ARTICLE IN PRESS

Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Norm-based deontic logic for access control, some computational results

Xin Sun [a,b,*], Xishun Zhao [a], Livio Robaldo [c]

[a] Institute of Logic and Cognition, Sun Yat-sen University, Guangzhou, China

[b] Department of Foundations of Computer Science, Faculty of Philosophy, The John Paul II Catholic University of Lublin, Lublin, Poland

[c] University of Luxembourg, Luxembourg

ARTICLE INFO

Article history:
Received 11 September 2016
Received in revised form
20 December 2016
Accepted 22 January 2017
Available online xxxx

Keywords:
Deontic logic
Access control
Computational complexity

ABSTRACT

In this paper we study the complexity of deontic logics grounded on *norm-based* semantics and apply norm-based deontic logic to access control. Four principal norm-based deontic logics have been proposed so far: imperative logic, input/output logic, deontic default logic and deontic defeasible logic. We present the readers that imperative logic is complete for the 2ed level of the polynomial hierarchy and deontic default logic is located in the 3ed level of the polynomial hierarchy. We then show how it is possible to impose restrictions to imperative logic such that the complexity goes down to be tractable, allowing the logic to be used in practical applications. We focus on a specific application: access control.

1. Introduction

Deontic logic is a formal study of normative reasoning and norms. In 1951, the philosopher and logician Georg von Wright wrote a paper called "Deontic Logic" [1], which subsequently became the name of the research area. Von Wright's deontic logic is exactly the same as the modal logic KD. Such logic is later called standard deontic logic (SDL). With the work of Meyer [2], deontic logic became a part of computer science. SDL has been a useful tool in the specification and reasoning of access control policies because key notions in access control such as permission, prohibition and obligation are exactly the subjects of SDL [3–5].

Deontic logic provides a mathematically rigorous language for modeling access control policies. The vagueness and ambiguity of informal language disappear in the formal language of deontic logic. Deontic logic is also associated with a sound and complete axiomatic characterization. The interpretation of the normative concepts is axiomatically constructed in deontic logic. As a consequence of completeness, the framework is guaranteed to be consistent. Without consistency, the move to the implementation level would be meaningless.

Different approaches of deontic logic, alternative to SDL, have been studied in the past 6 decades including imperative logic [6,7], dynamic deontic logic [2,8], deontic STIT logic [9,10], input/output logic [11], deontic default logic [12,13] and deontic defeasible logic [14,15]. Those results are summarized in the handbook of deontic logic [16,17]. In imperative logic, input/output logic, deontic default logic and deontic defeasible logic, norms are explicitly represented. The truth value of deontic propositions in those logics are explained not by some set of possible worlds, but with references to a set of given norms. Such a non-possible world semantics has been originally termed in Hansen [18] as 'norm-based semantics'. We then use norm-based deontic logic as a general term to refer input/output logic, imperative logic, deontic default logic and deontic defeasible logic and use deontic modal logic to refer those approaches which adopt possible world semantics such as SDL.

Norms are the first class citizens in norm-based deontic logic. Norms are everywhere in our daily life and also in access control. For example:

- You **should** drive on the right side.
- Alice is **permitted** to read file-1 on Mondays.
- Bob is **forbidden** to write on file-2.
- Carol is **obliged** to delete all related files when he finishes his task.

In general, we view norms as normative rules which are used to regulated agent's behavior. A norm is a rule in the sense that it contains both a **premise** and a **consequence**. The premise describes the situation in which it is triggered, while the consequence prescribes the demand of the norm. Norms are normative in the sense

* Corresponding author at: Institute of Logic and Cognition, Sun Yat-sen University, Guangzhou, China.
E-mail address: xin.sun.logic@gmail.com (X. Sun).

http://dx.doi.org/10.1016/j.future.2017.01.028
0167-739X/

that they classify what is obligatory, permitted or forbidden. An access control policy is a set of norms defining which user is to be granted access to which resource under which circumstances. Compared to SDL, norm-based deontic logic has the following advantages.

1. Norm-based deontic logic solves the contrary-to-duty paradox. The contrary-to-duty paradox is the most notorious paradox in deontic logic. The original phrasing of the paradox requires a formalization of the following scenario in which the sentences are mutually consistent and logically independent [19].
   (a) It ought to be that John goes to help his neighbors.
   (b) It ought to be that if John goes to help his neighbors, then he tells them he is coming.
   (c) If John does not go to help his neighbors, then he ought not to tell them he is coming.
   (d) John does not go to help.

   But the formalization of the above scenario using SDL is either inconsistent or not logically independent. Being not able to solve the contrary-to-duty paradox is seen as one of the most serious limitations of SDL. The contrary-to-duty scenario is also found in access control and it is called the "violation of obligation" in Benferhat et al. [20] and corresponds to the policy of reaction to new intrusions in Cuppens [5].

   Norm-based deontic logic, on the other hand, gives consistent and logically independent formalization of the above scenario, therefore solves the contrary-to-duty paradox. In general, norm-based deontic logic provides correct prescriptions in situations where some norms are already violated [21].
2. Norm-based deontic logic offers a formal mechanism to deal with normative conflicts.

   Consider the following scenario taken from Hansen [7], which is sometimes called the 'order puzzle': before you go to a party, you become the recipient of various imperative sentences:
   (a) Your mother says: if you drink anything, then do not drive.
   (b) Your best friend says: if you go to the party, then you drive.
   (c) Some acquaintance says: if you go to the party, then have a drink with me.

   Assume mother is more important than best friend, who is more important than acquaintance. What will you do? Intuitively, you should obey your mother and your best friend, and hence do the driving and not accept your acquaintance's invitation. However, it is not so clear what formal mechanism could explain this reasoning. Handling normative conflicts is also an important issue in access control and is discussed in Benferhat et al. [20]. SDL is unable to handle such conflicting imperatives. On the other hand, norm-based deontic logic appears as suitable tools to formalize such reasoning.
3. Norm-based deontic logic characterizes various notions of permission.

   Permission is probably the most important notion in the specification of an access control policy [22,23]. Philosophically, it is common to distinguish between two kinds of permission: negative permission and positive permission. Negative permission is straightforward to describe: something is negatively permitted according to certain norms iff it is not prohibited by those norms. That is, iff there is no obligation to the contrary. Positive permission is more elusive. Intuitively, something is positively permitted according to certain norms iff it can be derived from those norms. But what exactly does "derive" mean? In mathematics we can derive theorems in a "straight" way or by contradiction. These two methods of derivation give two different notions of positive permission. Makinson and van der Torre [24] introduces these two types of positive permission as static and dynamic permission. Other notions of permission, such as permission as *exception*, have been studied in [25,26]. All these notions of permission are useful in access control and can be captured by norm-based deontic logics, while SDL is only able to capture negative permission.

The above advantages of norm-based deontic logic shows that comparing to SDL, norm-based deontic logic is a better tool to be applied in the specification and reasoning of access control policies. Among those existing norm-based deontic logics, imperative logic is the most suitable for access control because different notions of permission can be uniformly expressed in imperative logic.

For the existing norm-based deontic logics, the computational complexity of input/output logic and deontic defeasible logic is studied in [27,26]. In this paper, we study the complexity of imperative logic and deontic default logic (Sections 3 and 5.1). We show that both imperative logic and deontic default logic are decidable but computationally intractable. We then impose restrictions to obtain some tractable imperative logic such that we can practically apply them to access control (Section 4). For the sake of readability, we put all proofs in the Appendix.

2. Background: complexity theory

We assume the readers are familiar with notions like Turing machines and the complexity classes P, NP and coNP. Oracle Turing machines and some complexity classes related to oracle Turing machines will be used in this paper.

Definition 1 (*Oracle Turing Machine [28]*)**.** An oracle for a language L is a device that is capable of reporting whether any string w is a member of L. An oracle Turing machine M^L is a modified Turing machine that has the additional capability of querying an oracle. Whenever M^L writes a string on a special oracle tape it is informed whether that string is a member of L, in a single computation step.

P^{NP} is the class of problems solvable by a deterministic polynomial time Turing machine with an NP oracle. $\mathrm{NP}^{\mathrm{NP}}$ is the class of problems solvable by a non-deterministic polynomial time Turing machine with an NP oracle. Σ_2^p is another name for $\mathrm{NP}^{\mathrm{NP}}$. Π_2^p is another name for $\mathrm{coNP}^{\mathrm{NP}}$. Δ_{i+1}^p is $\mathrm{P}^{\Sigma_i^p}$ and Σ_{i+1}^p is $\mathrm{NP}^{\Sigma_i^p}$.

3. Imperative logic

If some given norms come into conflict, the best an agent can be expected to do is to follow a maximal subset of those norms. Intuitively, a priority ordering over the norms can be helpful in resolving conflicts, but a formal resolution mechanism has been difficult to provide. In particular, reasoning about prioritized norms is overshadowed by problems such as the order puzzle that are not satisfactorily resolved by many existing approaches such as Brewka [29], Marek and Truszczynski [30]. Based on input/output logic [11], Hansen [7] develops prioritized imperative logic which overcomes those difficulties.

3.1. Input/output logic

Input/output logic takes its origin in the study of conditional norms. The basic idea is: norms are conceived as a deductive machine, like a black box which produces normative statements as output, when we feed it factual statements as input. In input/output logic, a norm is an ordered pair of formulas $(a, x) \in L_{\mathbb{P}} \times L_{\mathbb{P}}$, where $L_{\mathbb{P}}$ is the language of propositional logic build from the set of propositional atoms $\mathbb{P}$. There are two types of norms which are used in input/output logic, mandatory norms and permissive norms. A mandatory norm $(a, x) \in O$ is read as "given a, x is obligatory". A permissive norm $(a, x) \in P$ is read as "given a, x is permitted". Mandatory norms are called commands or imperatives in imperative logic, while permissive norms are called licenses or authorizations. To distinguish these two types of norms in notation, we may represent commands as $a \Rightarrow_o x$ and licenses as $a \Rightarrow_p x$. In this paper, we will however stick the notation

Please cite this article in press as: X. Sun, et al., Norm-based deontic logic for access control, some computational results, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.028