) \cup \{(s_2, s_2)\}$,
(4) $\mathtt{r}(i, a_1)(s_0) = \mathtt{r}(i, a_2)(s_1) = s_2$, $\mathtt{r}(i, a_j)(s_k) = \emptyset$ otherwise,
(5) $\mathtt{c}(i, a_1)(s_0) = \mathtt{c}(i, a_2)(s_1) = \mathbf{1}$, $\mathtt{c}(i, a_j)(s_k) = \mathbf{0}$ otherwise.

Now we have:

- $\mathcal{M}, s_0 \models \mathbf{Cannot}_i(a_1 + a_2, p)$, since $\{s \mid (s_0, s) \in \mathrm{R}(i)\} = \{s_0, s_1\}$ and $\pi(\mathbf{A}_i(a_1 + a_2), s_j) = \mathbf{0}$ for $j = 0, 1$.
- $\mathcal{M}, s_0 \not\models \mathbf{Cannot}_i(a_1, p)$ since $\pi(<\mathrm{do}_i(a_1)>p \wedge \mathbf{A}_i a_1, s_0) = \mathbf{1}$ and $\mathcal{M}, s_0 \not\models \mathbf{Cannot}_i(a_2, p)$ since $\pi(<\mathrm{do}_i(a_2)>p \wedge \mathbf{A}_i a_2, s_1) = \mathbf{1}$.

Hence the model $\mathcal{M}$ indeed shows the third case.
⊠

In lemma 6.27 it is in a nice way expressed that the external choice is completely out of the control of the agent. In particular the fact that the implication in the last case of lemma 6.27 is not valid seems reasonable: for should the agent conclude from the fact that s/he knows that $\alpha_1 + \alpha_2$ is either incorrect or unfeasible that one of α_1 and α_2 is incorrect or unfeasible, s/he seems to have some knowledge concerning the choice that the external environment makes. This would contradict our intuitive ideas as given in section 3.

Despite the intuitively nice behaviour of the Can-predicate and the Cannot-predicate when defined as in 6.26, it could be interesting to investigate other definitions of these predicates that resemble the definitions given in section 5.2. For these alternative definitions the extension to a first-order framework could be necessary.

7. Discussion

In this paper we have investigated ways to incorporate both internal and external nondeterministic actions in a system that formalizes the reasoning of agents on their abilities and the effects of the actions that they perform. To this end we defined the notions of finite computation sequences and finite computation runs: these are finite strings of atomic actions and tests. The results that we obtain seem to be intuitively correct.

Our further research regarding the topics introduced in this paper will mainly be focused on two points: firstly we would like to define a unified framework in which internal and external nondeterminism can be combined, and secondly it seems interesting to investigate a first-order variant of the framework defined in this paper, in which it is possible that the agents reason with and about the sequences of semi-atomic actions that constitute an action.

References

[AP86] K.R. Apt and G.D. Plotkin. Countable nondeterminism and random assignment. *Journal of the ACM*, 33(4):724–767, 1986.

[BBKM84] J.W. de Bakker, J.A. Bergstra, J.W. Klop, and J.-J.Ch. Meyer. Linear time and branching time semantics for recursion with merge. *Theoretical Computer Science*, 34:135–156, 1984.

[Bro86] M. Broy. A theory of nondeterminism, parallellism, communication and concurrency. *Theoretical Computer Science*, 45:1–61, 1986.

[Gam91] L.T.F. Gamut. *Logic, Language, and Meaning*, volume 1: Introduction to Logic. The University of Chicago Press, Chicago and London, 1991.

[Gol92] R. Goldblatt. *Logics of Time and Computation*, volume 7 of *CSLI Lecture Notes*. CSLI, Stanford, 1992. Second edition.

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

[Har84] D. Harel. Dynamic logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 2, pages 497–604. Reidel, Dordrecht, 1984.

[HLM93] W. van der Hoek, B. van Linder, and J.-J. Ch. Meyer. A logic of capabilities. Technical Report IR-330, Vrije Universiteit Amsterdam, 1993.

[HM85] J.Y. Halpern and Y.O. Moses. A guide to the modal logics of knowledge and belief. In *Proc. 9th IJCAI*, pages 480–490, 1985.

[Hoa85] C.A.R. Hoare. *Communicating Sequential Processes*. Prentice-Hall International, 1985.

[HR83] J. Halpern and J. Reif. The propositional dynamic logic of deterministic, well-structured programs. *Theoretical Computer Science*, 27:127–165, 1983.

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

[Mey92] J.-J. Ch. Meyer. Free choice permissions and Ross's paradox: Internal vs external nondeterminism. In P. Dekker and M. Stokhof, editors, *Proc. 8th Amsterdam Colloquium*, pages 367–380. Universiteit van Amsterdam, 1992.

[MH] J.-J. Ch. Meyer and W. van der Hoek. Epistemic logic for AI and computer science. Manuscript.

[Moo80] R.C. Moore. Reasoning about knowledge and action. Technical Report 191, SRI International, 1980.

[Moo84] R.C. Moore. A formal theory of knowledge and action. Technical Report 320, SRI International, 1984.

[Pel87] D. Peleg. Concurrent dynamic logic. *Journal of the ACM*, 34(2):450–479, 1987.

[Tho93] B. Thomas. A logic for representing actions, beliefs, capabilities, and plans. Working Notes of the AAAI Spring Symposium on 'Reasoning about Mental States: Formal Theories and Applications', 1993.

[WM91] R.J. Wieringa and J.-J.Ch. Meyer. Actor-oriented specification of dynamic and deontic integrity constraints. In B. Talheim, J. Demetrovics, and H.-D. Gerhardt, editors, *3rd Symposium on Mathematical Fundamentals of Database and Knowledge Base Systems (MFDBS 91)*, volume 495 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 1991.