Synthese (2015) 192:1117–1138
DOI 10.1007/s11229-014-0604-1

Completely and partially executable sequences of actions in deontic context

Piotr Kulicki · Robert Trypuz

Received: 30 April 2014 / Accepted: 11 November 2014 / Published online: 22 November 2014

Abstract The paper offers a logical characterisation of multi-step actions in the context of deontic notions of obligation, permission and prohibition. Deontic notions for sequentially composed actions (procedures or instructions) are founded on deontic notions for one-step actions. The present work includes a formal study of situations where execution of a multi-step action has been unsuccessful and provides normative analysis of such actions.

Keywords Deontic action logic · Sequential composition of actions · Successful and unsuccessful actions · Logic of procedures

1 Introduction

When complex planned actions are executed, it sometimes happens that they can be started but cannot be continued until their successful end. In other words, a planned action, a procedure or an instruction, may be only partially executable if its realisation meets unexpected obstacles. The subject is especially important since nowadays procedures govern more and more activities in business, administration and other fields of social life. People are often assessed on the basis of the way how they follow procedures rather than the results they achieve.

What should one do when a procedure or an instruction cannot be continued at some point of its execution? How to make sure that a given procedure does not have

P. Kulicki (✉) · R. Trypuz
Faculty of Philosophy, The John Paul II Catholic University of Lublin,
Al. Raclawickie 14, Lublin, Poland
e-mail: kulicki@kul.pl

R. Trypuz
e-mail: trypuz@kul.pl

such a dead end trap? According to our best knowledge those issues have attracted little attention in action logic and action theory in general so far. They differ from the problems of standard planning research, where one is primarily focused on organising actions to achieve "as best as possible some prestated objectives" (Ghallab et al. 2004) and, on the other hand, from analysing preconditions which make attempts successful, as in Lorini and Herzig (2008).

In the paper we set up formal tools by means of which those questions can be expressed, hoping that further research will provide the answers. We found a deontic action logic adequate for the analysis of the execution of actions according to procedures or instructions.

In the early eighties of the twentieth century, through the works of Segerberg (1982) and Meyer (1987), deontic logic was brought back to von Wright's initial idea of studying deontic notions in close relation to actions. The results of research, described in the works (Trypuz and Kulicki 2009, 2010, 2011; Trypuz 2014) contributed to a better understanding of deontic action logic systems, especially those based on finite Boolean algebra of actions. In the systems in question Boolean algebra is treated as a simple theory of actions providing a precise meaning for action constructors such as: action complement, indeterministic choice between actions and parallel execution. We want to apply the results of that research to setting up the logic of possibly unsuccessful (partially successful) complex actions.

We want to focus particularly on a sequential composition of actions. We build a framework where we can analyse a way in which sequences of actions interplay with Boolean and deontic operators. We are able to express what we find most interesting and challenging – the deontic characterisation of partially executable complex actions, for instance, what it means to impose the regulation "you ought to do α and then β" or "you are allowed to do α and then β", taking into account the fact that action α can be carried out in several ways, some of which may exclude the possibility of executing β afterwards. Thus, we have to start with modelling sequentially composed actions themselves to obtain their representation adequate for answering the questions.

We derive our theory, directly or indirectly, from the modelling of sequences of actions, widely known in computer science in the form of such formalisms as Kleene algebra, propositional dynamic logic (PDL) (in which Boolean and Kleene algebras are directly or indirectly essential components) or Hoare logic (Kleene 1956; Fischer and Ladner 1979; Hoare 1969; Hoare et al. 2011). Those systems show their usefulness in the analysis of algorithms, programs and more general phenomena of agents' behaviour. They were also connected with deontic notions in e.g. Meyer (1987), Dignum et al. (1996), Meyden (1996), Castro and Maibaum (2009) and recently Dong and Li (2103); Prisacariu and Schneider (2012). Especially the article (Prisacariu and Schneider 2012) presents an approach similar to the one adopted in this paper. Its authors, Prisacariu and Schneider, built a deontic action logic on the foundation of an algebraic structure which they call synchronous Kleene algebra. The structure combines parallel and sequential execution of actions and a free choice operator on actions. The deontic characterisation of complex actions, including multi-step actions, is defined on the basis of deontic values of simple state-to-state transitions. They introduce an algebra of actions independent of their model-theoretic structure and then define the model

for an algebraically defined normal form. Moreover, they combine their deontic action logic with PDL.

In the present paper we propose an alternative system based on results presented in Trypuz and Kulicki (2011). We focus on the algebraic description of complex actions (especially sequentially composed ones) with a special interest in action identity defined on the basis of a model-theoretic structure for actions.

In most of the recent papers concerning deontic action logic PDL operators are employed. It is essential for PDL to consider what is true after a given action has been performed. We have decided not to express our results in that framework for two reasons. The first reason is that in our considerations we disregard the results of actions. We are only interested in finding out which complex action (consisting of many consecutive steps) should start at a certain point and be continued in consecutive situations. The second reason is simplicity: to solve our scientific problem we decided to use as simple formal tools as possible. We have found Segerberg-style deontic action logic from the 1980s less complex than PDL and at the same time adequate for our purposes.

The first attempt to design the proposed system was published in Kulicki and Trypuz (2012). Now we present a slightly different version avoiding the drawbacks of the previous one. We have also significantly changed the way the system is shown.

The paper is structured as follows. In Sect. 2 we formulate basic intuitions concerning the theory of successful and unsuccessful actions, build its model and provide an action algebra, sound and complete with respect to the model. We present a new interpretation of actions which takes into account their successful and unsuccessful manifestations. The key issues of action identity are also discussed here.

In Sect. 3 we put forward a semantic characterisation of deontic operators for sequentially composed actions. We build it on the deontic characterisation of one-step actions, i.e., we assume that for each situation we know which one-step actions are permitted, forbidden and obligatory (Sect. 3.2). While defining the deontic values of actions we take into account the distinction between successful and unsuccessful realisations of sequentially composed actions—we point out that it is of particular importance in the indeterministic environment. Satisfaction conditions for the basic formulas of deontic logic of multi-step actions are defined in Sect. 3.3.

In Sect. 4 we show how deontic properties of multi-step actions emerge from the properties of one-step actions. In particular, in Sect. 4.1, we introduce an example of one-step deontic logic being sound and complete with respect to a model of the kind. Then, in two subsequent sections, we derive a model for sequentially composed actions and the respective logic.

2 Action model and its formalisation

2.1 Basic intuitions

The basic element which we use to build a formal model for a deontic action logic is a representation of two types of entities: actions and states (situations). When we write

about actions we understand them primarily as action types, i.e., kinds of action, not action tokens, i.e., individual events.

In any reasonable formalisation of action theory, at least some of the actions bring about transitions between situations. Actions change something. When actions are understood as transitions two questions arise:

(1) Is it possible that a transition between the same two states is a result of carrying out two or more different actions?
(2) Should we distinguish between an action which an agent chooses to carry out in a given situation and its possible realisations understood as the different possible ways the action might unfold in time[1] (depending on the conditions)?

The answers to the questions may be different in various contexts. They depend on the notion of action and the granularity of description. In the present paper we do not assume any form of determinism [the determinism may have the form of a functionality restriction (Castro and Maibaum 2009; Kulicki and Trypuz 2012) or the assumption of linearity of time which effects actions (Lorini and Herzig 2008)]. A similar model with no deterministic restrictions was used in Gabbay et al. (2014).

As mentioned in the introduction, in this paper we are mostly interested in studying sequentially composed actions. They are constructed from one-step actions. An important issue is that such complex actions may be unsuccessful in two ways:

(1) the first step of an action is impossible in a given situation or
(2) an action can be started but, after completing a part of its performance, it cannot be continued in the intended direction.

One step actions can be unsuccessful in the first sense, whereas the second way of being unsuccessful is specific for multi-step actions.

The conception of actions as (successful or unsuccessful) attempts was discussed by Lorini and Herzig in (2008). According to the authors an agent performs an action in an unsuccessful way in a given situation if and only if it cannot try to perform the action there or its execution precondition does not hold. Ignoring other assumptions accepted by the authors (like, for example, the determinism of actions and the fact that all actions are one step transitions) their understanding of unsuccessfulness of actions conceptually corresponds to our first meaning of unsuccessfulness above.

Finally, let us note that in order to combine sequentially composed actions using parallel composition we adopt (like in Prisacariu and Schneider 2012, Lorini and Herzig 2008) a synchronicity assumption that all basic actions take the same amount of time. Thus we can think of our model as a system similar to an indeterministic Turing machine.

2.2 Formal model

Our action frame is a triple:

$$\mathcal{AS} = \langle \mathcal{W}, \mathcal{E}, \mathcal{S}tep \rangle$$

[1] See Sect. 5.2 in Trypuz (2007).

where $\mathcal{W}$ is a nonempty, finite set of situations (Kripkean possible worlds), $\mathcal{E}$ is a nonempty, finite set of basic action types used for a cross-situation identification of actions and $\mathcal{S}tep$ is a set of action steps.

Every element of $\mathcal{S}tep$ is a triple $\langle w_1, w_2, e\rangle$, where $w_1, w_2 \in \mathcal{W}$ are initial and final states respectively and $e \in \mathcal{E}$ is a label of an action causing a transition from w_1 to w_2. Intuitively elements of $\mathcal{S}tep$ represent different ways of performing actions in certain situations. We shall call them *action steps*. Subsets of $\mathcal{S}tep$ represent *actions*, so we can model a parallel execution of two actions by the intersection of respective sets of action steps and a free choice between two actions by a sum of the respective sets.

As mentioned in the previous subsection we do not impose any restrictions on a model. Thus, it may happen that we have an indeterministic execution of an action, e.g., $\langle w_1, w_2, e\rangle \in \mathcal{S}tep$ and $\langle w_1, w_3, e\rangle \in \mathcal{S}tep$ and $w_2 \neq w_3$ and, on the other hand, that a transition between the same two states is a result of execution of two different actions, e.g., $\langle w_1, w_2, e_1\rangle \in \mathcal{S}tep$ and $\langle w_1, w_2, e_2\rangle \in \mathcal{S}tep$ and $e_1 \neq e_2$.

The set of all action steps executable in the state w will be referred to by "$exe(w)$" and defined as follows:

$$exe(w) \triangleq \{\langle w, w', e\rangle : \langle w, w', e\rangle \in \mathcal{S}tep\} \tag{1}$$

To model actions that are sequentially composed of other actions we introduce transitions which enable us to grasp the intuitive notion of a sequence of action steps.

A *transition* is a triple $\langle w_1, w_2, s\rangle$, where $w_1, w_2 \in S$ and $s \in \mathcal{S}eq$, where $\mathcal{S}eq$ is a set of finite, possibly empty, sequences of action steps from $\mathcal{S}tep$, w_1 is an initial state of the first transition, w_2 is a final state of the last transition and each element of s starts at the state in which the previous one ends. We allow s to be empty and in that case $w_1 = w_2$. The sequence s, the third element of our triple, if nonempty, contains complete information about the transition, since w_1 occurs in its first element and w_2—in its last element. Thus, the two remaining elements: w_1 and w_2 are redundant in that case. We keep them explicitly shown for the sake of readability. It is worth mentioning that Segerberg in (2009) builds his action theory for the purpose of deontic consideration in a similar way. The basic elements of his action theory are paths (which correspond to our sequences) being sequences of points. Each finite path has its first and last element. Two paths can be combined into one if the last element of the first path is the same as the last element of the second path. In Segerberg's framework it is possible to extract all the steps a path is made of and two paths with the same first and last point are not necessarily the same.

Henceforward we shall use the symbol $\mathcal{T}rans$ for the set of all transitions constructed from action steps from $\mathcal{S}tep$. More formally we can define the notion of transition (i.e. a sequence of action steps) in $\mathcal{AS}$ inductively in the following way:

- For any $w \in \mathcal{W}$ a triple $\langle w, w, []\rangle$[2] is a transition.
- If $\langle w_1, w_2, s\rangle$ $(s \in \mathcal{S}eq)$ is a transition and $\langle w_2, w_3, e\rangle$ is an action step from $\mathcal{S}tep$ then $\langle w_1, w_3, s'\rangle$, where s' is a result of adding the action step $\langle w_2, w_3, e\rangle$ at the end of the sequence s, is also a transition.

[2] The symbol [] denotes an empty sequence of action types.

Using the concept of transition which we have just introduced, we can represent an arbitrary action by a set of transitions. In particular, one-step actions are represented by sets of one-step transitions, i.e., transitions with a sequence of length 1.

In a natural way we can define a sequential composition of sets of transitions. Let T_1 and T_2 be subsets of $\mathcal{T}rans$. Then[3]

$$T_1 \circ T_2 \triangleq \{\langle w_1, w_2, s\rangle \mid \exists\, w', s_1, s_2\ (s = s_1 \oplus s_2\ \&\ \langle w_1, w', s_1\rangle \in T_1\ \&\langle w', w_2, s_2\rangle \in T_2)\} \tag{2}$$

Similarly to our definition of the function exe, now we define a function "exe^*" as a set of transitions executable in a given situation, formally:

$$exe^*(w) \triangleq \{\langle w, w', s\rangle : \langle w, w', s\rangle \in \mathcal{T}rans\} \tag{3}$$

2.3 Identity of actions in the model

The key issue that enables us to operate on actions is the identity relation. To define it let us first introduce the appropriate formal language[4]:

$$\alpha \ ::= \ \mathbf{0} \mid \mathbf{skip} \mid a_i \mid \alpha \sqcup \alpha \mid \alpha \sqcap \alpha \mid \alpha; \alpha \tag{4}$$

where a_i belongs to a finite set of *basic actions* (or in other words *action generators*) Act_0, $\mathbf{0}$ is the impossible action, **skip** is a special action of doing nothing analogous to the *skip* program used in the theory of programming, $\alpha \sqcup \beta$—α or β is a free choice between α and β; $\alpha \sqcap \beta$—α and β is a parallel execution of α and β. Act is a set of all actions which can be expressed in the language ($Act_0 \subseteq Act$).

One-step actions (basic actions and their combinations achieved by the use of operators $\sqcap$ and $\sqcup$ or, in other words, actions which do not contain sequential composition ";" and **skip**) are interpreted as non-empty sets of one-step transitions. By synchronicity assumption all one-step actions take the same amount of time. The special actions $\mathbf{0}$ and **skip** do not take any time—$\mathbf{0}$ is impossible, so it cannot be executed, **skip** can be understood as doing nothing in no time.

When we come to the multiple-step actions, we consider transitions that are fully realised from their beginning to the end and transitions that represent only the initial, proper part of the action and cannot lead to its end (i.e., they finish in a state in which the remaining part of the action cannot be carried out). We shall refer to the former as *successful transitions*, and to the latter as *unsuccessful transitions* (see Fig. 1 and its caption).

[3] The symbol $\oplus$ stands for a concatenation of sequences from $\mathcal{S}eq$.

[4] We do not introduce action complement (negation) to our language. The operator is not essential for our task and in the context of sequential composition causes the well known intuitive and technical problems (see e.g. Broersen 2004).

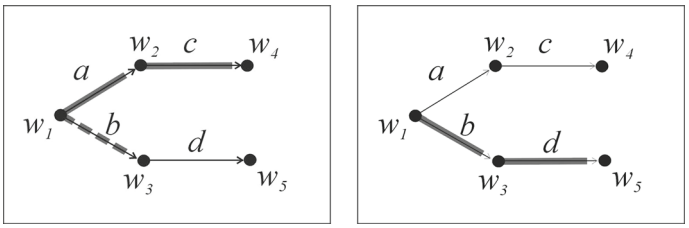

Fig. 1 In the figure on the *left*, we can see the interpretation of action "$(a; c) \sqcup (b; c)$" (for better readability, we use language *letters* as labels for transitions). *Bold* and *non-dashed lines* (transitions) represent their successful parts whereas the bold and dashed ones—unsuccessful parts. An agent has a choice to carry out $a; c$ (a and then c) or $b; c$ (b and then c). Only the first choice guarantees a successful execution of the whole action. If the agent decided in w_1 to do b, it would be trapped—there is no way to carry out c after b. In the figure on the *right*, we can see the interpretation of action "$b; d$" in the same model. There is only one way to fulfil the action and it is a successful path

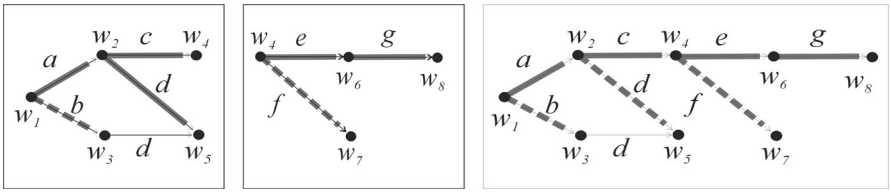

Fig. 2 In the first two figures we can see actions $\alpha = (a; (c \sqcup d)) \sqcup (b; c)$ and $\beta = (e; g) \sqcup (f; e)$, respectively. To execute α an agent has a choice either to do a and then c or d, or to carry out b and then c. We can see that the first option is completely successful, whereas the second one is only unsuccessful. Similarly we should go through the interpretation of action β to understand its nature. In the third figure we have the interpretation of action $\alpha; \beta$ in the same model. After sequentially combining the two actions, the sequence leading from w_1 through w_2 to w_5 becomes unsuccessful because there is no way to perform β in the state w_5

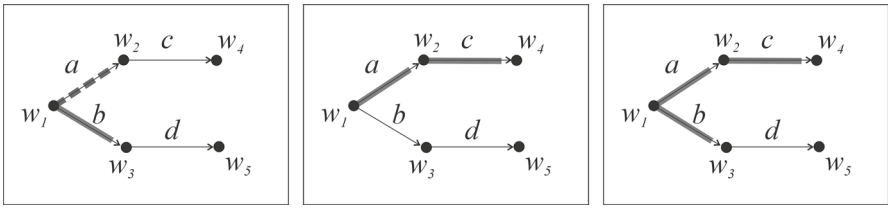

Fig. 3 In all the three figures we have a model with four transitions: a, b, c, d. In the first two figures one can see a visualisation of two complex actions: $\alpha = (a; d) \sqcup b$ and $\beta = a; c$ respectively. In the third figure on the right the free choice between actions α and β, i.e. $\alpha \sqcup \beta$, is presented. Let us note that the action $a; d$ in the first figure cannot be succesfully performed—only its initial part, a, is executable. That fact is represented by the bold and dashed line over the transition a. However, the line does not occur in the model of action $\alpha \sqcup \beta$. That is because a is the initial part of a succesful sequence realising the action β

Now we can ask, whether it is possible to predict the successful and unsuccessful transitions of complex actions on the basis of successful and unsuccessful transitions they are made up of? The answer if positive. Before we formally set it out, let us illustrate how it works on examples—see Figs. 2, 3 and 4 illustrating sequential composition, free choice and parallel composition of actions respectively.

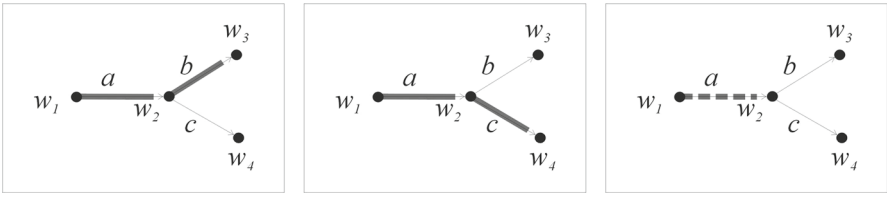

Fig. 4 In the first two figures we can see actions $\alpha = a; b$ and $\beta = a; c$. In the third figure on the right we can see the action $\alpha \sqcap \beta$. The intersection of paths representing α and β is empty, so is the successful part of $\alpha \sqcap \beta$. The common initial part of α and β—a—becomes an unsuccessful part of $\alpha \sqcap \beta$. It is worth stressing that the action a is *per se* still executable here, but from the point of view of the "context", i.e., what was really planned, carrying its out does not fulfill intended direction, i.e., the action $\alpha \sqcap \beta$

2.4 Interpretation of successful and unsuccessful parts of actions

Formally we represent successful and unsuccessful transitions as two separate sets. The sets are determined by two functions: "*suc*" and "*unsuc*". The function "*suc*" is defined below.

Interpretation function *suc* for successful transitions

$suc : Act \longrightarrow 2^{Seq}$ is defined as follows:

$$\langle w_1, w_2, [\langle w_1, w_2, e\rangle]\rangle \in suc(a_i) \Longrightarrow \forall\, w_1', w_2'\ \langle w_1', w_2', [\langle w_1', w_2', e\rangle]\rangle \in suc(a_i) \tag{5}$$

$$suc(\mathbf{skip}) = \{\langle w, w, []\rangle : w \in \mathcal{W}\} \tag{6}$$

$$suc(\mathbf{0}) = \emptyset \tag{7}$$

$$suc(\alpha \sqcup \beta) = suc(\alpha) \cup suc(\beta) \tag{8}$$

$$suc(\alpha \sqcap \beta) = suc(\alpha) \cap suc(\beta) \tag{9}$$

$$suc(\alpha; \beta) = suc(\alpha) \circ suc(\beta) \tag{10}$$

The interpretation for basic actions a_i are sets of one-step transitions—see (5). The intended interpretation for successful basic one-step action is such that (i) it is always successful, when it is in the model and (ii) one-step transitions with the same label belong to the interpretation of the same action generator. Condition (6) states that **skip** can be carried out in any situation. It is worth stressing that its execution does not take any time. We do it without passing through any sequences. The impossible action **0** can never be successfully executed.

For unsuccessful transitions the situation is more complicated since they behave less nicely in the context of the operations on actions. Thus, to define *unsuc* (for unsuccessful transitions) we need to introduce three auxiliary functions on sets of transitions.

The first function is "*beg*". For a set of transitions T, the function provides a set of their initial fragments (i.e., *beginnings*):

$$beg(T) = \{\langle w_1, w_2, s_1\rangle \in \mathcal{T}rans : \exists\, w_3, s_2, s\ (s_1 \oplus s_2 = s\ \&\ \langle w_1, w_3, s\rangle \in T\ \&\ \langle w_2, w_3, s_2\rangle \in \mathcal{T}rans)\} \tag{11}$$

Let us notice that the initial fragments do not need to be proper, so we have $T \subseteq beg(T)$.

The next function is "*max*". From a set of transitions T it selects only *maximal* transitions, i.e., no proper initial fragment of a transition from T can be an element of the resulting set:

$$max(T) = \{u \in T : \forall\, u' \in T\ (u \in beg(\{u'\}) \Longrightarrow u' = u)\} \tag{12}$$

The last of the three functions is "fin". Its arguments are two sets of transitions: T_1 and T_2. The function results in a set of transitions from T_1 which have no continuation in T_2:

$$fin(T_1, T_2) = \{\langle w_1, w_2, s\rangle : \langle w_1, w_2, s\rangle \in T_1\ \&\ \neg\exists\, w_3, s'\langle w_2, w_3, s'\rangle \in T_2\} \tag{13}$$

Below, we define the interpretation function *unsuc*, using the functions *beg*, *max* and fin.

Interpretation function *unsuc* **for unsuccessful transitions**
Assuming that

$$all(\alpha) = suc(\alpha) \cup unsuc(\alpha)$$

$unsuc : Act \longrightarrow 2^{Seq}$ is defined as follows:

$$unsuc(a_i) = unsuc(\mathbf{skip}) = unsuc(\mathbf{0}) = \emptyset \tag{14}$$

$$unsuc(\alpha \sqcup \beta) = max\big((unsuc(\alpha) - beg(suc(\beta))) \cup (unsuc(\beta) - beg(suc(\alpha)))\big) \tag{15}$$

$$unsuc(\alpha \sqcap \beta) = max\big(beg(all(\alpha)) \cap beg(all(\beta))) - beg(suc(\alpha) \cap suc(\beta))\big) \tag{16}$$

$$unsuc(\alpha; \beta) = unsuc(\alpha) \cup fin(suc(\alpha), all(\beta)) \cup (suc(\alpha) \circ unsuc(\beta)) \tag{17}$$

Condition (14) states that action generators (being one-step actions) and **skip** do not have unsuccessful executions. Moreover, the impossible action **0** cannot be executed in any way, so it has no unsuccessful transitions either. Condition (15) states that unsuccessful transitions of a free choice of actions α and β consists of unsuccessful

transitions of α and β that are not the beginnings of successful transitions of β and α respectively. Function max eliminates non-maximal elements from the resulting set to maintain uniqueness. Condition (16) constructs $unsuc(\alpha \sqcap \beta)$ as a set of maximal common beginnings of successful and unsuccessful transitions connected with α and β that are not successful for $\alpha \sqcap \beta$. Condition (17) states that the set of unsuccessful transitions of $\alpha; \beta$ consists of unsuccessful transitions of α, elements of $suc(\alpha)$ that do not have a continuation in β and unsuccessful transitions of β attached as a continuation to α.

Let us notice that an element of $unsuc(\alpha)$ cannot be an initial fragment of any element of $suc(\alpha)$, formally:

$$unsuc(\alpha) \cap beg(suc(\alpha)) = \emptyset \tag{18}$$

Finally, the interpretation of actions is a function into a pair of sets of transitions. It is defined as follows:

$$\mathcal{I}(\alpha) = \langle suc(\alpha), unsuc(\alpha) \rangle \tag{19}$$

Provided

$$suc(\alpha, w) = suc(\alpha) \cap exe^*(w) \tag{20}$$

$$unsuc(\alpha, w) = unsuc(\alpha) \cap exe^*(w) \tag{21}$$

by $\mathcal{I}^w(\alpha)$ we understand a local interpretation of an action, i.e., interpretation relativised to some situation w:

$$\mathcal{I}^w(\alpha) = \langle suc(\alpha, w), unsuc(\alpha, w) \rangle \tag{22}$$

2.5 Axiomatisation of action algebra

In this section we list some important tautologies of the system constituting its axiomatisation. The set of tautologies contains: monoid axioms for "$\sqcup$" and "**0**" (associativity and left-right identity):

$$(\alpha \sqcup \beta) \sqcup \gamma = \alpha \sqcup (\beta \sqcup \gamma) \tag{23}$$

$$\alpha \sqcup \mathbf{0} = \mathbf{0} \sqcup \alpha = \alpha \tag{24}$$

monoid axioms for ";" and "**skip**" (associativity and left-right identity):

$$\alpha; (\beta; \gamma) = (\alpha; \beta); \gamma \tag{25}$$

$$\alpha; \mathbf{skip} = \mathbf{skip}; \alpha = \alpha \tag{26}$$

commutativity axiom for "$\sqcup$":

$$\alpha \sqcup \beta = \beta \sqcup \alpha \tag{27}$$

";" distributes over "⊔" on both the left and right:

$$\alpha; (\beta \sqcup \gamma) = (\alpha; \beta) \sqcup (\alpha; \gamma) \qquad (\alpha \sqcup \beta); \gamma = (\alpha; \gamma) \sqcup (\beta; \gamma) \tag{28}$$

0 is the left annihilator for ";":

$$\mathbf{0}; \quad \alpha = \mathbf{0} \tag{29}$$

idempotency axiom for "⊔":

$$\alpha \sqcup \alpha = \alpha \tag{30}$$

It is worth mentioning that the structure $(Act, \sqcup, ;, \mathbf{0}, \mathbf{skip})$ does not form idempotent semiring, because **0** is not the right annihilator for ";", i.e., the formula:

$$\alpha; \quad \mathbf{0} = \mathbf{0}$$

(which is e.g. an axiom of Kleene algebra) is not a tautology. Additionally we have idempotency, commutativity and associativity axioms for "⊓", distributivity "⊓" over "⊔" and absorption:

$$\alpha \sqcap \alpha = \alpha \tag{31}$$
$$\alpha \sqcap \beta = \beta \sqcap \alpha \tag{32}$$
$$(\alpha \sqcap \beta) \sqcap \gamma = \beta \sqcap (\alpha \sqcap \gamma) \tag{33}$$
$$\alpha \sqcap (\beta \sqcup \gamma) = (\alpha \sqcap \beta) \sqcup (\alpha \sqcap \gamma) \tag{34}$$
$$(\alpha \sqcap \beta) \sqcup \beta = \beta \,\&\, \alpha \sqcap (\alpha \sqcup \beta) = \alpha \tag{35}$$

Finally, two laws limited to the restricted kinds of complex actions are tautologies: weak distributivity and intersection with **skip**. Weak distributivity of "⊓" over ";" takes the following form:

$$\alpha, \beta \, do \, not \, contain \text{ ";" } and \text{ "}\mathbf{skip}\text{"} \Longrightarrow (\alpha; \gamma) \sqcap (\beta; \delta) = (\alpha \sqcap \beta); (\gamma \sqcap \delta) \tag{36}$$

Formula (36) corresponds to an axiom of synchronous Kleene algebra from Prisacariu and Schneider (2012). Intersection with **skip** is governed by the following law:

$$\alpha \, does \, not \, contain \text{ "}\mathbf{skip}\text{"} \Longrightarrow (\alpha \sqcap \mathbf{skip}) = \mathbf{0} \tag{37}$$

The law does not hold for an arbitrary action α. Let us consider α which is equal to $\beta \sqcup \mathbf{skip}$, when β does not contain **skip**. Then

$$(\alpha \sqcap \mathbf{skip}) = (\beta \sqcup \mathbf{skip}) \sqcap \mathbf{skip} = \mathbf{0} \sqcup \mathbf{skip} = \mathbf{skip} \tag{38}$$

We can generalise the above equations to the following form:

$$(\alpha \sqcap \mathbf{skip}) = \mathbf{0} \, or \, (\alpha \sqcap \mathbf{skip}) = \mathbf{skip} \tag{39}$$

Let us now focus on actions of the form $\alpha; \mathbf{0}$. Intuitively we understand them as the ones that cannot be continued from some point. We have already noticed that in our system such actions do not have to equal $\mathbf{0}$ (i.e., they are not always completely impossible). We can use that fact to observe that any unsuccessful action in our algebra fulfils the following equation:

$$\alpha = \alpha; \quad \mathbf{0} \tag{40}$$

Any action different from $\mathbf{0}$, satisfying condition (40) is a *dead end trap*. Formally we define the operator **trap**:

$$\mathbf{trap}(\alpha) \triangleq (\alpha \neq \mathbf{0}) \wedge (\alpha = \alpha; \mathbf{0}) \tag{41}$$

2.6 Soundness and completeness of the algebra with respect to the intended model

To verify that (23)–(37) are indeed tautologies, one must check that both sides of equations have the same values of functions *suc* and *unsuc*. For *suc* it is quite an easy task. For *unsuc* it is not that straightforward, but the proofs can be obtained by routine algebraic transformations. Equations (23)–(37) allow us to transform each action into an equivalent action in the normal form defined below.

Normal form of an action
Any action of a from:

$$a_1 \sqcap \ldots \sqcap a_n,$$

where $a_1, \ldots a_n \in Act_0 (n \geq 1)$ is a quasiatom. Action **skip** and any action of the form:

$$\alpha_1; \ldots; \alpha_n,$$

where $\alpha_1, \ldots \alpha_{n-1}$ $(n \geq 1)$ are quasi-atoms and α_n is a quasiatom or $\mathbf{0}$ is a sequent. An action is in the normal form iff it is a free choice of sequents, i.e. has the form:

$$\alpha_1 \sqcup \ldots \sqcup \alpha_n,$$

where $\alpha_1, \ldots \alpha_n$ $(n \geq 1)$ are sequents.

Theorem 1 *For any action α there exists an action β in the normal form such that $\alpha = \beta$.*

Proof To prove the theorem it is enough to use distributivity laws (28), (34) and (36), absorption law (35), and (37). □

Theorem 2 *Axiomatisation is complete*

Proof The theorem is a consequence of the previous one. Let α and β be actions which are not equal in the algebra. If so, the difference between their normal forms α_1 and β_1 is other than the order of disjuncts and conjuncts or their repetitions. Let us now consider a model in which all basic actions occurring in α_1 and β_1 are executable. In such a model the interpretations of α_1 and β_1 differ, so α and β are not equal in the model. □

3 Deontic operators for sequentially composed actions

3.1 Intuitive introduction

We assume that deontic characterisation of sequentially composed actions is determined by deontic characterisation of one-step actions which constitute them. That excludes direct regulation of deontic values of multi-step actions. It is a simplification (one may even say a limitation), but it has the advantage of making our model more transparent. Still we can create or verify complex procedures by analysing what should be done in each state of their realisation. Then an agent can execute them being sure that it will comply with local regulations.

We assume that for each situation we know which one-step actions are permitted, forbidden and obligatory. On that basis we reconstruct deontic characterisation of sequentially composed actions. The difficulty lies in the fact that, after the execution of the first step of a sequentially composed action an agent is in a different situation, with different local *deontology* (i.e., description of the situation including norms). We must also remember that we may find nondeterministic choice at each step. To set out a deontic value of an action we have to look at its every possible execution.

Accepting such an approach, what remains to be done is to define how deontic operators interact with a sequential composition. A one-step deontic action logic which is the base for the realisation of purpose can be chosen from many existing ones, see e.g., Trypuz and Kulicki (2013) to review some of them. This makes our solution flexible enough to be adopted to different normative scenarios.

To define the deontic value of an action we need to take into account successful and unsuccessful executions of sequentially composed actions. It is especially important if we assume that an agent does not have to be aware of all the obstacles it can face and all the regulations which apply to its plan during its realisation.

Thus, taking into account what has been said above we say that a multi-step action is permitted in our theory if each action step making it up is permitted in a situation where it is planned to be triggered. Also unsuccessful attempts to perform a permitted action have to be permitted—an agent should not find itself in a situation when choosing a way to perform a permitted action ends up in a situation where following the plan further is not permitted, even if that attempt is at the end unsuccessful as a dead end trap. A multi-step action is prohibited if each possible successful or unsuccessful realisation of that action contains at least one prohibited action step.

As far as obligatory multi-step actions are concerned they are treated as good procedures and instructions. As such they should always lead to their final result. There should be no way to fail during their realisation; thus there should be no dead

end traps in obligatory actions. Moreover, we believe that any initial fragment of an obligatory action should be obligatory. Thus, an action is obligatory only if every way of its realisation is required at each step.

Finally it seems reasonable to explain why the unsuccessful realisations of actions are taken into account when considering deontics. In our paper we tackle a situation in which one starts with deontic modelling of one-step action and then, on that basis, tries to do the same with complex actions, especially sequences. Unsuccessfulness of complex actions naturally emerges when one can freely combine basic actions to create a plan. That type of unsuccessfulness is not a result of "unexpected causes" but rather the nature of actions which are combined; some sequences are unsuccessful by nature, e.g., none can go to Warsaw and see La Gioconda (since the painting is not exhibited there).

3.2 Formalisation of deontology

Let us now put forward a formal model for the above intuitions. Let us notice that some intuitions will be expressed on the level of satisfaction conditions in the next section.

The semantic definition of our logic is based on the following deontic extension of the action frame $\mathcal{AS}$ used in the previous section:

$$\mathcal{DAS} = \langle \mathcal{W}, \mathcal{E}, \mathcal{S}tep, \mathcal{LEG}, \mathcal{ILL}, \mathcal{REQ} \rangle,$$

where $\mathcal{W}$, $\mathcal{E}$ and $\mathcal{S}tep$ are characterised as in $\mathcal{AS}$ and $\mathcal{LEG}$, $\mathcal{ILL}$, $\mathcal{REQ}$ are deontic functions from $\mathcal{W}$ to $2^{2^{\mathcal{S}tep}}$, representing (sets of sets of) action steps: legal, illegal and required in a specific state respectively. At this point we do not specify their properties and mutual relations—we shall do that in Sect. 4.1. Here we would like to focus the reader's attention only on the way in which the counterparts of deontic functions for sequences emerge from $\mathcal{LEG}$, $\mathcal{ILL}$, $\mathcal{REQ}$.

Thus, on the basis of the functions $\mathcal{LEG}$, $\mathcal{ILL}$ and $\mathcal{REQ}$ we define similar functions describing legal, illegal and required sets of transitions. Let us use symbols $\mathcal{LEG}^*$, $\mathcal{ILL}^*$ and $\mathcal{REQ}^*$ respectively for the new functions. They transform the states from $\mathcal{W}$ into $2^{2^{\mathcal{T}rans}}$. We intend $\mathcal{LEG}^*(s)$ and $\mathcal{REQ}^*(s)$ to be defined in such a way that each step of each transition is legal (required) in an appropriate state and that $\mathcal{ILL}^*(s)$ must have some illegal steps in each transition.

For formal definitions of the sets we need the following auxiliary functions: ini, fso and rem. Let $T \subseteq \mathcal{T}rans$ be a set of transitions and $w \in \mathcal{W}$ a state.

Having in mind Fig. 1 one may think that $T = \{\langle w_1, w_4, s_1\rangle, \langle w_1, w_5, s_2\rangle\}$, where $s_1 = [\langle w_1, w_2, a\rangle, \langle w_2, w_4, c\rangle]$ and $s_2 = [\langle w_1, w_3, b\rangle, \langle w_3, w_5, d\rangle]$. We shall refer to that example below in order to explain the newly introduced functions.

Function ini for a given set of transitions T and a given situation w_1 provides a set of *initial* steps:

$$ini(T, w_1) = \{t \in \mathcal{S}tep : \exists\, w' \in \mathcal{W}, s \in Seq \text{ such that } \langle w_1, w', [t] \oplus s\rangle \in T\} \tag{42}$$

In our example $ini(T, w_1) = \{[\langle w_1, w_2, a\rangle], [\langle w_1, w_3, b\rangle]\}$.

Function fso (*first step outcomes*) for a given set of transitions T and a given situation w_1 returns a set of states that are reachable from w_1 by the first step of any transition from T:

$$fso\,(T, w_1) = \{w' \in \mathcal{W} : \exists\, t \in ini(T, w_1), e \in \mathcal{E} \text{ such that } t = \langle w_1, w', e\rangle \in \mathcal{S}tep\} \tag{43}$$

In our example $fso\,(T, w_1) = \{w_2, w_3\}$.

Finally, *rem*, *remainders* of a set of transitions T with respect to the initial state w_1 and the first step outcome w_2, returns a set of transitions starting from w_2 obtained from transitions from T by removing their first steps:

$$rem(T, w_1, w_2) = \{\langle w_2, w_3, s\rangle \in \mathcal{T}rans : w_2 \in fso\,(T, w_1)\,\&\, \exists\, t \in \mathit{Step} \text{ such that } \langle w_1, w_3, [t|s]\rangle \in T\} \tag{44}$$

In our example, $rem(T, w_1, w_2) = \{\langle w_2, w_4, [\langle w_2, w_4, c\rangle]\rangle\}$ and $rem(T, w_1, w_3) = \{\langle w_3, w_5, [\langle w_3, w_5, d\rangle]\rangle\}$.

Below we formally define $\mathcal{LEG}^*$, $\mathcal{ILL}^*$ and $\mathcal{REQ}^*$ with the use of ini, fso and rem.

Definitions of $\mathcal{LEG}^*$, $\mathcal{ILL}^*$ and $\mathcal{REQ}^*$ for transitions
For any $T \subseteq Trans$:

$T \in \mathcal{ILL}^*(w)$ iff $\forall\, t = \langle w_1, w_2, s\rangle \in T$, there exists an action step u of the sequence s, and a set X, such that $u \in X \in \mathcal{ILL}(w)$.

$T \in \mathcal{LEG}^*(w)$ iff $T = \emptyset$ or the following two conditions hold:
(i) $ini(T, w) \in \mathcal{LEG}(w)$,
(ii)$\forall\, w' \in fso(T, w), rem(T, w, w') \in \mathcal{LEG}^*(w')$.

$T \in \mathcal{REQ}^*(w)$ iff $T = \emptyset$ or the following two conditions hold:
(i) $ini(T, w) \in \mathcal{REQ}(w)$ or $ini(T, w) = \emptyset$,
(ii)$\forall\, w' \in fso(T, w), rem(T, w, w') \in \mathcal{REQ}^*(w')$.

It is worth mentioning that for the purposes of the recursive definition of the $\mathcal{REQ}^*$, $\emptyset$ is an element of $\mathcal{REQ}^*(w)$ for any w. That allows us also to treat the initial fragments of obligatory actions as obligatory. At the same time we do not want to treat the impossible action **0** itself as obligatory. We shall prevent that by placing an appropriate requirement in satisfaction conditions for the operator "O" (see below).

3.3 Satisfaction conditions

The basic language of $\mathcal{DAL}$ is defined in the following way:

$$\varphi \;::=\; \alpha = \alpha \mid \mathrm{P}(\alpha) \mid \mathrm{F}(\alpha) \mid \mathrm{O}(\alpha) \mid \neg\varphi \mid \varphi \wedge \varphi, \tag{45}$$

where α stands for the names of actions defined by (4), "$\mathtt{P}(\alpha)$" – α is (strongly) permitted; "$\mathtt{F}(\alpha)$" – α is forbidden, "$\mathtt{O}(\alpha)$" – α is obligatory.

Now we are ready to formulate satisfaction conditions for a formula of $\mathcal{DAL}$ in a state s of a model $\mathcal{M}$:

Satisfaction conditions for multi-step actions
Providing $\mathcal{I}(\alpha) = \langle suc(\alpha), unsuc(\alpha)\rangle$ and
$\mathcal{I}^w(\alpha) = \langle suc(\alpha, w), unsuc(\alpha, w)\rangle$:

$$
\begin{array}{lcl}
\mathcal{M}, w \models \mathtt{F}(\alpha) & \Longleftrightarrow & suc(\alpha, w) \cup unsuc(\alpha, w) \in \mathcal{ILL}^*(w) \\
\mathcal{M}, w \models \mathtt{P}(\alpha) & \Longleftrightarrow & suc(\alpha, w) \cup unsuc(\alpha, w) \in \mathcal{LEG}^*(w) \\
\mathcal{M}, w \models \mathtt{O}(\alpha) & \Longleftrightarrow & suc(\alpha, w) \in \mathcal{REQ}^*(w)\ \&\ \\
 & & suc(\alpha, w) \neq \emptyset\ \&\ unsuc(\alpha, w) = \emptyset \\
\mathcal{M}, w \models \alpha = \beta & \Longleftrightarrow & \mathcal{I}(\alpha) = \mathcal{I}(\beta) \\
\mathcal{M}, w \models \neg\varphi & \Longleftrightarrow & \mathcal{M}, w \not\models \varphi \\
\mathcal{M}, w \models \varphi \wedge \psi & \Longleftrightarrow & \mathcal{M}, w \models \varphi\ \&\ \mathcal{M}, w \models \psi
\end{array}
$$

For a permitted (forbidden) action we require that both successful and unsuccessful transitions are legal (illegal). For obligatory actions we require that such an action cannot be unsuccessfully started, so the set of unsuccessful transitions has to be empty. Obligatory actions have to be possible, so the set of successful transitions has to be non-empty. Of course transition sets for all states have to be required.

The concepts of satisfiability, validity and tautology are defined in the standard way.

Finally, let us notice that for nonempty one-step actions satisfaction conditions are much simpler and can be expressed as below.

Satisfaction conditions for nonempty one-step sequence
Providing $\mathcal{I}^w(\alpha) = \langle suc(\alpha, w), \emptyset\rangle$,
where $suc(\alpha, w) = \{\langle w, w', [\langle w, w', e\rangle]\rangle, \ldots\}$,

$$
\begin{array}{lcl}
\mathcal{M}, w \models \mathtt{F}(\alpha) & \Longleftrightarrow & suc(\alpha, w) \in \mathcal{ILL}^*(w) \\
 & \Longleftrightarrow & \{\langle w, w', e\rangle, \ldots\} \in \mathcal{ILL}(w) \\
\mathcal{M}, w \models \mathtt{P}(\alpha) & \Longleftrightarrow & suc(\alpha, w) \in \mathcal{LEG}^*(w) \\
 & \Longleftrightarrow & \{\langle w, w', e\rangle, \ldots\} \in \mathcal{LEG}(w) \\
\mathcal{M}, w \models \mathtt{O}(\alpha) & \Longleftrightarrow & suc(\alpha, w) \in \mathcal{REQ}^*(w) \\
 & \Longleftrightarrow & \{\langle w, w', e\rangle, \ldots\} \in \mathcal{REQ}(w)
\end{array}
$$

4 From one-step to multi-step deontic logic

As we have stated at the end of the previous section, satisfaction conditions for one-step actions can be expressed by using only sets $\mathcal{LEG}(w)$, $\mathcal{ILL}(w)$ and $\mathcal{REQ}(w)$

(without "$*$"). It means that a logic for sequential actions emerges from a logic for one-step actions. In the next section we use a fragment of a minimal one-step deontic action logic from Trypuz and Kulicki (2013) without action negation as an example. Then in Sects. 4.2 and 4.3 we extend it to a multi-step case.

4.1 A model and logic for one-step actions

In this system we connect deontic notions with actions both on the level of syntax and in formal semantics. That is why $\mathcal{LEG}(w)$, $\mathcal{ILL}(w)$ and $\mathcal{RE}\mathcal{Q}(w)$ have been defined as functions from $\mathcal{W}$ to $2^{2^{Step}}$. Since in the present paper we use a nondeterministic model in which an action can have different outcomes even if it is executed twice in the same state, we have to modify slightly their definitions.

Now the functions $\mathcal{LEG}(w)$, $\mathcal{ILL}(w)$ and $\mathcal{RE}\mathcal{Q}(w)$ have values in 2^{Step}. We also assume that whenever $t_1 = \langle w, w_1, e\rangle$ belongs to some X in $\mathcal{LEG}(w)$, $\mathcal{ILL}(w)$ or $\mathcal{RE}\mathcal{Q}(w)$, then for any w_2, $t_2 = \langle w, w_2, e\rangle$ also belongs to X. The condition has to do with the fact that an action undertaken by an agent may have different outcomes independently of the agent's will and behaviour and it states that the action (not the outcomes) are the subject of deontic qualification.

Other properties of $\mathcal{LEG}(s)$, $\mathcal{ILL}(s)$ and $\mathcal{RE}\mathcal{Q}(s)$ are characterised as follows. For any $w \in \mathcal{W}$ and $X, Y \in 2^{Step}$, $\mathcal{LEG}(w)$ and $\mathcal{ILL}(w)$ satisfy the following principles:

$$X \in \mathcal{LEG}(w) \,\&\, Y \subseteq X \Longrightarrow Y \in \mathcal{LEG}(w) \tag{46}$$

$$X \in \mathcal{LEG}(w) \,\&\, Y \in \mathcal{LEG}(s) \Longrightarrow X \cup Y \in \mathcal{LEG}(w) \tag{47}$$

$$X \in \mathcal{ILL}(w) \,\&\, Y \subseteq X \Longrightarrow Y \in \mathcal{ILL}(w) \tag{48}$$

$$X \in \mathcal{ILL}(w) \,\&\, Y \in \mathcal{ILL}(w) \Longrightarrow X \cup Y \in \mathcal{ILL}(w) \tag{49}$$

For any w, $\mathcal{LEG}(w)$ and $\mathcal{ILL}(w)$ have only the empty set in common:

$$\mathcal{LEG}(w) \cap \mathcal{ILL}(w) = \{\emptyset\} \tag{50}$$

A necessary condition for two elementary transitions to be required is that their intersection should be required too[5]:

$$X \in \mathcal{RE}\mathcal{Q}(w) \,\&\, Y \in \mathcal{RE}\mathcal{Q}(w) \Longrightarrow X \cap Y \in \mathcal{RE}\mathcal{Q}(w) \tag{51}$$

An impossible action is never required:

$$\emptyset \notin \mathcal{RE}\mathcal{Q}(w) \tag{52}$$

There is no state $w \in \mathcal{W}$ in which all transitions are illegal:

$$exe(w) \notin \mathcal{ILL}(w) \tag{53}$$

[5] Henceforward universal quantifications over whole formulas are left implicit.

If X is an action required in w and its intersection with action Y is empty, then Y is illegal in s:

$$X \in \mathcal{RELQ}(w) \,\&\, X \cap Y = \emptyset \Longrightarrow Y \in \mathcal{ILL}(w) \tag{54}$$

The following tautologies of the system, along with the algebra of one-step actions, form its axiomatisation:

$$\mathtt{P}(\alpha \sqcup \beta) \equiv \mathtt{P}(\alpha) \wedge \mathtt{P}(\beta) \tag{55}$$
$$\mathtt{F}(\alpha \sqcup \beta) \equiv \mathtt{F}(\alpha) \wedge \mathtt{F}(\beta) \tag{56}$$
$$\mathtt{F}(\alpha) \wedge \mathtt{P}(\alpha) \equiv (\alpha = \mathbf{0}) \tag{57}$$
$$\mathtt{O}(\alpha) \wedge \mathtt{O}(\beta) \rightarrow \mathtt{O}(\alpha \sqcap \beta) \tag{58}$$
$$\neg(\mathtt{O}(\alpha) \wedge \mathtt{F}(\alpha)) \tag{59}$$
$$\mathtt{O}(\alpha) \wedge (\alpha \sqcap \beta = \mathbf{0}) \rightarrow \mathtt{F}(\beta) \tag{60}$$

Formula (60) is a complement-free way to express the economy law from Trypuz and Kulicki (2013):

$$\mathtt{O}(\alpha) \rightarrow \mathtt{F}(\overline{\alpha})$$

From (60) and (59) we can derive

$$\neg\mathtt{O}(\mathbf{0}) \tag{61}$$

and

$$\mathtt{O}(\alpha) \wedge \mathtt{O}(\beta) \rightarrow (\alpha \sqcap \beta \neq \mathbf{0}) \tag{62}$$

4.2 Multi-step model consequences

The above definitions determine the values of functions $\mathcal{LEG}^*$, $\mathcal{ILL}^*$ and $\mathcal{RELQ}^*$ (for any state as their arguments) to inherit most properties imposed on the values of functions $\mathcal{LEG}$, $\mathcal{ILL}$ and $\mathcal{RELQ}$. Thus, for any state $w \in \mathcal{W}$ we have:

$$T_1 \in \mathcal{LEG}^*(w) \& T_2 \subseteq T_1 \Longrightarrow T_2 \in \mathcal{LEG}^*(w) \tag{63}$$
$$T_1 \in \mathcal{LEG}^*(w) \& T_2 \in \mathcal{LEG}^*(w) \Longrightarrow (T_1 \cup T_2) \in \mathcal{LEG}^*(w) \tag{64}$$
$$T_1 \in \mathcal{ILL}^*(w) \& T_2 \subseteq T_1 \Longrightarrow T_2 \in \mathcal{ILL}^*(w) \tag{65}$$
$$T_1 \in \mathcal{ILL}^*(w) \& T_2 \in \mathcal{ILL}^*(w) \Longrightarrow (T_1 \cup T_2) \in \mathcal{ILL}^*(w) \tag{66}$$
$$T_1 \in \mathcal{RELQ}^*(w) \& T_2 \in \mathcal{RELQ}^*(w) \Longrightarrow (T_1 \cap T_2) \in \mathcal{RELQ}^*(w) \tag{67}$$
$$exe^*(w) \notin \mathcal{ILL}^*(w) \tag{68}$$

Let us sketch the proofs of the above properties. (63) holds by (46) since $ini(T_2) \subseteq ini(T_1)$, $fso(T_2, w) \subseteq fso(T_1, w)$ and, for any $w' \in fso(T_2, w)$, $rem(T_2, w, w') \subseteq rem(T_1, w, w')$. (64) holds by (47) since $ini(T_1 \cup T_2) = ini(T_1) \cup ini(T_2)$, $fso(T_1 \cup T_2, w) = fso(T_1, w) \cup fso(T_2, w)$ and, for any $s' \in fso(T_1 \cup T_2, w)$, $rem(T_1 \cup T_2, w, w') = rem(T_1, w, w') \cup rem(T_1, w, w')$. (65) and (66) are obvious. For (67), if $T_1 \cap T_2 = \emptyset$ then the property holds by the definition of $\mathcal{RELQ}^*$. Otherwise we

have to check that $ini(T_1 \cap T2) \in \mathcal{RE}\mathcal{Q}$ and that for each $w' \in fso(T_1 \cap T2, w)$ $rem(T_1 \cap T_2, w', w'') \in \mathcal{RE}\mathcal{Q}^*$. The former is guarantied by (51). For the latter the remainders have to be analysed step by step and in each stage (51) should be applied. For (68) it is enough to notice that a set of one-step transitions that are based on action labels that are not illegal in w (the set is not empty by (53)) is not a member of $\mathcal{ILL}^*(w)$. Thus, by (65), it is not a member of $exe(w)$ either.

Moreover, we have an additional property concerning $\emptyset$ and transition with an empty sequence of steps $\langle w, w, [] \rangle$ for any w:

$$\emptyset \in \mathcal{RE}\mathcal{Q}^*(w) \tag{69}$$

$$\langle w, w, [] \rangle \in \mathcal{RE}\mathcal{Q}^*(w) \tag{70}$$

(69) is guaranteed by the definition of $\mathcal{RE}\mathcal{Q}^*$. For (70) we need to notice that $init\langle w, w, [] \rangle = \emptyset$ and there is no $w' \in fso(\langle w, w, [] \rangle)$. Thus, conditions (i) and (ii) from the definition of $\mathcal{RE}\mathcal{Q}^*$ are fulfilled.

On the other hand, the following counterpart of property (54) does not hold:

$$X \in \mathcal{RE}\mathcal{Q}^*(w) \quad and \quad X \cap Y = \emptyset \Longrightarrow Y \in \mathcal{ILL}^*(w) \tag{71}$$

$X \cap Y = \emptyset$ when X and Y have elements of different lengths.

4.3 Multi-step logic

Let us start with the laws of multi-step logic with deontic operators without sequential composition and **skip**.

It is interesting to what extent the multi-step logic inherits the laws of the one-step logic. Formulas (55), (56), (57) and (59) are tautologies in the multi-step system.

To see that formula (55) is valid, it is enough to recall that $suc(\alpha \sqcup \beta) = suc(\alpha) \cup suc(\beta)$ and notice that $unsuc(\alpha \sqcup \beta) \subseteq beg(all(\alpha) \cup all(\beta))$ and $unsuc(\alpha) \cup unsuc(\beta) \subseteq beg(all(\alpha \sqcup \beta))$. Thus, by the satisfaction condition for $\mathtt{P}$ (55) is valid. The validity of (56) follows immediately from the satisfaction conditions for $\mathtt{F}$, and properties (66). The validity of (57) follows immediately from the conditions for $\mathtt{P}$ and $\mathtt{F}$. To see that (59) is valid let us suppose α is obligatory in a state w. Then, for w, $suc(\alpha) \neq \emptyset$ and $suc(\alpha, w) \in \mathcal{RE}\mathcal{Q}^*(w) \subseteq \mathcal{LEG}^*(w)$. Thus, for any $u \in suc(\alpha)$ each action label must be an element of respective $\mathcal{RE}\mathcal{Q}(w') \subseteq \mathcal{LEG}(w')$. But $\mathcal{LEG}(w')$ and $\mathcal{ILL}(w')$ are disjoint for any state. Thus, u cannot contain an illegal fragment and consequently $\neg\mathtt{F}(\alpha)$.

On the other hand (58), (60) and (62) are not tautologies. However, the following weaker versions of (58) and (60) are:

$$\alpha \neq \mathbf{skip} \wedge \beta \neq \mathbf{skip} \rightarrow (\mathtt{O}(\alpha) \wedge \mathtt{O}(\beta) \wedge \neg\mathbf{trap}(\alpha \sqcap \beta) \rightarrow \mathtt{O}(\alpha \sqcap \beta)) \tag{72}$$

$$\alpha \neq \mathbf{skip} \wedge \beta \neq \mathbf{skip} \rightarrow (\mathtt{O}(\alpha) \wedge (\alpha \sqcap \beta = \mathbf{0}) \rightarrow \mathtt{F}(\beta)) \tag{73}$$

Basic laws with sequential composition or **skip** are as follows:

$$\mathtt{O}(\alpha;\beta) \rightarrow \mathtt{O}(\alpha) \tag{74}$$
$$\mathtt{F}(\alpha) \rightarrow \mathtt{F}(\alpha;\beta) \tag{75}$$
$$\mathtt{P}(\alpha;\beta) \rightarrow \mathtt{P}(\alpha) \tag{76}$$
$$\mathtt{P}(\alpha) \rightarrow \mathtt{P}(\alpha;\mathbf{0}) \tag{77}$$
$$\neg\mathtt{O}(\alpha;\mathbf{0}) \tag{78}$$
$$\mathtt{O}(\mathbf{skip}) \tag{79}$$

The validity of (74)–(79) follows directly from the satisfaction conditions for P, F and O. We shall not explain all of them. Let us just focus on two which at first glance may seem paradoxical.

Formula (77) is a derivative of formula $\mathtt{P}(\mathbf{0})$ for one step actions (it follows directly from (57)), which we have accepted after Segerberg (1982). It states that an impossible action is permitted as it cannot have any bad effects. Any attempt to perform action $a;\mathbf{0}$ can contain only fragments of a. Thus, if a is permitted, then $a;\mathbf{0}$ should also be permitted. The difference is that a is completely realised and action $a;\mathbf{0}$ is not, since after a an agent should perform an impossible action (which of course is not possible) but it should not affect permissibility.

Formula (79) states that **skip** understood as "no action in no time" is obligatory. Why should it be so? The technical reason is that **skip** can be added at the beginning and at the end of any action α without changing it. If α is obligatory, to keep the consistency of the satisfaction conditions, **skip** should also be obligatory. Intuitively it is not that obvious. To defend (79) we can only say that **skip** is inevitable – in no time it is impossible to do anything else then **skip**. Thus, (79) corresponds to the law of standard deontic logic stating that tautology is obligatory.

The following formula reflects the requirement of no traps[6] (dead ends) for obligatory actions:

$$\mathbf{trap}(\alpha) \rightarrow \neg\mathtt{O}(\alpha \sqcup \beta) \tag{80}$$

A procedure must not give an agent the possibility of making a wrong choice. (80) is essential for our notion of obligatory complex actions in which we connect them with procedures.

(80) was assumed in the satisfaction condition for obligation. In fact the requirement on the model level is even stronger—there can be no dead end transitions within the interpretation of obligatory action. However, in the language of logic we cannot express it fully since we do not have the access to single transitions.

We conjecture that formulas (23)–(80) with the rules of Modus Ponens and Substitution constitute a complete axiomatisation of the system, but we have not formulated the proof yet.

[6] **trap** is defined by (41).

Among their consequences are:

$$\mathtt{P}(\alpha) \rightarrow \mathtt{P}(\mathbf{skip}) \tag{81}$$

$$\mathtt{P}(\alpha) \rightarrow \mathtt{P}(\alpha \sqcap \beta) \tag{82}$$

$$\mathtt{F}(\alpha) \rightarrow \mathtt{F}(\alpha \sqcap \beta) \tag{83}$$

$$\mathtt{P}(\mathbf{0}) \quad \mathtt{F}(\mathbf{0}) \quad \neg\mathtt{O}(\mathbf{0}) \tag{84}$$

5 Conclusion and future perspectives

We have presented a deontic action framework taking into account unsuccessful sequences of actions. It has been founded on deontic action logic and its model for one-step actions. Within that setting it is possible to treat actions as instructions or procedures that an agent can realise step-by-step being sure that each step of a permitted action is permitted and each way of starting an obligatory action allows continuing such an action to its end. In our framework we have introduced the interpretation function for actions that takes into account their successful and unsuccessful executions.

The system can be extended in a straightforward way by imposing additional conditions on the model (e.g. determinism of action outcomes) or by enriching the language with modalities and PDL operators.

Another interesting extension of the system could be obtained by adding action negation (complement). Such an extension is, however, more challenging because of the well known technical problems occurring in systems in which sequential composition coexists with action negation.

Acknowledgments We would like to thank Prof. J.-J.Ch. Meyer and the two anonymous reviewers for valuable remarks on the earlier versions of this work. This research was supported by the National Science Centre of Poland (DEC-2011/01/D/HS1/04445).

References

Broersen, J. (2004). Action negation and alternative reductions for dynamic deontic logics. *Journal of Applied Logic*, *2*(1), 153–168.

Castro, P. F., & Maibaum, T. (2009). Deontic action logic, atomic boolean algebra and fault-tolerance. *Journal of Applied Logic*, *7*(4), 441–466.

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

Dong, H., & Li, X. (2013). A deontic action logic for complex actions. In: D. Grossi, O. Roy, H. Huang (Eds.), *Logic, rationality, and interaction*. Lecture Notes in Computer Science (Vol. 8196, pp. 311–315). Springer, Berlin. doi:10.1007/978-3-642-40948-6_24.

Fischer, M. J., & Ladner, R. E. (1979). Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, *18*(2), 194–211.

Gabbay, D., Gammaitoni, L., & Sun, X. (2014). The paradoxes of permission an action based solution. Journal of Applied Logic (0). doi:10.1016/j.jal.2014.01.003.

Ghallab, M., Nau, D., & Traverso, P. (2004). *Automated planning theory and practice*. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Hoare, C. A. R. (1969). An axiomatic basis for computer programming. *Communications of the ACM*, *12*(10), 576–580.

Hoare, T., Möller, B., Struth, G., & Wehrman, I. (2011) Concurrent kleene algebra and its foundations. *The Journal of Logic and Algebraic Programming*, *80*(6), 266–296. http://dblp.uni-trier.de/db/journals/jlp/jlp80.html#HoareMSW11.

Kleene, S. C. (1956). Representation of events in nerve nets and finite automata. In C. E. Shannon & J. McCarthy (Eds.), *Automata studies* (pp. 3–41). Princeton: Princeton University Press.

Kulicki, P., & Trypuz, R. (2012). A deontic action logic with sequential composition of actions. In: T. Ågotnes, J. Broersen, D. Elgesem (Eds.), *Deontic logic in computer science*. Lecture Notes in Computer Science (Vol. 7393/2012, pp. 184–198). Boston: Springer.

Lorini, E., & Herzig, A. (2008). A logic of intention and attempt. *Synthese*, *163*(1), 45–77.

Meyer, J. (1987). A different approach to deontic logic: Deontic logic viewed as variant of dynamic logic. *Notre Dame Journal of Formal Logic*, *29*(1), 109–136.

Prisacariu, C., & Schneider, G. (2012). A dynamic deontic logic for complex contracts. *The Journal of Logic and Algebraic Programming*, *81*(4), 458–490.

Segerberg, K. (1982). A deontic logic of action. *Studia Logica*, *41*, 269–282.

Segerberg, K. (2009). Blueprint for a dynamic deontic logic. *Journal of Applied Logic*, *7*(4), 388–402.

Trypuz, R. (2007). Formal ontology of action: A unifying approach. Ph.D. thesis, Università degli Studi di Trento, ICT International Doctorate School.

Trypuz, R. (Ed.). (2014). *Krister Segerberg on logic of actions, outstanding contributions to logic* (Vol. 1). Berlin: Springer.

Trypuz, R., & Kulicki, P. (2009). A systematics of deontic action logics based on boolean algebra. *Logic and Logical Philosophy*, *18*, 263–279.

Trypuz, R., & Kulicki, P. (2010). Towards metalogical systematisation of deontic action logics based on boolean algebra. In *Proceedings of the 10th international conference deontic logic in computer science*. Lecture Notes in Computer Science (Vol. 6181, pp. 132–147). New York: Springer.

Trypuz, R., & Kulicki, P. (2011). A norm-giver meets deontic action logic. *Logic and Logical Philosophy*, *20*, 59–72.

Trypuz, R., & Kulicki, P. (2013). On deontic action logics based on Boolean algebra. *Journal of Logic and Computation*. doi:10.1093/logcom/ext057.

van der Meyden, R. (1996). The dynamic logic of permission. *Journal of Logic and Computation*, *6*(3), 465–479.