Logical Modalities within the European AI Act: An Analysis

Lara Lawniczak
lara.lawniczak@uni-bamberg.de
University of Bamberg
Bamberg, Bavaria, Germany

Christoph Benzmüller
christoph.benzmueller@uni-bamberg.de
University of Bamberg
Bamberg, Bavaria, Germany
Freie Universität Berlin, Berlin, Germany

Abstract

The paper presents a comprehensive analysis of the European AI Act in terms of its logical modalities, with the aim of preparing its formal representation, for example, within the logic-pluralistic Knowledge Engineering Framework and Methodology (LogiKEy). LogiKEy develops computational tools for normative reasoning based on formal methods, employing Higher-Order Logic (HOL) as a unifying meta-logic to integrate diverse logics through shallow semantic embeddings. This integration is facilitated by Isabelle/HOL, a proof assistant tool equipped with several automated theorem provers. The modalities within the AI Act and the logics suitable for their representation are discussed. For a selection of these logics, embeddings in HOL are created, which are then used to encode sample paragraphs. Initial experiments evaluate the suitability of these embeddings for automated reasoning, and highlight key challenges on the way to more robust reasoning capabilities.

CCS Concepts

• **Computing methodologies** → **Knowledge representation and reasoning**; • **Applied computing** → **Law**; • **Theory of computation** → *Higher order logic*; *Automated reasoning*.

Keywords

European AI Act, Modalities, Non-classical logics, Higher-order Logic, Proof Assistants, Automated Reasoning

ACM Reference Format:

Lara Lawniczak and Christoph Benzmüller. 2025. Logical Modalities within the European AI Act: An Analysis. In *Proceedings of International Conference on Artificial Intelligence and Law (ICAIL)*. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

In the last decade, the representation of, and the reasoning with, legal information has gained growing attention in the AI and Law community. Alongside established approaches like Akoma Ntoso [28] and LegalRuleML [3, 29], research is currently being invested in the exploration and application of the logic-pluralistic knowledge engineering framework and methodology LogiKEy [8], which develops computational tools for normative reasoning based on formal methods.

An interesting and relevant use case for the LogiKEy framework is the European AI Act [16], which came into force in August 2024. The AI Act classifies AI systems into three risk levels, and assigns different rules and regulations to each category. If successful, the logical formalisation of key concepts of the AI Act within the LogiKEy framework could, for example, prepare the way for sophisticated automated or interactive compliance checks and support in various ways the technical enforcement of the new regulation. It is not only the AI Act itself that is of interest, but also the specific follow-up regulations and standardisations that will be triggered by the AI Act, which will then impose specific regulatory constraints on concrete applications of AI systems.

Before adequate logical representations can be developed, it is essential to identify the modalities that occur in the AI Act and the logics, or combinations of logics, suited to adequately represent these. This paper addresses this challenge as its main contribution. In the long run, however, it is rather the concrete *instantions* of the abstract legislation of the AI Act, i.e. the more concrete laws that are expected to follow from the adoption of the AI Act in the following years, that are relevant and interesting for formalisation. We expect that the findings on the identified modalities will (to a large extent) overlap and apply to them as well.

The paper is organized as follows. Section 2 lists and discusses the modalities found in the relevant parts of the AI Act. The findings of this section are relevant beyond the LogiKEy context. Section 3 explores potential logic systems for representing these modalities. Section 4 briefly presents preliminary results on embedding selected logics in the LogiKEy framework and using them to formalize some exemplary sections of the AI Act. Section 5 concludes the paper.

2 Modalities in the AI Act

This section presents the modalities (and other challenges) identified in the AI Act and illustrates them with examples. In order to find all the modalities in the document, it was read several times, with different foci of interest, and visualized with the help of mind maps and tables.

2.1 Obligations, Prohibitions, and Permissions

The AI Act is a European piece of legislation intended to regulate the use of AI systems. In it *Obligations* are expressed with the word "shall", [1] as in this sentence from Article 8:

> "High-risk AI systems shall comply with the requirements established in this section." [16, §8(1)]

[1] Within legal acts such as the AI Act, the world "shall" can also have performative or constitutive functions instead of deontic ones. However, the primary function of "shall" in the AI Act, is the introduction of deontic obligations.

ICAIL, June 16–20, 2025, Chicago, IL

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

The negated shall, "shall not", is employed to express *Prohibitions*. An example is:

> "Where an importer has sufficient reason to consider that a high-risk AI system is not in conformity with this Regulation, or is falsified, or accompanied by falsified documentation, it shall not place the system on the market until it has been brought into conformity." [16, §23(2)]

For the expression of *Permissions*, two strategies can be identified in the AI Act: They are introduced by either "may" or "is/are empowered to".

2.2 Contrary-to-Duty-Obligations

A special type of obligations that occurs within the AI Act deserves attention: *contrary-to-duty obligations* (CTDs) are obligations that arise only if a primary obligation is not fulfilled, meaning that it is "conditional on [...] violating [a] primary obligation" [26].

An example of a CTD in the AI Act can be found in Article 20:

> "Providers of high-risk AI systems which consider or have reason to consider that a high-risk AI system that they have placed on the market or put into service is not in conformity with this Regulation shall immediately take the necessary corrective actions to bring that system into conformity, to withdraw it, to disable it, or to recall it, as appropriate." [16, §20(1)]

It has been stated before that high-risk systems must comply with the requirements [16, §16(1)]. This is the primary obligation in the given context. However, obligations are not always fulfilled; they may also be violated. In the event of such a violation, the obligation to take corrective action becomes relevant. This is a CTD, applicable only when the primary obligation has been violated. CTD situations are often represented in a typical structure following Chisholm [14]. The discussed example then looks as follows:

(1) It ought to be that high-risk AI systems comply with the requirements in the regulation.
(2) It ought to be that if a high-risk AI system does not comply with the requirements, providers take corrective actions.
(3) If a system complies with the requirements, the provider must not take any corrective actions. [2]
(4) Concrete Situation: The system does not comply with the requirements.

Notice that some of these obligations in this example are agentive for the provider. The aspects of agency and agentive obligations are discussed separately in Section 3. First, two special kinds of CTDs are considered.

2.2.1 Contrary-to-Duty-Obligations Involving Multiple Agents. Some CTDs within the AI Act relate to multiple agents, which complicates matters, as illustrated by the following example from §36:

> "If the notifying authority comes to the conclusion that the notified body no longer meets the requirements laid down in Article 31 or that it is failing to fulfil its obligations, it shall restrict, suspend or withdraw the designation as appropriate, depending on the seriousness of the failure to meet those requirements or fulfil those obligations." [16, §36(4)]

For simplicity, the notion of an agent belief, expressed in the words *comes to the conclusion that*, is ignored (since beliefs are the subject of Section 4). Also, disregard the temporal notion contained in *no longer*, since CTDs involving temporality are discussed in Section 2.2.2.

The focus here lies on the involvement of two distinct agents: The notifying authority and the notified body. The situation is as follows: The notified body has several primary obligations that are specified within the AI Act [16, §31]. If it violates these obligations, another obligation arises. However, this obligation is agentive for the notifying authority, not for the notified body that violated the primary obligation.

2.2.2 Contrary-to-Duty-Obligations Involving Temporality. There are cases in the AI Act where CTDs occur in combination with temporal constraints, stating that a certain property used to be fulfilled but *no longer* is. An example is:

> "Where a notifying authority has sufficient reason to consider that a notified body no longer meets the requirements laid down in Article 31, or that it is failing to fulfil its obligations, the notifying authority shall without delay investigate the matter with the utmost diligence." [16, §36(4)]

To understand the CTD, §30 must be taken into account. There it is stated that the notifying authority "may notify only conformity assessment bodies which have satisfied the requirements laid down in Article 31" [16, §30(1)]. Unfortunately, this example does not involve only temporality but also multiple agents as discussed in the Section 2.2.1. To focus on only temporality, we reformulate the regulation as follows:

> Only conformity assessment bodies that fulfill the requirements in Article 31 before the notification may be notified (adapted from Article 30). If a conformity assessment body that was notified (= notified body) no longer meets the requirements laid down in Article 31, the matter shall be investigated further with the utmost diligence (adapted from Article 36).

Now it becomes visible how this example is different from a typical CTD: Technically speaking, the notified body is only obligated to fulfill the requirements in Article 31 at one point in time *before* being notified. Hence, the fact that a notified body *no longer* fulfills the requirements in Article 31 does not equal a violation of the previous obligation. The obligation to further investigate the matter then is just a normal obligation, not a CTD, and could be expressed as such. In this case, the logic representing this situation must not only provide adequate obligation operators, but must also be able to express temporality in order to properly capture the notion of *before* (as discussed in Section 2.5).

Another possibility is disregarding the temporality here and translating the example to match the usual CTD structure:

(1) It ought to be that notified bodies fulfill the requirements in Article 31.

[2] It has been brought to our attention that an interpretation as "need not" might be more appropriate. We are thankful for this remark and aim to uncover the differences between these interpretations in the future.

(2) It ought to be that if a notified body does not comply with the requirements in Article 31, further investigations are started.
(3) If a notified body does comply with the requirements in Article 31, no further investigation must be started.
(4) Concrete Situation: A notified body does not fulfill the requirements in Article 31.

Reformulated as above, the example can be treated like a typical CTD.

2.3 Agency and Agentive Obligations

Since different agents interact with an AI system during its lifetime, it is plausible that these agents have different duties towards the system based on their relation to it. Such obligations then are not general but agentive: They only hold for a specific type of agent, e.g. for providers or importers. For instance, the sentence

> "Providers of high-risk AI systems shall (...) have a quality management system in place which complies with Article 17" [16, §16(c)]

is such an agentive obligation because it expresses what the provider ought to do. Agentive obligations are different from general obligations, and must be distinguishable.

2.4 Beliefs of Agents

Apart from agentive obligations, another modality relating to agency is crucial in the AI Act: Agent beliefs. For example, Article 20 states:

> "Providers of high-risk AI systems which consider or have reason to consider that a high-risk AI system that they have placed on the market or put into service is not in conformity with this Regulation shall immediately take the necessary corrective actions to bring that system into conformity, to withdraw it, to disable it, or to recall it, as appropriate." [16, §20(1)]

Here, the provider's belief that a high-risk AI system is not in conformity with the regulation is relevant, not whether the system is, in fact, not conforming. Beliefs can be either right or wrong, and it is essential to differentiate them from facts.

2.5 Temporality and Temporal Notions

Time is an important concept within the AI Act, with many articles specifying obligations that must be fulfilled before certain events occur. For instance, Article 23 states:

> "Before placing a high-risk AI system on the market, importers shall ensure that the system is in conformity with this Regulation by verifying that (...) "[16, §23(1)]

This obligation for importers is tied to a specific period and point in time; it must be fulfilled *before* the system is placed on the market.

Additionally, some sentences contain hidden temporal notions, such as this phrase in Article 9: "When implementing the risk management system as provided for in paragraphs 1 to 7, providers shall…" [16, §9(9)]. The sentence implies simultaneity which becomes obvious if it is rephrased as follows: "While implementing the risk management system, providers are obligated to (...)."

2.6 Exceptions

Within the AI Act, the concept of exceptions from general rules emerges as a recurring theme. Consider this sentence from Article 5:

> "The following AI practices shall be prohibited: (...) the use of 'real-time' remote biometric identification systems in publicly accessible spaces for the purposes of law enforcement, unless and in so far as such use is strictly necessary for one of the following objectives:..." [16, §5(1)]

This sentence outlines a general rule – the prohibition of certain AI practices – followed by exceptions based on specific objectives.

2.7 Fuzziness

Fuzzy statements can be found in many places within the AI Act. They can be identified by their vagueness and usually involve statements that are not true or false, but true or false to a certain degree. The following expressions are examples of such fuzziness:

> "unless and in as far as such use is strictly necessary for one of the following objectives" [16, §5(1)], "to the extent to which..." [16, §7], and "at a minimum" [16, §11(1)].

All these statements express vague notions or involve degrees of truth or necessity.

2.8 Combinations of Modalities

The modalities of the AI Act do not appear in isolation. Some of the examples discussed in Section 2.2 have already shown that two or more modalities are often combined within a single sentence. While this was ignored in the previous discussion for the sake of simplicity, it cannot be ignored when constructing a logical system intended to cover the entire AI Act, or larger coherent parts of it. Such a system must not only accurately represent all the relevant modalities individually, but also capture what happens when they are combined.

3 Which Logics?

In this section we discuss logics that seem suitable to facilitate a representation of the AI Act in HOL.

Standard Deontic Logic (SDL) is the most studied system of deontic logic. It contains a monadic deontic obligation operator depending on an accessibility relation and can successfully express and reason with obligations, permissions, and prohibitions [26]. However, SDL reaches its limits and leads to inconsistencies in CTD scenarios [18].

A logic that enables expressing and reasoning with CTDs is Dyadic Deontic Logic (DDL) by Carmo and Jones [12, 13]. It differentiates between ideal and actual obligations and introduces a conditional obligation operator, thereby enabling the expression of CTDs that arise when ideal obligations have been violated. Related to DDL is Åqvist's system E for conditional obligation [2].

To represent agentive obligations, a modal logic of agency is needed. While there are many ideas for formulating such a logic [1, 11, 17, 23, 38, 39], the most prominent theory of agency is a branch called Seeing-To-It-That (STIT) logic that originated in the works of Belnap [4] and Belnap and Perloff [5]. STIT theory introduces a STIT operator of the form $stit_a\ F$, which expresses that an agent

a sees to it that *F* holds. Given that agency appears in the AI Act mainly via agentive obligations, a suitable agency logic must be able to formulate ought-to-do statements for specific agents. Several authors have researched this issue and presented their approaches, which could be apt to help represent the AI Act [21, 22, 27, 37, 40].

Moving to the representation of agent beliefs, the field called epistemic logic becomes interesting. Epistemic logic allows for the exploration of different ideas of knowledge and beliefs of agents and their logical relations [34]. Popular epistemic logics include modal logics S4 and S5 based on possible world semantics, and providing modal operators representing what agents *know* and *believe*.

For the representation of temporal constructs, we may turn to temporal logic, a class of modal logics that can formally represent and reason with temporality [19]. A popular system is called Tense Logic (TL) and dates back to the work of Prior [25, 31–33]. TL treats propositions as tensed and introduces temporal operators into the language, thereby allowing for the representation of temporal relations like the ones in the AI Act.

Non-monotonic logic provides a way to deal with exceptions from general rules. It is designed to handle defeasible inference, enabling reasoners to retract conclusions when warranted by additional information, such as the presence of conditions allowing for an exception [36].

Finally, a logic that can do justice to vague and fuzzy statements is called fuzzy logic. Fuzzy logic assigns a degree of truth to a proposition that is expressed on a scale within the real unit interval $[0, 1]$, with 0 equal to total falsehood and one equal to total truth [15]. From the point of view of LogiKEy, however, the fuzzy logic variants of Hájek [20] are particularly interesting, since they can be seen as generalized multi-value logics and as such are more amenable to formalization; in fact, [20] provides a good blue-print for doing so in LogiKEy extending the formalization idea presented in [35].

4 Initial Experiments

In this section we give a brief summary of the findings and results from the experiments on embedding some of the logics using the LogiKEy approach and using these logics to represent parts of the AI Act; for details see [24]. Experiments have so far focused on only a selection of the identified modalities, namely obligations, prohibitions, permissions, CTDs, agency, and agentive obligations.

LogiKEy leverages higher-order logic (HOL) as a unifying meta-logic to enable the modelling of diverse object logics via shallow semantic embeddings [6]. This integration is facilitated, for example, within the higher-order proof assistant tool *Isabelle/HOL*, which comes with state-of-the-art automated theorem provers (ATP), satisfiability modulo theories (SMT) solvers, and model finders; however, other proof assistant systems, such as Lean or Coq, or the TPTP infrastructure, could be employed instead.

To represent obligations (as well as permissions and prohibitions), SDL was identified as a suitable logic in Section 3. A trustworthy embedding of SDL in Isabelle/HOL using the LogiKEy approach already exists [7] and has been used to represent parts from the AI Act, particularly Article 5 on prohibited AI systems. This was successful: The paragraphs could be adequately represented, and both Sledgehammer (an automated theorem proving tool integrated with Isabelle/HOL) and Nitpick [10] (a model finder integrated with Isabelle/HOL) could reason with them correctly [24]. Sledgehammer was used to automatically check the validity of example judgements, and Nitpick to find countermodels for invalid judgements and also to prove the consistency of whole contexts.

Like SDL, DDL already has a faithful embedding in Isabelle/HOL [9]. Using this embedding it was possible to represent selected CTDs from the AI Act without problems. The formalization of the selected examples from the AI Act was straightforward, and Nitpick and Sledgehammer could work with them successfully [24]. Since DDL can express everything SDL can and more [18], the DDL embedding is suitable for representing obligations, prohibitions, permissions, and CTDs. Encodings of Article 5 and selected CTDs from the AI Act in DDL can be found in the appendix.[3] Further work includes similar experiments adapting an existing embedding of Åqvist's system E in Isabelle/HOL [30].

Additionally, logics for the representation of agency and agentive obligations were explored. In particular, an embedding into HOL of Temporal Deontic STIT Logic [37], a logic appropriate for representing both actions of agents and agentive obligations, was created. While most axioms postulated in [37] were provable via Sledgehammer and none were refuted, Nitpick could not find a model confirming the consistency of the embedding. This indicates that TDS logic has (presumably) only infinite models, making it unsuitable for model finding/checking in the context of any formalization of AI Act [24] with existing tools.

As an alternative to TDS, the authors explored the possibility of extending DDL with additional operators for the representation of agency and agentive obligations. Two different variants were introduced: One representing agent types as constants, with separate accessibility relations and obligation operators for each agent, and one representing agent types as function types, with a general accessibility relation and a general obligation operator taking an agent as an input parameter. Both variants introduced a STIT-like operator as a constant and equipped it with minimal axiomatization, namely to ensure that anything an agent sees to actually holds [24].

While the first variant is more restrictive, with agentive obligations always applying to a generic agent type without the possibility of specifying further constraints or relations, the second variant is more flexible, allowing groupings based on characteristics of agent types or relations and formulating obligations for specific subsets of agents. Unfortunately, the second variant performs worse in other aspects: It struggles with the representation of CTDs, and Nitpick can not find a model for cardinalities of i (domain of possible worlds) bigger than 1. For the first variant, a model is found up to a cardinality of i=2, and CTDs can be represented without problems. For both variants, all but one of the prominent lemmas for DDL as studied by Benzmüller et al. could be proven [24]. Whether either of the variants can be adapted or simplified to perform better remains an open question. The embedding of TDS and the two extensions of DDL can be found in the appendix.

The experiments have shown that while elementary concepts such as obligations and CTDs can be formalized and reasoned with correctly in HOL, the representation of complex constructs such as agency requires more sophisticated logics. Reasoning can be

[3]The Isabelle/HOL source files can also be downloaded here: http://logikey.org/tree/master/2025-ICAIL-Data.

challenging for model finders and automated theorem provers, and further experiments need to be conducted to explore feasibility and automation performance as the studied contexts grow.

5 Conclusion

An extensive analysis of the AI Act was carried out, which led to the identification of different contained modalities that pose a challenge to attempts to formalise and reason with it: obligations, permissions, prohibitions, CTDs, agency and agentive obligations, beliefs of agents, temporality and temporal notions, fuzziness, and exceptions. Suggestions for logics suitable to represent these modalities were presented, and results of some first experiments with these logics and examples from the AI Act were reported. As of now, experiments have only been conducted for the following modalities (and some combinations of) : obligations, permissions, prohibitions, CTDs, agency, and agentive obligations.

While the presented project is not yet completed, the reported material provides a starting point and references for future research. In the LogiKEy context, the challenge is to create embeddings of the suggested logics, choose and provide suitable logic combinations, and experiment with increasingly larger parts of the AI Act.

Based on the information gathered here, the possibility of representing legal information in HOL and using it in automated reasoning should be further explored and improved.

Acknowledgements. We are grateful to the anonymous reviewers for their helpful feedback and comments.

References

[1] A. R. Anderson. 1962. Logic, Norms, and Roles. *Ratio (Misc.)* 4, 1 (1962), 36–49.
[2] Lennart Åqvist. 2002. Deontic Logic. In *Handbook of Philosophical Logic: Volume 8*, D. M. Gabbay and F. Guenthner (Eds.). Springer Netherlands, Dordrecht, 147–264. https://doi.org/10.1007/978-94-010-0387-2_3
[3] Tara Athan, Guido Governatori, Monica Palmirani, Adrian Paschke, and Adam Wyner. 2015. LegalRuleML: Design Principles and Foundations. In *Reasoning Web. Web Logic Rules*, Wolfgang Faber and Adrian Paschke (Eds.). Vol. 9203. Springer, Cham, 151–188. https://doi.org/10.1007/978-3-319-21768-0_6
[4] Nuel Belnap. 1991. Backwards and Forwards in the Modal Logic of Agency. *Philosophy and Phenomenological Research* 51, 4 (Dec. 1991), 777. https://doi.org/10.2307/2108182
[5] Nuel Belnap and Michael Perloff. 1988. Seeing to It That: A Canonical Form for Agentives. *Theoria* 54, 3 (1988), 175–199. https://doi.org/10.1111/j.1755-2567.1988.tb00717.x
[6] Christoph Benzmüller. 2019. Universal (Meta-)Logical Reasoning: Recent Successes. *Science of Computer Programming* 172 (2019), 48–62. https://doi.org/10.1016/j.scico.2018.10.008
[7] Christoph Benzmüller and Xavier Parent. 2018. First Experiments with a Flexible Infrastructure for Normative Reasoning. https://doi.org/10.48550/ARXIV.1804.02929
[8] Christoph Benzmüller, Xavier Parent, and Leendert van der Torre. 2020. Designing Normative Theories for Ethical and Legal Reasoning: LogiKEy Framework, Methodology, and Tool Support. *Artificial Intelligence* 287 (2020), 103348. https://doi.org/10.1016/j.artint.2020.103348
[9] Christoph Benzmüller, Ali Farjami, and Xavier Parent. 2022. Dyadic Deontic Logic in HOL: Faithful Embedding and Meta-Theoretical Experiments. In *New Developments in Legal Reasoning and Logic*, Shahid Rahman, Matthias Armgardt, and Hans Christian Nordtveit Kvernenes (Eds.). Vol. 23. Springer International Publishing, Cham, 353–377. https://doi.org/10.1007/978-3-030-70084-3_14
[10] Jasmin Christian Blanchette and Tobias Nipkow. 2010. Nitpick: A Counterexample Generator for Higher-Order Logic Based on a Relational Model Finder. In *ITP 2010 (Lecture Notes in Computer Science, Vol. 6172)*, Matt Kaufmann and Lawrence C. Paulson (Eds.). Springer, 131–146. https://doi.org/10.1007/978-3-642-14052-5_11
[11] Mark A. Brown. 1988. On the logic of ability. *Journal of Philosophical Logic* 17, 1 (Feb. 1988), 1–26. https://doi.org/10.1007/BF00249673
[12] José Carmo and Andrew J. I. Jones. 2002. Deontic Logic and Contrary-to-Duties. In *Handbook of Philosophical Logic: Volume 8*, D. M. Gabbay and F. Guenthner (Eds.). Springer, Dordrecht, 265–343. https://doi.org/10.1007/978-94-010-0387-2_4
[13] José Carmo and Andrew J. I. Jones. 2022. Carmo and Jones' logic for contrary-to-duty obligations revised. *J. Log. Comput.* 32, 7 (2022), 1352–1364. https://doi.org/10.1093/LOGCOM/EXAC026
[14] R. M. Chisholm. 1963. Contrary-to-Duty Imperatives and Deontic Logic. *Analysis* 24, 2 (1963), 33–36. https://doi.org/10.1093/analys/24.2.33
[15] Petr Cintula, Christian G. Fermüller, and Carles Noguera. 2023. Fuzzy Logic. In *The Stanford Encyclopedia of Philosophy* (Summer 2023 ed.), Edward N. Zalta and Uri Nodelman (Eds.). Metaphysics Research Lab, Stanford University, Stanford, CA, USA.
[16] EU commission. 2024. AI Act. https://eur-lex.europa.eu/eli/reg/2024/1689/oj
[17] Frederic B. Fitch. 1963. A Logical Analysis of Some Value Concepts. *The Journal of Symbolic Logic* 28, 2 (1963), 135–142. http://www.jstor.org/stable/2271594
[18] D. Gabbay, J. Horty, X. Parent, R. van der Meyden, and L. van der Torre (Eds.). 2013. *Handbook of Deontic Logic and Normative Systems*. Vol. 1. College Publications, London, UK.
[19] Valentin Goranko and Antje Rumberg. 2023. Temporal Logic. In *The Stanford Encyclopedia of Philosophy* (Fall 2023 ed.), Edward N. Zalta and Uri Nodelman (Eds.). Metaphysics Research Lab, Stanford University, Stanford, CA, USA.
[20] Petr Hájek. 1998. *Metamathematics of Fuzzy Logic*. Trends in Logic, Vol. 4. Kluwer, Dordrecht. https://doi.org/10.1007/978-94-011-5300-3
[21] John Horty. 2001. *Agency and Deontic Logic*. Oxford University Press, New York.
[22] John F. Horty and Nuel Belnap. 1995. The Deliberative Stit: A Study of Action, Omission, Ability, and Obligation. *Journal of Philosophical Logic* 24, 6 (1995), 583–644. http://www.jstor.org/stable/30227231
[23] Stig Kanger. 1972. Law and logic. *Theoria* 38, 3 (1972), 105–132. https://doi.org/10.1111/j.1755-2567.1972.tb00928.x
[24] Lara Lawniczak. 2024. *Towards Automated Ethical Reasoning: Representing the AI Act in Higher-Order-Logic*. Master's thesis. Otto-Friedrichs-Universität Bamberg.
[25] Czestaw Lejewski. 1959. Time and Modality, by A. N. Prior, Clarendon Press: Oxford University Press, 1957. Pp. Viii + 148. *Philosophy* 34, 128 (1959), 56–. https://doi.org/10.1017/s0031819100029776
[26] Paul McNamara and Frederik Van De Putte. 2022. Deontic Logic. In *The Stanford Encyclopedia of Philosophy* (Fall 2022 ed.), Edward N. Zalta and Uri Nodelman (Eds.). Metaphysics Research Lab, Stanford University, Stanford, CA, USA.
[27] Y. Murakami. 1998. Utilitarian Deontic Logic. In *Advances in Modal Logic*, Marcus Kracht, Maarten de Rijke, Heinrich Wansing, and Michael Zakharyaschev (Eds.). CSLI Publications, Stanford, CA, USA, 211–230.
[28] OASIS Open. 2018. *Akoma Ntoso Version 1.0*. OASIS. https://www.oasis-open.org/2018/09/11/akoma-ntoso-v1-0-akn-oasis-standard-published/
[29] Monica Palmirani, Guido Governatori, Antonino Rotolo, Said Tabet, Harold Boley, and Adrian Paschke. 2011. LegalRuleML: XML-based rules and norms. In *RuleML 2011*, Frank Olken, Monica Palmirani, and Davide Sottara (Eds.). LNCS, Vol. 7018. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-24908-2
[30] Xavier Parent and Christoph Benzmüller. 2024. Conditional Normative Reasoning as a Fragment of HOL. *Journal of Applied Non-Classical Logics* 34, 4 (2024), 561–592. https://doi.org/10.1080/11663081.2024.2386917
[31] A. N. Prior. 1959. Thank Goodness That's Over. *Philosophy* 34, 128 (1959), 12–17. https://doi.org/10.1017/s0031819100029685
[32] Arthur N. Prior. 1967. *Past, Present and Future*. Clarendon P., Oxford,.
[33] Arthur N. Prior. 1968. *Papers on Time and Tense*. Oxford University Press, UK, Oxford, UK.
[34] Rasmus Rendsvig, John Symons, and Yanjing Wang. 2024. Epistemic Logic. In *The Stanford Encyclopedia of Philosophy* (Summer 2024 ed.), Edward N. Zalta and Uri Nodelman (Eds.). Metaphysics Research Lab, Stanford University, Stanford, CA, USA.
[35] Alexander Steen and Christoph Benzmüller. 2016. Sweet SIXTEEN: Automation via Embedding into Classical Higher-Order Logic. *Logic and Logical Philosophy* 25 (2016), 535–554. https://doi.org/10.12775/LLP.2016.021
[36] Christian Strasser and G. Aldo Antonelli. 2019. Non-monotonic Logic. In *The Stanford Encyclopedia of Philosophy* (summer 2019 ed.), Edward N. Zalta (Ed.). Metaphysics Research Lab, Stanford University, Stanford, CA, USA.
[37] Kees van Berkel and Tim Lyon. 2019. A Neutral Temporal Deontic STIT Logic. In *LORI 2019*, Patrick Blackburn, Emiliano Lorini, and Meiyun Guo (Eds.). LNCS, Vol. 11813. Springer, 340–354. https://doi.org/10.1007/978-3-662-60292-8_25
[38] Georg Henrik Von Wright. 1963. *Norm and Action: A Logical Enquiry*. Routledge and Kegan Paul, New York, NY, USA.
[39] Georg Henrik Von Wright. 1981. On the Logic of Norms and Actions. In *New Studies in Deontic Logic: Norms, Actions, and the Foundations of Ethics*, Risto Hilpinen (Ed.). Springer Netherlands, Dordrecht, 3–35. https://doi.org/10.1007/978-94-009-8484-4_1
[40] Ming Xu. 2015. Combinations of Stit with Ought and Know. *Journal of Philosophical Logic* 44, 6 (2015), 851–877. https://doi.org/10.1007/s10992-015-9365-7

6 Appendix

6.1 Article 5 in DDL

The following screenshots illustrate the representation of the following part of Article 5 in Isabelle HOL:

> "1. The following AI practices shall be prohibited:
> (a) the placing on the market, the putting into service or the use of an AI system that deploys subliminal techniques beyond a person's consciousness or purposefully manipulative or deceptive techniques, with the objective, or the effect of materially distorting the behaviour of a person or a group of persons by appreciably impairing their ability to make an informed decision, thereby causing them to take a decision that they would not have otherwise taken in a manner that causes or is reasonably likely to cause that person, another person or group of persons significant harm;
> (b) the placing on the market, the putting into service or the use of an AI system that exploits any of the vulnerabilities of a natural person or a specific group of persons due to their age, disability or a specific social or economic situation, with the objective, or the effect, of materially distorting the behaviour of that person or a person belonging to that group in a manner that causes or is reasonably likely to cause that person or another person significant harm;
> (c) the placing on the market, the putting into service or the use of AI systems for the evaluation or classification of natural persons or groups of persons over a certain period of time based on their social behaviour or known, inferred or predicted personal or personality characteristics, with the social score leading to either or both of the following:
> (i) detrimental or unfavourable treatment of certain natural persons or groups of persons in social contexts that are unrelated to the contexts in which the data was originally generated or collected;
> (ii) detrimental or unfavourable treatment of certain natural persons or groups of persons that is unjustified or disproportionate to their social behaviour or its gravity;" [16, §5(1)].

As hinted at in Figure 1 in the import section, the embedding of DDL in Isabelle/HOL from [9] is reused here.

Lines 7-14 define constants, which are then used in lines 16-27 to define the rules given in paragraph 1, a-c. Lines 29-42 construct an example case and test two lemmas. The first constructs a conclusion that should hold given the rules and the properties of the example system. As expected, it can be proven using the Sledgehammer tool. On the other hand, the second one is missing part of the information from the previous lemma and should not hold. Nitpick finds a counterexample.

Note how the formalization of the theorem statements in Figure 2 distinguishes between global (H1-C1) and local (F1-F3) assumptions.

```
1  theory prohibited
2    imports
3     DDL
4     types
5  begin
6
7  consts (*Predicates/relations*)
8    subliminal_technique::"aiSys⇒σ" (*deploys a subliminal technique beyond a persons consciousness*)
9    has_consequence::"aiSys⇒consequence⇒σ" (*system has or may have a consequence*)
10   has_purpose::"aiSys⇒purpose⇒σ" (*system has a purpose*)
11   exploits_vulnerabilities_group::"aiSys⇒quality_person⇒σ"
12   exploits_vulnerable_group::"aiSys⇒σ" (*exploits any of the vulnerabilities of a specific group due to a certain quality*)
13   employed_by_pauthorities::"aiSys⇒σ" (*employed by public authorities or on their behalf*)
14   prohibited :: "aiSys ⇒ σ" (*placing on market, putting into service, or use*)
15
16 abbreviation "H1 ≡ ⌊∀x::aiSys. ((has_consequence x harm_physical) ∨
17 (has_consequence x harm_psychological)) ↔ has_consequence x harm⌋"
18 abbreviation "H2 ≡ ⌊∀x::aiSys. ((exploits_vulnerabilities_group x age) ∨
19 (exploits_vulnerabilities_group x physcial_disability) ∨ (exploits_vulnerabilities_group x mental_disability))
20  ↔ exploits_vulnerable_group x⌋"
21 abbreviation "A1 ≡ ⌊∀x::aiSys. subliminal_technique x  ∧  has_consequence x harm ∧
22 has_purpose x distort_behavior ⟶ ○<prohibited x>⌋"
23 abbreviation "B1 ≡ ⌊∀x::aiSys. exploits_vulnerable_group x ∧ has_purpose x distort_behavior ∧
24 has_consequence x harm ⟶ ○<prohibited x>⌋"
25 abbreviation "C1 ≡ ⌊∀x::aiSys. employed_by_pauthorities x ∧ has_purpose x eval_trustworthiness_over_time ∧
26 (has_consequence x detri_treatment_unrelated_context ∨ has_consequence x detri_treatment_unjustified_disprop)
27 ⟶ ○<prohibited x>⌋"
```

Figure 1: Article 5 in DDL, Part 1

```
consts
  x::aiSys

abbreviation "F1 w ≡ (subliminal_technique x) w"
abbreviation "F2 w ≡ (has_consequence x harm_physical) w"
abbreviation "F3 w ≡ (has_purpose x distort_behavior) w"

theorem Result1a: "H1 ∧ H2 ∧ A1 ∧ B1 ∧ C1 ⟶ ⌊(F1 ∧ F2 ∧ F3)  → (○<prohibited x>)⌋"
  by metis

theorem Result1b: "H1 ∧ H2 ∧ A1 ∧ B1 ∧ C1 ⟶ ⌊F1 ∧ F2  → (○<prohibited x>)⌋"
  nitpick [user_axioms] oops (*counterexample found*)

lemma True nitpick [satisfy, user_axioms, card i =3, show_all] oops
```

Figure 2: Article 5 in DDL, Part 2

6.2 Selected CTD Examples in DDL

Figures 3 and 4 show representations of two CTDs. The first arises from a combination of Articles 16 and 20:

> "Providers of high-risk AI systems shall:
> (a) ensure that their high-risk AI systems are compliant with the requirements set out in Section 2; [16, §16]
>
> 1. Providers of high-risk AI systems which consider or have reason to consider that a high-risk AI system that they have placed on the market or put into service is not in conformity with this Regulation shall immediately take the necessary corrective actions to bring that system into conformity, to withdraw it, to disable it, or to recall it, as appropriate. They shall inform the distributors of the high-risk AI system concerned and, where applicable, the deployers, the authorised representative and importers accordingly." [16, §20]

The same excerpt from Article 16 results in another CTD when combined with the following part from Article 24:

> "2. Where an importer has sufficient reason to consider that a high-risk AI system is not in conformity with this Regulation, or is falsified, or accompanied by falsified documentation, it shall not place the system on the market until it has been brought into conformity." [16, §24]

Again, the embedding of DDL is used [9].

In both cases, consistency is proven and Isabelle can successfully reason in the CTD structure, with Sledgehammer proving the desired lemmas and Nitpick providing counterexamples wrong statements.

```
consts
  compliance_req::"aiSys⇒σ"
  inform_com_auth::"aiSys⇒σ"
  kill_everyone::"aiSys⇒σ"

(*CTD example DDL:*)
consts
  l::aiSys

(*interesting part: CTD; Trying to create the typical structure*)
axiomatization where
A0: "⌊(high_risk l)⌋" and
A1: "⌊∀x::aiSys.(high_risk x) → ○<(compliance_req x)>⌋" and
A2: "⌊∀x::aiSys. ¬(compliance_req x) ∧ (high_risk x) → ○<(inform_com_auth x)>⌋" and
(*implicit: If the compliance with the requirements is a given, the provider is obligated to not inform authorities
of non-compliance (since that would make no sense*)
A3: "⌊∀x::aiSys. ○<(compliance_req x) ∧ (high_risk x) → (¬(inform_com_auth x))>⌋" and
Situation: "⌊¬ (compliance_req l)⌋ι"

(***Some Experiments***)
lemma True nitpick [satisfy, user_axioms, show_all] oops(*Consistency-check: Nitpick finds a model.*)

lemma "⌊○<(inform_com_auth l)>⌋ι" using A0 A2 Situation by auto
lemma "⌊○<¬(inform_com_auth l)>⌋ι" (*nitpick [user_axioms] (*counterexample found*)*) oops
lemma "⌊○<(kill_everyone l)>⌋ι" (*nitpick[user_axioms] (*counterexample found*)*) oops
```

Figure 3: Article 20, CTD Example in DDL

```
consts
  system_in_conformity::"aiSys⇒σ"
  not_on_market::"aiSys⇒σ"

(*CTD structure*)
consts l::aiSys

axiomatization where
A0: "⌊high_risk l⌋ι" and
A1: "⌊∀x::aiSys. (high_risk x) → ○<(system_in_conformity x)>⌋" and
A2: "⌊∀x::aiSys. ¬ (system_in_conformity x) ∧ (high_risk x) → ○<(not_on_market x)>⌋" and
(*implicit: If the system is in conformity, the importer is obligated to not see to it that the system is not placed
on the market*)
A3: "⌊∀x::aiSys. ○<(system_in_conformity x) → (¬(not_on_market x))>⌋" and
Situation: "⌊¬ (system_in_conformity l)⌋ι"

(***Some Experiments***)
lemma True nitpick [satisfy, user_axioms, show_all] oops (*Consistency-check: Nitpick finds a model.*)

lemma "⌊○<(not_on_market l)>⌋ι" using A0 A2 Situation by auto
lemma "⌊○<¬(not_on_market l)>⌋ι" nitpick [user_axioms, show_all] oops (*counterexample found*)
```

Figure 4: Article 24, CTD Example in DDL

6.3 TDS Embedding

Figures 5, 6, 7, 8, and 9 show the embedding of TDS that has been created by the first author.

Lines 8-17 define the types of worlds, agents, and the accessibility relations. The constants defined in lines 19-30 represent the current world, accessibility relations of the logic, and the set of agents. In the axiomatization, the first author defines the set of agents as containing the two agents a1 and a2, specify reflexivity, symmetry, and transitivity for all equivalence relations, seriality, transitivity, and the inverse relation for the future relation RG, and the constraints imposed on the different relations (lines 37-83). Afterward, lines 85-94 define the lifted connectives, whereas lines 96-111 define the operators of TDS logic. Lines 113-115 define notions of validity. Finally, line 117 calls Nitpick to find a model of the embedding. As explained in Section 4, this does not succeed. The infinity proof is visible in Figures 10 and 11.

For a complete explanation of the TDS embedding and the infinity proof, please refer to [24].

```
theory Tstit_Deontic_clean (*TDS logic*)
  imports Main
begin

declare [[show_types]]
nitpick_params[user_axioms, show_all, format=2] (*global parameter setting for nitpick*)

typedecl i (*possible world*)
type_synonym σ = "(i⇒bool)"
type_synonym γ = "σ⇒σ"
type_synonym ϱ = "σ⇒σ⇒σ"
type_synonym δ = "i⇒i⇒bool" (* type of accessibility relations between worlds *)

datatype ag = a1 | a2  (* datatype of mutually different agents; we provide 2, more can be added as needed *)

type_synonym ω = "ag⇒i⇒i⇒bool"   (* type of agent dependent accessibility relations between worlds *)
type_synonym ν = "(ag⇒bool)⇒i⇒i⇒bool" (* type of set-of-agents dependent accessibility relations between worlds *)

```

Figure 5: TDS Embedding Part 1

```
consts
  cw::i (*current world*)

  RBox::δ (*worlds that are alternatives to each other: if (w, w1) then w1 is an alternative to w*)
  R_ag::ω (*worlds that are actual choices for agent a, set of alternatives that are forced by agents i choice or action at world w*)
  R_set::ν (*worlds that are actual choices for the set of agents Ag*)
  R_ag_ought::ω (*set of alternatives that agent a ought to chose at moment m*)

  RG::δ (*all worlds that are the strict future of world w: (w, w1) means that w1 is the strict future of w*)
  RH::δ (*all worlds that are the strict past of world w: (w, w1) means that w1 is the strict past of w*)

  Ag::"ag⇒bool" (*set of all agents*)

definition Inv::"δ ⇒ δ" ("Inv _") where
  "Inv R ≡ λy x. R x y"

lemma True nitpick [satisfy, user_axioms, show_all] oops (*empty assignment*)

axiomatization where
  a1Set: "Ag a1" and
  a2Set: "Ag a2"
```

Figure 6: TDS Embedding Part 2

```
axiomatization where
 (*reflexivity, symmetry, and transitivity for all equivalence relations*)
  accReR_ag:  "∀a w. (R_ag a) w w" and
  accSymR_ag: "∀a w v. (R_ag a) w v ⟶ (R_ag a) v w" and
  accTraR_ag: "∀a w v u. ((R_ag a) w v ∧ (R_ag a) v u) ⟶ (R_ag a) w u" and

  accReRBox:  "∀w. RBox w w" and
  accSymRBox: "∀w v. RBox w v ⟶ RBox v w" and
  accTraRBox: "∀w v u. (RBox w v ∧ RBox v u) ⟶ RBox w u" and

  accReR_set:  "∀s w. (R_set s) w w" and
  accSymR_set: "∀s w v. (R_set s) w v ⟶ (R_set s) v w" and
  accTraR_set: "∀s w v u. ((R_set s) w v ∧ (R_set s) v u) ⟶ (R_set s) w u" and

  RG_serial: "(∀x. (∃y. (RG x y)))" and (*seriality of RG*)
  RG_trans: "(∀x y z. (RG x y) ∧ (RG y z) ⟶ (RG x z))" and (*transitivity of RG*)
  Inv: "(Inv RG) = RH" and (*RH is the inverse of RG*)

  C1: "∀a w1 w2. (R_ag a) w1 w2 ⟶ RBox w1 w2" and (*agents can only choose between alternatives*)
  (*independence of agents/choices → if w1 and w2 are alternatives to each other, there exists a world w which is
  part of the actual choice of all agents → see tests*)
  C2: "∀w1 w2. (RBox w1 w2) ⟶ (∃w. ∀a. (R_ag a) w1 w)" and
  (*independence of agents/choices → if w1 and w2 are alternatives to each other, there exists a world w which is
  part of the actual choice of all agents*)
  C3: "∀S w1 w2. ((R_set S) w1 w2) ⟶ (∀a. S a ⟶ (R_ag a) w1 w2)" and (*choices of agents in group Agt are
  made up of the choices of the intersection of choices of each individual agent*)

  T4: "∀u v w. ((RG w u) ∧ (RG w v)) ⟶ ((RG v u) ∨ (RG u v) ∨ u = v)" and (*future*)
  T5: "∀u v w. ((RH w u) ∧ (RH w v)) ⟶ ((RH v u) ∨ (RH u v) ∨ u = v)" and (*past*)
  (* If v is in the future of w and u and v are in the same moment,  then there exists an alternative z
  in the collective choice of all agents at w such that u is in the future of z.*)
  T6: "∀v w u S. (RG w v) ∧ (RBox v u) ⟶ (∃z. ((R_set S) w z) ∧ (RG z u))" and
  T7: "∀w v. ((RBox w v)) ⟶ ¬(RG w v)" and (*if worlds are in the same moment, they can't be in each others future*)
```

Figure 7: TDS Embedding Part 3

```
  (*ideal worlds accessible at a moment are alternatives to the current world*)
  D8: "∀a. ∀w v. ((R_ag_ought a) w v) ⟶ (RBox w v)" and
  (*at every moment for each agent there is a choice available that is an ideal choice*)
  D9: "∀a. ∀w. (∃v. (RBox w v) ∧ (∀u. ((R_ag a) v u) ⟶ ((R_ag_ought a) w u)))" and
  (*for each agent, if a world is ideal from the perspective of a particular world at a moment, that world is ideal from
  the perspective of any world at that same moment, ideal worlds are settled upon moments*)
  D10: "∀a. ∀w v u z. (RBox w v) ∧ (RBox w u) ∧ ((R_ag_ought a) u z) ⟶ ((R_ag_ought a) v z)" and
  (*Every ideal world extends to a complete ideal choice, no choice contains both ideal and non-ideal worlds*)
  D11: "∀a. ∀w v. ((R_ag_ought a) w v) ⟶ (∃u. (RBox w u) ∧ ((R_ag a) u v) ∧ (∀z. ((R_ag a) u z) ⟶ ((R_ag_ought a) w z)))"

  (*Logical connectives lifted*)
  abbreviation tdsNot::γ ("¬_") where "¬A ≡ λw. ¬A(w)"
  abbreviation tdsAnd::ϱ ("_∧_") where "A∧B ≡ λw. A(w)∧B(w)"
  abbreviation tdsOr::ϱ ("_ ∨ _") where "A∨B ≡ λw. A(w)∨B(w)"
  abbreviation tdsImp::ϱ ("_⟶_") where "A⟶B ≡ λw. A(w)⟶B(w)"
  abbreviation tdsEquiv::ϱ ("_⟷_") where "A⟷B ≡ λw. A(w)⟷B(w)"
  abbreviation tdsBox::γ ("□") where "□A ≡ λw. ∀v. A(v)"
  abbreviation tdsDia::γ ("◇") where "◇A ≡ ¬□(¬A)"
  abbreviation tdsTop::σ ("⊤") where "⊤ ≡ λw. True"
  abbreviation tdsBot::σ ("⊥") where "⊥ ≡ λw. False"

  (*Operators*)
  abbreviation tdsCstit::"ag⇒γ" ("[_] _") where "[i] A ≡ λw. (∀y. ((R_ag i) w y) ⟶ (A y))"  (*Chellas Stit*)
  abbreviation tdsCstitPoss::"ag⇒γ" ("<_>_") where "<i> A ≡ ¬([i] (¬ A))" (*Possibility Group Chellas stit*)
  abbreviation tdsCstitGr::"γ" ("[Ag]_") where "[Ag] A ≡ λw. (∀v. ((R_set Ag) w v) ⟶ (A v))" (*Chellas stit group*)
  abbreviation tdsCstitGrPoss::"γ" ("<Ag>_") where "<Ag> A ≡ ¬([Ag] (¬ A))"
  abbreviation tdsDstit::"ag⇒γ" ("[_]d _") where "[i]d A ≡ ([i]A) ∧ ¬(□A)" (*Dstit*)
  abbreviation tdsDstitPoss::"ag⇒γ" ("<_>d _") where "<i>d A ≡ ¬ ([i]d (¬ A))" (*Dstit Poss*)
  abbreviation tdsDstitGr::γ ("[Ag]d_") where "[Ag]d A ≡ ([Ag] A) ∧ (¬(□A))" (*Dstit group*)
  abbreviation tdsDstitGrPoss::γ ("<Ag>d_") where "<Ag>d A ≡ ¬([Ag]d(¬ A))"
  abbreviation tdsOughtToDo::"ag⇒γ" ("⊗ _ _") where "⊗ i A ≡ λw. (∀v. ((R_ag_ought i) w v) ⟶ (A v))" (*OughtToDo Operator*)
  abbreviation tdsOughtToDoD::"ag⇒γ" ("⊗d _ _") where "⊗d i A ≡ (⊗ i A) ∧ (◇ ¬ A)" (*OughtToDo Operator*)
```

Figure 8: TDS Embedding Part 4

```
  abbreviation tdsG::γ ("G_") where "G A ≡ λw. (∀v. ((RG w v) ⟶ (A v)))" (*A will always be true in the future*)
  abbreviation tdsH::γ ("H_") where "H A ≡ λw. (∀v. ((RH w v) ⟶ (A v)))" (*A has always been true in the past*)
  abbreviation tdsP::γ ("P_") where "P A ≡ ¬ (H (¬ A))" (*it has not always been true that not A*)
  abbreviation tdsF::γ ("F_") where "F A ≡ ¬ (G (¬ A))" (*it will not always be true that not A*)

  (*Validity*)
  abbreviation tdsValidLocal :: "(i⇒bool) ⇒ bool" ("⊨ _") where "⊨ A ≡ A cw"
  abbreviation tdsValidGlobal :: "(i ⇒ bool) ⇒ bool" ("⌊_⌋") where "⌊A⌋ ≡ ∀w. A w"

lemma True nitpick [satisfy, user_axioms, show_all] oops

end
```

Figure 9: TDS Embedding Part 5

```
62  abbreviation "infinity ≡ ∃M. (∃z::i. ¬(M z) ∧ (∃G. (∀y::i. (∃x. (M x) ∧ (G x) = y))))"
63
64  lemma assumes
65  (*axioms for RG and RH*)
66  RG_serial: "(∀x. (∃y. (RG x y)))" and (*seriality of RG*)
67  RG_trans: "(∀x y z. (RG x y) ∧ (RG y z) ⟶ (RG x z))" and (*transitivity of RG*)
68  C7: "∀w v. ((RBox w v) ∧ w ≠ v) ⟶ ¬(RG w v)" (*if worlds are in the same moment, they can't be in each others future*)
69  shows "infinity" nitpick [show_all, user_axioms] (* countermodel found, but only with the unwanted addition in C7*)
```

Proof state ☑ Auto update Update Search:

```
Nitpicking formula...
Nitpick found a counterexample for card i = 1:
  Types:
    i × i [boxed] = {(i₁, i₁)}
    Tstit_Deontic.ag = {a1, a2}
  Constants:
    Ag = (λx. _)(a1 := True, a2 := True)
    RBox = (λx. _)(i₁ := (λx. _)(i₁ := True))
    RG = (λx. _)(i₁ := (λx. _)(i₁ := True))
    R_ag = (λx. _)(a1 := (λx. _)(i₁ := (λx. _)(i₁ := True)), a2 := (λx. _)(i₁ := (λx. _)(i₁ := True)))
    R_set =
      (λx. _)
      ((λx. _)(a1 := True, a2 := True) := (λx. _)(i₁ := (λx. _)(i₁ := True)),
       (λx. _)(a1 := True, a2 := False) := (λx. _)(i₁ := (λx. _)(i₁ := True)),
       (λx. _)(a1 := False, a2 := True) := (λx. _)(i₁ := (λx. _)(i₁ := True)),
       (λx. _)(a1 := False, a2 := False) := (λx. _)(i₁ := (λx. _)(i₁ := True)))
```

Figure 10: TDS Infinity Proof Part 1

```
71  lemma assumes
72  (*axioms for RG and RH*)
73  RG_serial: "(∀x. (∃y. (RG x y)))" and (*seriality of RG*)
74  RG_trans: "(∀x y z. (RG x y) ∧ (RG y z) ⟶ (RG x z))" and (*transitivity of RG*)
75  C7: "∀w v. ((RBox w v)) ⟶ ¬(RG w v)" (*if worlds are in the same moment, they can't be in each others future*)
76  shows "infinity" nitpick [show_all, user_axioms]
```

Proof state ☑ Auto update Update Search:

```
Nitpicking formula...
Nitpick ran out of time
```

Figure 11: TDS Infinity Proof Part 2

6.4 Extended DDL First Variant

Figures 12, 13, 14, and 15 show the first variant of Extended DDL created by the first author [24].

In contrast to the usual DDL embedding, this version uses two additional sets of accessibility relations, which relate to two different agents d and b (lines 10-11). They are axiomatized just like the general accessibility relations of DDL (lines 28-48). These are then used to define generalized obligation operators which take relations (the general one or the ones for the agents) as inputs (lines 67-70). Shortcuts for the general obligation operator and the two agentive obligation operators are declared in lines 84-87.

Additionally, a constant representing the stit operator is introduced in line 14 and axiomatized in line 50.

A more detailed explanation can be found in [24].

```
theory DDL_agents_clean  (*DDL including STIT operator and agentive obligations*)
  imports
    Main
    types
begin

consts
cw::i (*current world*)
av::"i⇒σ" pv::"i⇒σ" ob::"σ⇒(σ⇒bool)"    (*general accessibility relations*)
avd::"i⇒σ" pvd::"i⇒σ" obd::"σ⇒(σ⇒bool)" (*accessibility relations for agent d*)
avb::"i⇒σ" pvb::"i⇒σ" obb::"σ⇒(σ⇒bool)" (*accessibility relations for agent b*)

(*stit operator*)
stit::"ag⇒σ⇒σ" (*ag sees to it that*)

axiomatization where
  ax_distinct: "d ≠ b" and
  ax_3a: "∀w.∃x. av(w)(x)" and ax_4a: "∀w x. av(w)(x) ⟶ pv(w)(x)" and ax_4b: "∀w. pv(w)(w)" and
  ax_5a: "∀X.¬ob(X)(λx. False)" and
  ax_5b: "∀X Y Z. (∀w. ((Y(w) ∧ X(w)) ⟷ (Z(w) ∧ X(w)))) ⟶ (ob(X)(Y) ⟷ ob(X)(Z))" and
  ax_5ca: "∀X β. ((∀Z. β(Z) ⟶ ob(X)(Z)) ∧ (∃Z. β(Z))) ⟶
      (((∃y. ((λw. ∀Z. (β Z) ⟶ (Z w))(y) ∧ X(y))) ⟶ ob(X)(λw. ∀Z. (β Z) ⟶ (Z w))))" and
  ax_5c: "∀X Y Z. (((∃w. (X(w) ∧ Y(w) ∧ Z(w))) ∧  ob(X)(Y)  ∧  ob(X)(Z))  ⟶  ob(X)(λw. Y(w) ∧ Z(w)))" and
  ax_5d: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ ob(X)(Y) ∧ (∀w. X(w) ⟶ Z(w)))
                    ⟶ ob(Z)(λw. (Z(w) ∧ ¬X(w)) ∨ Y(w))" and
  ax_5e: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ ob(X)(Z) ∧ (∃w. Y(w) ∧ Z(w))) ⟶ ob(Y)(Z)" and
```

Figure 12: Extended DDL First Variant Part 1

```
  (*for agent d: providers*)
  axd_3a: "∀w.∃x. avd(w)(x)" and axd_4a: "∀w x. avd(w)(x) ⟶ pvd(w)(x)" and axa_4ba: "∀w. pvd(w)(w)" and
  axd_5a: "∀X.¬obd(X)(λx. False)" and
  axd_5b: "∀X Y Z. (∀w. ((Y(w) ∧ X(w)) ⟷ (Z(w) ∧ X(w)))) ⟶ (obd(X)(Y) ⟷ obd(X)(Z))" and
  axd_5ca: "∀X β. ((∀Z. β(Z) ⟶ obd(X)(Z)) ∧ (∃Z. β(Z))) ⟶
      (((∃y. ((λw. ∀Z. (β Z) ⟶ (Z w))(y) ∧ X(y))) ⟶ obd(X)(λw. ∀Z. (β Z) ⟶ (Z w))))" and
  axd_5c: "∀X Y Z. (((∃w. (X(w) ∧ Y(w) ∧ Z(w))) ∧  obd(X)(Y)  ∧  obd(X)(Z))  ⟶  obd(X)(λw. Y(w) ∧ Z(w)))" and
  axd_5d: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ obd(X)(Y) ∧ (∀w. X(w) ⟶ Z(w)))
                    ⟶ obd(Z)(λw. (Z(w) ∧ ¬X(w)) ∨ Y(w))" and
  axd_5e: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ obd(X)(Z) ∧ (∃w. Y(w) ∧ Z(w))) ⟶ obd(Y)(Z)" and

    (*for agent b: importers*)
  axb_3a: "∀w.∃x. avb(w)(x)" and axb_4a: "∀w x. avb(w)(x) ⟶ pvb(w)(x)" and axb_4ba: "∀w. pvb(w)(w)" and
  axb_5a: "∀X.¬obb(X)(λx. False)" and
  axb_5b: "∀X Y Z. (∀w. ((Y(w) ∧ X(w)) ⟷ (Z(w) ∧ X(w)))) ⟶ (obb(X)(Y) ⟷ obb(X)(Z))" and
  axb_5ca: "∀X β. ((∀Z. β(Z) ⟶ obb(X)(Z)) ∧ (∃Z. β(Z))) ⟶
      (((∃y. ((λw. ∀Z. (β Z) ⟶ (Z w))(y) ∧ X(y))) ⟶ obb(X)(λw. ∀Z. (β Z) ⟶ (Z w))))" and
  axb_5c: "∀X Y Z. (((∃w. (X(w) ∧ Y(w) ∧ Z(w))) ∧  obb(X)(Y)  ∧  obb(X)(Z))  ⟶  obb(X)(λw. Y(w) ∧ Z(w)))" and
  axb_5d: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ obb(X)(Y) ∧ (∀w. X(w) ⟶ Z(w)))
                    ⟶ obb(Z)(λw. (Z(w) ∧ ¬X(w)) ∨ Y(w))" and
  axb_5e: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ obb(X)(Z) ∧ (∃w. Y(w) ∧ Z(w))) ⟶ obb(Y)(Z)" and

  stit1: "∀a F w. ((stit a F) w) ⟶ F w"
```

Figure 13: Extended DDL First Variant Part 2

6.5 Extended DDL Second Variant

Finally, Figures 16, 17, 18, and 19 show the second variant of Extended DDL created by the first author [24].

Whereas the first variant introduced one set of accessibility relations for each additional agent, the second one works with a single, generalized set of accessibility relations, with the relations taking an agent as an input parameter (line 11). The axiomatization for this new set of accessibility relations can be found in lines 27-36. Using the new generalized accessibility relations, generalized obligation operators are defined in lines 54-60. They also take an agent as an input parameter. Abbreviations for the non-agentive and agentive accessibility relations are introduced in lines 83-85.

The design of the stit operator, including its axiomatization, is equivalent to the first variant.

For a more detailed explanation, please refer to [24].

```
abbreviation ddlneg::γ ("¬_"[52]53) where "¬A ≡ λw. ¬A(w)"
abbreviation ddland::ϱ (infixr"∧"51) where "A∧B ≡ λw. A(w)∧B(w)"
abbreviation ddlor::ϱ (infixr"∨"50) where "A∨B ≡ λw. A(w)∨B(w)"
abbreviation ddlimp::ϱ (infixr"→"49) where "A→B ≡ λw. A(w)⟶B(w)"
abbreviation ddlequiv::ϱ (infixr"↔"48) where "A↔B ≡ λw. A(w)⟷B(w)"

abbreviation ddlbox::γ ("□") where "□A ≡ λw.∀v. A(v)"  (*A = (λw. True)*)
abbreviation ddldia::γ  ("◇") where "◇ A ≡ ¬□(¬A)"

(*Necessity/possibility for agents*)
abbreviation ddlboxa_g::ζ ("□ₐ") where "□ₐ rel A ≡ λw. (∀x. (rel (w)(x) ⟶ A(x)))"  (*in all actual worlds*)
abbreviation ddlboxp_g::ζ ("□ₚ") where "□ₚ rel A ≡ λw. (∀x. (rel (w)(x) ⟶ A(x)))" (*in all potential worlds*)
abbreviation ddldia_g::ζ ("◇ₐ") where "◇ₐ rel A ≡ ¬□ₐ rel (¬A)"
abbreviation ddldiap_g::ζ ("◇ₚ") where "◇ₚ rel A ≡ ¬□ₚ rel (¬A)"

(*generalised obligation operators with relation as a parameter*)
abbreviation ddlo_g::ν ("O _ ⟨_|_⟩") where "O rel ⟨B|A⟩ ≡ λw. rel (A)(B)"  (*it ought to be A, given B *)
abbreviation ddloa_g::μ  ("Oₐ ") where "Oₐ rel1 rel2 A ≡ λw. rel1(rel2(w))(A) ∧ (∃x. rel2(w)(x) ∧ ¬A(x))" (*actual obligation*)
abbreviation ddlop_g::μ  ("Oₚ") where "Oₚ rel1 rel2 A ≡ λw. rel1(rel2(w))(A) ∧ (∃x. rel2(w)(x) ∧ ¬A(x))"  (*primary obligation*)

abbreviation ddltop::σ ("⊤") where "⊤ ≡ λw. True"
abbreviation ddlbot::σ ("⊥") where "⊥ ≡ λw. False"

(*Possibilist Quantification.*)
abbreviation ddlforall ("∀") where "∀Φ ≡ λw.∀x. (Φ x w)"
abbreviation ddlforallB (binder"∀"[8]9) where "∀x. φ(x) ≡ ∀φ"
abbreviation ddlexists ("∃") where "∃Φ ≡ λw.∃x. (Φ x w)"
abbreviation ddlexistsB (binder"∃"[8]9) where "∃x. φ(x) ≡ ∃φ"
```

Figure 14: Extended DDL First Variant Part 3

```
abbreviation ddlvalid::"σ ⇒ bool" ("⌊_⌋"[7]105) where "⌊A⌋ ≡ ∀w. A w"   (*Global validity*)
abbreviation ddlvalidcw::"σ ⇒ bool" ("⌊_⌋ₗ"[7]105) where "⌊A⌋ₗ ≡ A cw" (*Local validity (in cw)*)

(* A is obligatory *)
abbreviation ddlobl::γ ("○<_>") where "○<A> ≡  O ob ⟨A|⊤⟩"      (*New syntax: A is obligatory.*)
abbreviation ddlobld::γ ("○d<_>") where "○d<A> ≡  O obd ⟨A|⊤⟩"  (*New syntax: A is obligatory for agent d.*)
abbreviation ddloblb::γ ("○b<_>") where "○b<A> ≡  O obb ⟨A|⊤⟩"  (*New syntax: A is obligatory for agent b.*)

(* Consistency *)
lemma True nitpick [satisfy,user_axioms,show_all, card i = 2] oops
```

Figure 15: Extended DDL First Variant Part 4

```
theory DDL_agents_mod  (*DDL including STIT operator and agentive obligations*)
  imports
    Main
    types_2
begin

consts
 cw::i (*current world*)
 av::"i⇒σ" pv::"i⇒σ" ob::"σ⇒(σ⇒bool)" (*general accessibility relations*)

 av_g::"ag⇒i⇒σ" pv_g::"ag⇒i⇒σ" ob_g::"ag⇒(σ⇒(σ⇒bool))" (*agent-dependent accessibility relations*)

 (*stit operator*)
 stit::"ag⇒σ⇒σ" (*ag sees to it that*)

axiomatization where
 ax_3a: "∀w.∃x. av(w)(x)" and ax_4a: "∀w x. av(w)(x) ⟶ pv(w)(x)" and ax_4b: "∀w. pv(w)(w)" and
 ax_5a: "∀X.¬ob(X)(λx. False)" and
 ax_5b: "∀X Y Z. (∀w. ((Y(w) ∧ X(w)) ⟷ (Z(w) ∧ X(w)))) ⟶ (ob(X)(Y) ⟷ ob(X)(Z))" and
 ax_5ca: "∀X β. ((∀Z. β(Z) ⟶ ob(X)(Z)) ∧ (∃Z. β(Z))) ⟶
      (((∃y. ((λw. ∀Z. (β Z) ⟶ (Z w))(y) ∧ X(y))) ⟶ ob(X)(λw. ∀Z. (β Z) ⟶ (Z w))))" and
 ax_5c: "∀X Y Z. (((∃w. (X(w) ∧ Y(w) ∧ Z(w))) ∧  ob(X)(Y)  ∧  ob(X)(Z))  ⟶  ob(X)(λw. Y(w) ∧ Z(w)))" and
 ax_5d: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ ob(X)(Y) ∧ (∀w. X(w) ⟶ Z(w)))
                    ⟶ ob(Z)(λw. (Z(w) ∧ ¬X(w)) ∨ Y(w))" and
 ax_5e: "∀X Y Z. ((∀w. Y(w) ⟶ X(w)) ∧ ob(X)(Z) ∧ (∃w. Y(w) ∧ Z(w))) ⟶ ob(Y)(Z)" and
```

Figure 16: Extended DDL Second Variant Part 1

```
(*agent-dependent axioms*)
axg_3a: "∀w a.∃x. av_g a (w)(x)" and axg_4a: "∀w x a. av_g a (w)(x) ⟶ pv_g a (w)(x)" and axg_4ba: "∀w a. pv_g a (w)(w)" and
axg_5a: "∀X a.¬ob_g a (X)(λx. False)" and
axg_5b: "∀X Y Z a. (∀w. ((Y(w) ∧ X(w)) ⟷ (Z(w) ∧ X(w)))) ⟶ (ob_g a (X)(Y) ⟷ ob_g a (X)(Z))" and
axg_5ca: "∀X β a. ((∀Z. β(Z) ⟶ ob_g a (X)(Z)) ∧ (∃Z. β(Z))) ⟶
     (((∃y. ((λw. ∀Z. (β Z) ⟶ (Z w))(y) ∧ X(y))) ⟶ ob_g a (X)(λw. ∀Z. (β Z) ⟶ (Z w))))" and
axg_5c: "∀X Y Z a. (((∃w. (X(w) ∧ Y(w) ∧ Z(w))) ∧  ob_g a (X)(Y)  ∧  ob_g a(X)(Z))  ⟶  ob_g a (X)(λw. Y(w) ∧ Z(w)))" and
axg_5d: "∀X Y Z a. ((∀w. Y(w) ⟶ X(w)) ∧ ob_g a (X)(Y) ∧ (∀w. X(w) ⟶ Z(w)))
                  ⟶ ob_g a (Z)(λw. (Z(w) ∧ ¬X(w)) ∨ Y(w))" and
axg_5e: "∀X Y Z a. ((∀w. Y(w) ⟶ X(w)) ∧ ob_g a (X)(Z) ∧ (∃w. Y(w) ∧ Z(w))) ⟶ ob_g a (Y)(Z)" and

stit1: "∀a F w. ((stit a F) w) ⟶ F w"

abbreviation ddlneg::γ ("¬_"[52]53) where "¬A ≡ λw. ¬A(w)"
abbreviation ddland::ϱ (infixr"∧"51) where "A∧B ≡ λw. A(w)∧B(w)"
abbreviation ddlor::ϱ (infixr"∨"50) where "A∨B ≡ λw. A(w)∨B(w)"
abbreviation ddlimp::ϱ (infixr"→"49) where "A→B ≡ λw. A(w)⟶B(w)"
abbreviation ddlequiv::ϱ (infixr"↔"48) where "A↔B ≡ λw. A(w)⟷B(w)"
abbreviation ddlbox::γ ("□") where "□A ≡ λw.∀v. A(v)"
abbreviation ddldia::γ  ("◇") where "◇ A ≡ ¬□(¬A)"

(*(*Necessity/possibility for agents*)
abbreviation ddlboxa_g::η ("□ₐ") where "□ₐ i A ≡ λw. (∀x. av_g i (w)(x) ⟶ A(x))"  (*in all actual worlds*)
abbreviation ddlboxp_g::η ("□ₚ") where "□ₚ i A ≡ λw. (∀x. pv_g i (w)(x) ⟶ A(x))" (*in all potential worlds*)
abbreviation ddldiaa_g::η ("◇ₐ") where "◇ₐ i A ≡ ¬□ₐ i (¬A)"
abbreviation ddldiap_g::η ("◇ₚ") where "◇ₚ i A ≡ ¬□ₚ i (¬A)"*)
```

Figure 17: Extended DDL Second Variant Part 2

```
(*generalised operators with agents as a parameter*)
 abbreviation ddlo_g::χ ("O _ ⟨_|_⟩") where "O i ⟨B|A⟩ ≡
  λw. ob_g i A B"  (*Agent i ought to A, given B *)
 abbreviation ddloa_g::η  ("Oₐ _ ") where "Oₐ i A ≡
 λw. (ob_g i (av_g i (w))(A)) ∧ (∃x. av_g i (w)(x) ∧ ¬A(x))" (*actual obligation*)
 abbreviation ddlop_g::η  ("Oₚ _") where "Oₚ i A ≡
 λw. ob_g i (pv_g i (w))(A) ∧ (∃x. pv_g i (w)(x) ∧ ¬A(x))"  (*primary obligation*)

(*non-agentive necessity, possibility and obligation operators*)
abbreviation ddlboxa::γ ("□ₐ") where "□ₐA ≡ λw. (∀x. (av(w)(x) ⟶ A(x)))"  (*in all actual worlds*)
abbreviation ddlboxp::γ ("□ₚ") where "□ₚA ≡ λw. (∀x. pv(w)(x) ⟶ A(x))" (*in all potential worlds*)
abbreviation ddldiaa::γ ("◇ₐ") where "◇ₐA ≡ ¬□ₐ(¬A)"
abbreviation ddldiap::γ ("◇ₚ") where "◇ₚA ≡ ¬□ₚ(¬A)"
abbreviation ddlo::ϱ ("O⟨_|_⟩"[52]53) where "O⟨B|A⟩ ≡ λw. ob(A)(B)"  (*it ought to be ψ, given φ *)
abbreviation ddloa::γ  ("Oₐ") where "Oₐ A ≡ λw. ob(av(w))(A) ∧ (∃x. av(w)(x) ∧ ¬A(x))" (*actual obligation*)
abbreviation ddlop::γ  ("Oₚ") where "Oₚ A ≡ λw. ob(pv(w))(A) ∧ (∃x. pv(w)(x) ∧ ¬A(x))"  (*primary obligation*)

abbreviation ddltop::σ ("⊤") where "⊤ ≡ λw. True"
abbreviation ddlbot::σ ("⊥") where "⊥ ≡ λw. False"

(*Possibilist Quantification.*)
abbreviation ddlforall ("∀") where "∀Φ ≡ λw.∀x. (Φ x w)"
abbreviation ddlforallB (binder"∀"[8]9) where "∀x. φ(x) ≡ ∀φ"
abbreviation ddlexists ("∃") where "∃Φ ≡ λw.∃x. (Φ x w)"
abbreviation ddlexistsB (binder"∃"[8]9) where "∃x. φ(x) ≡ ∃φ"
```

Figure 18: Extended DDL Second Variant Part 3

Received 27 January 2025; revised 12 March 2025; accepted 5 June 2025

```
79
80 abbreviation ddlvalid::"σ ⇒ bool" ("⌊_⌋"[7]105) where "⌊A⌋ ≡ ∀w. A w"   (*Global validity*)
81 abbreviation ddlvalidcw::"σ ⇒ bool" ("⌊_⌋ı"[7]105) where "⌊A⌋ı ≡ A cw" (*Local validity (in cw)*)
82
83 (*New syntax *)
84 abbreviation ddlobl::γ ("○<_>") where "○<A> ≡  O⟨A|⊤⟩"
85 abbreviation ddlobl_g::η ("○_<_>") where "○ i <A> ≡  O i ⟨A|⊤⟩"
86
87 (* Consistency *)
88 lemma True nitpick [satisfy,user_axioms,show_all,card i=2, card ag=2, timeout=100] (*no model for i>1*) oops
89
90 end
```

Figure 19: Extended DDL Second Variant Part 4