Towards Efficient Reasoning with Intensional Concepts

Jesse Heyninck[1] , **Ricardo Gonçalves**[2] , **Matthias Knorr**[2] , **João Leite**[2]
[1]Technische Universität Dortmund
[2]NOVA LINCS, Departamento de Informátia, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
jesse.heyninck@tu-dortmund.de, {rjrg,mkn,jleite}@fct.unl.pt

Abstract

Recent developments triggered by initiatives such as the Semantic Web, Linked Open Data, the Web of Things, and geographic information systems resulted in the wide and increasing availability of machine-processable data and knowledge in the form of data streams and knowledge bases. Applications building on such knowledge require reasoning with modal and intensional concepts, such as time, space, and obligations, that are defeasible. For example, in the presence of data streams, conclusions may have to be revised due to newly arriving information. The current literature features a variety of domain-specific formalisms that allow for defeasible reasoning using specific intensional concepts. However, many of these formalisms are computationally intractable and limited to one of the mentioned application domains. In this paper, we define a general method for obtaining defeasible inferences over intensional concepts, and we study conditions under which these inferences are efficiently computable.

1 INTRODUCTION

In this paper, we develop a solution that allows us to efficiently reason with intensional concepts, such as time, space and obligations, providing defeasible/non-monotonic inferences in the presence of large quantities of data.

Initiatives such as the Semantic Web, Linked Open Data, and the Web of Things, as well as modern Geographic Information Systems, resulted in the wide and increasing availability of machine-processable data and knowledge in the form of data streams and knowledge bases. To truly take advantage of this kind of knowledge, it is paramount to be able to reason in the presence of *intensional* or *modal* concepts, which has resulted in an increased interest in formalisms, often based on rules with defeasible inferences, that allow for reasoning with time (Anicic et al. 2012; Gonçalves, Knorr, and Leite 2014; Brandt et al. 2018; Beck, Dao-Tran, and Eiter 2018; Brewka et al. 2018; Walega, Kaminski, and Grau 2019), space (Brenton, Faber, and Batsakis 2016; Walega, Schultz, and Bhatt 2017; Izmirlioglu and Erdem 2018; Suchan et al. 2018), and possibility or obligations (Panagiotidi, Nieves, and Vázquez-Salceda 2009; Gonçalves and Alferes 2012; Governatori, Rotolo, and Riveret 2018; Beirlaen, Heyninck, and Straßer 2019). Examples of such concepts may be found in applications with data referring for example to time (e.g., operators such as "next", "at time", "during interval T") or space (e.g., "at place P", "within a given radius", "connected to"), but also legal reasoning (e.g., "is obliged to", "is permitted").

Example 1. *Consider an airport that constantly receives data from sensors, cameras, etc. for monitorization, which, in combination with, e.g., facial recognition algorithms, allows one to automatize and optimize relevant tasks, such as boarding, checking in, and giving or denying access to certain parts of the airport. For example, checked-in passengers that are missing for boarding can be traced and alerted, and, in the case of non-compliance to proceed to the gate given the constraints on time and location, be subject to penalties. Also, passengers that comply with relevant security procedures can be automatically boarded, and irregularities can be communicated to the security personnel for possible intervention, allowing for a more efficient allocation of (human) resources, and a better-functioning and safer airport.*

In this context, efficient reasoning with non-monotonic rules over intensional concepts is indeed mandatory, since a) rules allow us to encode monitoring and intervention guidelines and policies in a user-friendly and declarative manner; b) conclusions may have to be revised in the presence of newly arriving information; c) different intensional concepts need to be incorporated in the reasoning process; and d) timely decisions are required, even in the presence of large amounts of data, as in streams. However, relevant existing work usually deals with only one kind of intensional concepts (as detailed before), and, in general, the computational complexity of the proposed formalisms is too high, usually due to both the adopted underlying formalism and the unrestricted reasoning with expressive intensional concepts.

In this paper, we introduce a formalism that allows us to reason with defeasible knowledge over intensional concepts. We build on so-called intensional logic programs (Orgun and Wadge 1992), extended with non-monotonic default negation, and equip them with a novel three-valued semantics with favorable properties. In particular, we define a well-founded model in the line of the well-founded semantics for logic programs (Gelder, Ross, and Schlipf 1991). Provided the adopted intensional operators satisfy certain properties, which turn out to be aligned with practical applications such as the one outlined in Ex. 1, the well-founded model is unique, minimal among the three-valued models,

in the sense of only providing derivable consequences, and, crucially, its computation is tractable. Our approach allows us to add to relevant related work in the sense of providing a well-founded semantics to formalisms that did not have one, which we illustrate on a relevant fragment of LARS programs (Beck, Dao-Tran, and Eiter 2018).

The remainder of the paper is structured as follows. We introduce intensional logic programs in Sec. 2, define our three-valued semantics in Sec. 3, show how to compute the well-founded model in Sec. 4, discuss the complexity and related work in Secs. 5 and 6, respectively, before we conclude.

2 INTENSIONAL LOGIC PROGRAMS

In this section, building on previous work by Orgun and Wadge [1992], we introduce intensional logic programs, a very expressive framework that allows us to reason with intensional concepts, such as time, space, and obligations, in the presence of large quantities of data, including streams of data. Such intensional logic programs are based on rules, as used in normal logic programs, enriched with atoms that introduce the desired intensional concepts. The usage of default negation in the rules is a distinctive feature compared to the original work (Orgun and Wadge 1992) and is particularly well-suited to model non-monotonic and defeasible reasoning (Gelfond 2008) and allows us to capture many other forms of non-monotonic reasoning, see, e.g., (Caminada et al. 2015; Chen et al. 2010).

To assign meaning to intensional programs, we rely on the framework of neighborhood semantics (Pacuit 2017), a generalization of the Kripke semantics, that easily allows us to capture a wide variety of intensional operators. In this section, we introduce neighborhood frames to assign semantics to intensional operators, and leave the definition of our novel three-valued semantics for such programs to the next section.

We start by defining the basic elements of our language. We consider a function-free first-order signature $\Sigma = \langle P, C\rangle$, a set X of variables, and a set of *operation symbols* $\mathcal{O}$, such that the sets P (of predicates), C (of constants), X and $\mathcal{O}$ are mutually disjoint. The set of atoms over Σ and X is defined in the usual way. We say that an atom is ground if it does not contain variables, and we denote by $\mathcal{A}_\Sigma$ the set of all ground atoms over Σ. In what follows, and without loss of generality, we leave the signature Σ implicit and consider only the set of ground atoms over Σ, denoted by $\mathcal{A}$.

The set $\mathcal{O}$ contains the symbols representing the various intensional operators ∇. Based on these, we introduce intensional atoms.

Definition 1. *Given a set of atoms $\mathcal{A}$ and a set of operation symbols $\mathcal{O}$, the set $\mathcal{I}^{\mathcal{A}}_{\mathcal{O}}$ of* intensional atoms over $\mathcal{A}$ and $\mathcal{O}$ *is defined as $\mathcal{I}^{\mathcal{A}}_{\mathcal{O}} = \{\nabla p \mid p \in \mathcal{A} \text{ and } \nabla \in \mathcal{O}\}$, and the set of* program atoms $\mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ *is defined as $\mathcal{L}^{\mathcal{A}}_{\mathcal{O}} = \mathcal{A} \cup \mathcal{I}^{\mathcal{A}}_{\mathcal{O}}$.*

We can define intensional logic programs as sets of rules with default negation, denoted by $\sim$, over program atoms.

Definition 2. *Given a set of atoms $\mathcal{A}$ and a set of operation symbols $\mathcal{O}$, an* intensional logic program $\mathcal{P}$ over $\mathcal{A}$ and $\mathcal{O}$ *is a finite set of rules of the form:*

$$A \leftarrow A_1, \ldots, A_n, \sim B_1, \ldots, \sim B_m \tag{1}$$

where $A, A_1, \ldots, A_n, B_1, \ldots, B_m \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$. We call A the head *of the rule, and $A_1, \ldots, A_n, \sim B_1, \ldots, \sim B_m$ its* body.

We also call $\mathcal{P}$ simply a *program* when this does not cause confusion and *positive* if it does not contain default negation. Intensional logic programs are highly expressive as intensional operators can appear arbitrarily anywhere in the rules, in particular in rule heads and in scope of default negation.

Example 2. *Consider a fragment of the setting in Ex. 1 with two gateways: a and b. The area before gateway a is α, the area between gateway a and b is β, and the area behind gateway b is γ. α and β are transit zones where one is not allowed to wait. For simplicity we assume a finite timeline $T = \{1, 2\}$.*[1] *Consider the set of operators $\mathcal{O}_1$:*

$$\mathcal{O}_1 = \{\mathsf{O}, \Box_t, @_{t,\ell}, @_\ell, \triangleleft_t \mid t \in T, \ell \in \{\alpha, \beta, \gamma\}\}$$

where O expresses that "something is obligatory", $\Box_t$ means "something is the case at time t and every place ℓ", $@_\ell$ means "something is the case at location ℓ", $@_{t,\ell}$ means "something is the case at time t and location ℓ", and $\triangleleft_t$ means "something is the case at or before time t". We use a signature $\langle P, C\rangle$ where the set C of constants contains identifiers representing persons, including p representing Petra, and the set P of predicates is composed of the following unary predicates: $\texttt{passed}_a$, $\texttt{passed}_b$ `move`, `is` *and* `called`*. They express that x passed through gate a or b, x moves, x is at a spatio-temporal point, and x is called, respectively.*

Consider program $\mathcal{P}$ composed of the following rules: [2]

$$\texttt{called}(x) \leftarrow \mathsf{O}\texttt{move}(x), \sim \texttt{move}(x) \tag{2}$$

$$\mathsf{O}\texttt{move}(x) \leftarrow \sim @_\gamma \texttt{is}(x) \tag{3}$$

$$\Box_t\texttt{move}(x) \leftarrow @_{t+1,\beta}\texttt{is}(x), @_{t,\alpha}\texttt{is}(x) \tag{4}$$

$$\Box_t\texttt{move}(x) \leftarrow @_{t+1,\gamma}\texttt{is}(x), @_{t,\beta}\texttt{is}(x) \tag{5}$$

$$@_{t,\beta}\texttt{is}(x) \leftarrow \triangleleft_t\texttt{passed}_a(x), \sim \triangleleft_t\texttt{passed}_b(x) \tag{6}$$

$$@_{t,\gamma}\texttt{is}(x) \leftarrow \triangleleft_t\texttt{passed}_b(x) \tag{7}$$

$$\Box_1\texttt{passed}_a(p) \leftarrow \tag{8}$$

Rule (2) encodes that if a person should move, but does not, she will be called. Rule (3) encodes that a person ought to move if she is not at γ. In the case of rules (4) and (5), a person moved if she was at two different locations at two subsequent time points. Rule (6) encodes that if a person passed through gate a, but not through gate b, she is at β, whereas rule (7) imposes that she is at γ if she passed through gate b. Finally, rule (8) asserts that Petra passed through gate a at time 1.

[1] Note that for applications such as the one described here, in practice, considering an arbitrarily large, but finite timeline does indeed suffice.

[2] In the course of this example, we use variables to ease the presentation. They represent the ground instantiation of such rules with all possible constants in the usual way. Time variable t also represents all possible values.

In order to give semantics to intensional operators, we follow the same ideas as employed by Orgun and Wadge [1992] and consider the neighborhood semantics, a strict generalization of Kripke-style semantics that allows capturing intensional operators (Pacuit 2017) such as temporal, spatial, or deontic operators, even those that do not satisfy the normality property imposed by Kripke frames (Chellas 1980).

We start by recalling neighborhood frames.

Definition 3. *Given a set of operation symbols $\mathcal{O}$, a* neighborhood frame *(over $\mathcal{O}$) is a pair $\mathfrak{F} = \langle W, N\rangle$ where W is a non-empty set (of worlds) and $N = \{\theta_\nabla \mid \nabla \in \mathcal{O}\}$ is a set of* neighborhood functions $\theta_\nabla : W \to \wp(\wp(W))$.[3]

Thus, in comparison to Kripke frames, instead of a relation over W, neighborhood frames have functions for each operator that map worlds to a set of sets of worlds. These sets intuitively represent the atoms necessary (according to the correspondent intensional operator) at that world.

Example 3. *The operators from Ex. 2 are given semantics using a neighborhood frame where the set of worlds W_1 is composed of triples $(t, \ell, \star)$ where $t \in T$ is a time point, $\ell \in \{\alpha, \beta, \gamma\}$ is a location and $\star \in \{\mathbb{I}, \mathbb{A}\}$ indicates if the world is the actual world $\mathbb{A}$ or the ideal world $\mathbb{I}$ (postulating ideal worlds is a standard technique for giving semantics to modal operators (McNamara 2019)). The neighborhoods of $\mathcal{O}_1$ are defined, for $t, t' \in T$, $\ell, \ell' \in \{\alpha, \beta, \gamma\}$, $w \in W_1$ and $\star \in \{\mathbb{I}, \mathbb{A}\}$, as:*

- $\theta_{\mathsf{O}}((t, \ell, \star)) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{I}) \in W'\}$;
- $\theta_{\Box_t}(w) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{A}) \in W'$ *for every* $\ell \in \{\alpha, \beta, \gamma\}\}$.
- $\theta_{@_\ell}((t, \ell', \star)) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{A}) \in W'\}$;
- $\theta_{@_{t,\ell}}(w) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{A}) \in W'\}$;
- $\theta_{\lhd_t}(w) = \{W' \subseteq W_1 \mid (t', \ell, \mathbb{A}) \in W'$, *for some* $t' \leq t$ *and for some* $\ell \in \{\alpha, \beta, \gamma\}\}$.

Intuitively, $\theta_{\mathsf{O}}((t, \ell, \star))$ consists of all the sets of worlds which include the ideal counterpart $(t, \ell, \mathbb{I})$ of $(t, \ell, \star)$; $\theta_{\Box_t}(w)$ consists of all the sets of worlds which includes all the actual worlds with time component t; $\theta_{\Box_t}(w)$ consists of all the sets of worlds that include all actual worlds with a time stamp t; $\theta_{@_\ell}(w)$ contains all sets of worlds that contains at least one actual world with a space component ℓ; a set of worlds is in $\theta_{@_{t,\ell}}(w)$ if it contains $(t, \ell, \mathbb{A})$; finally, a set of worlds is in $\theta_{\lhd_t}(w)$ if it contains at least one actual world with a time stamp t' which is earlier or equal as t.

Thus, neighborhood functions θ can be both invariant under the input w, i.e., $\theta(w) = \theta(w')$ for any $w, w' \in W$ (e.g., $\theta_{\Box_t}$ and $\theta_{@_{t,\ell}}$), or variate depending on w (e.g., θ_{O} and $@_\ell$). This is why the above definitions of neighborhood functions that depend on w need to explicit the components of the world w, i.e., $(t, \ell, \star)$.

[3]Note that we often leave $\mathcal{O}$ implicit as N allows to uniquely determine all elements from $\mathcal{O}$. Also, to ease the presentation, we only consider unary intensional operators. Others can then often be represented using rules (cf. also (Orgun and Wadge 1992)).

3 THREE-VALUED SEMANTICS

In this section, we define a three-valued semantics for intensional logic programs as an extension of the well-founded semantics for logic programs (Gelder, Ross, and Schlipf 1991) that incorporates reasoning over intensional concepts. The benefit of this approach over the more commonly used two-valued models is that, although there are usually several such three-valued models, we can determine a unique minimal one – intuitively the one which contains all the minimally necessary consequences of a program – which can be efficiently computed. In fact, even for programs without intensional concepts, a unique two-valued minimal model does not usually exist (Gelfond and Lifschitz 1991).

We consider three truth values, "true", "false", and "undefined", where the latter corresponds to neither true nor false. Given a neighborhood frame, we start by defining interpretations that contain a valuation function which indicates in which worlds (of the frame) an atom from $\mathcal{A}$ is true ($W^\top$), and in which ones it is true or undefined (W^u), i.e., not false [4].

Definition 4. *Given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F} = \langle W, N\rangle$, an* interpretation I over $\mathcal{A}$ and $\mathfrak{F}$ *is a tuple $\langle W, N, V\rangle$ with a valuation function $V : \mathcal{A} \to \wp(W) \times \wp(W)$ s.t., for every $p \in \mathcal{A}$, $V(p) = (W^\top, W^u)$ with $W^\top \subseteq W^u$. If, for every $p \in \mathcal{A}$, $W^\top = W^u$, then we call I* total.

The subset inclusion on the worlds ensures that no $p \in \mathcal{A}$ can be true and false in some world simultaneously. This intuition of the meaning is made precise with the denotation of program atoms for which we use the three truth values. We denote the truth values true, undefined and false with $\top$, u, and $\bot$, respectively, and we assume that the language $\mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ contains a special atom u (associated to u).

Definition 5. *Given a set of atoms $\mathcal{A}$, a frame $\mathfrak{F}$, and an interpretation $I = \langle W, N, V\rangle$, we define the* denotation of $A \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ in I*:*

- $\|p\|^\dagger_I = W^\dagger$ *if* $A = p \in \mathcal{A}$, *with* $V(p) = (W^\top, W^u)$ *and* $\dagger \in \{\top, u\}$;
- $\|\mathsf{u}\|^u = W$ *and* $\|\mathsf{u}\|^\top = \emptyset$, *if* $A = \mathsf{u}$;
- $\|\nabla p\|^\dagger_I = \{w \in W \mid \|p\|^\dagger_I \in \theta_\nabla(w)\}$ *if* $A = \nabla p \in \mathcal{I}^{\mathcal{A}}_{\mathcal{O}}$ *and* $\dagger \in \{\top, u\}$;
- $\|A\|^\bot_I = W \setminus \|A\|^u_I$ *for* $A \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$.

For a formula $A \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ and an interpretation I, $\|A\|^\top_I$ is the set of worlds in which A is true, $\|A\|^u_I$ is the set of worlds in which A is not false, i.e., undefined or true, and $\|A\|^\bot_I$ is the set of worlds in which A is false. For atoms $p \in \mathcal{A}$, the denotation is straightforwardly derived from the interpretation I, i.e., from the valuation function V, and for the special atom u it is defined as expected (undefined in all worlds). For an intensional atom ∇p, w is in the denotation $\|\nabla p\|^\dagger_I$ of ∇p if the denotation of p (according to I) is a neighborhood of ∇ for w, i.e. $\|p\|^\dagger_I \in \theta_\nabla(w)$.

[4]We follow the usual notation in modal logic and interpretations explicitly include the corresponding frame.

We often leave the subscript I from $\|A\|_I^\dagger$ as well as the reference to $\mathcal{A}$ and $\mathfrak{F}$ for interpretations and programs implicit.

Example 4. *Consider* $\langle W_1, \{\theta_{\mathsf{O}}, \theta_{@_1,\alpha}, \theta_{@_\beta}\}\rangle$ *as in Ex. 3 and* $I_1 = \langle W_1, \{\theta_{\mathsf{O}}, \theta_{@_1,\alpha}, \theta_{@_\beta}\}, V\rangle$ *where:*
$V(\mathtt{passed}_a(p)) = (\{(1,\alpha,\mathbb{A})\},\{(1,\alpha,\mathbb{A})\})$
$V(\mathtt{move}(p)) = (\{(1,\alpha,\mathbb{I})\},\{(1,\alpha,\mathbb{I}),(2,\beta,\mathbb{A})\})$
Then the following are examples of denotations of intensional atoms:

$\|\mathsf{O}\mathtt{move}(p)\|_{I_1}^\top = \{(1,\alpha,\mathbb{A}),(1,\alpha,\mathbb{I})\}$
$\|@_{1,\alpha}\mathtt{passed}_a(p)\|_{I_1}^\top = W_1$
$\|@_\beta\mathtt{move}(p)\|_{I_1}^{\mathsf{u}} = \{(2,\ell,\star) \mid \ell \in \{\alpha,\beta,\gamma\}, \star \in \{\mathbb{A},\mathbb{I}\}\}$

We explain the first denotation $\|\mathsf{O}\mathtt{move}(p)\|_{I_1}^\top$*: since* $\|\mathtt{move}(p)\| \ni (1,\alpha,\mathbb{I})$ *and* $\{(1,\alpha,\mathbb{I})\} \in \theta_{\mathsf{O}}((1,\alpha,\mathbb{I}))$ *and* $\{(1,\alpha,\mathbb{I})\} \in \theta_{\mathsf{O}}((1,\alpha,\mathbb{A}))$*, we get the denotation* $\|\mathsf{O}\mathtt{move}(p)\|_{I_1}^\top$ *as stated above.*

Based on the denotation, we can now define our model notion, which is inspired by partial stable models (Przymusinski 1991), which come with two favorable properties, minimality and support. The former captures the idea of minimal assumption, the latter provides traceable inferences from rules. We adapt this notion here by defining a reduct that, given an interpretation, transforms programs into positive ones, for which a satisfaction relation and a minimal model notion are defined.

We start by adapting two orders for interpretations, the truth ordering, $\sqsubseteq$, and the knowledge ordering, $\sqsubseteq_k$. The former prefers higher truth values in the order $\bot < u < \top$, the latter more knowledge (i.e., less undefined knowledge). Formally, for interpretations I and I', and every $p \in \mathcal{A}$:

- $I \sqsubseteq I'$ iff $\|p\|_I^\dagger \subseteq \|p\|_{I'}^\dagger$, for every $\dagger \in \{\top, u\}$;
- $I \sqsubseteq_k I'$ iff $\|p\|_I^\top \subseteq \|p\|_{I'}^\top$ and $\|p\|_I^\bot \subseteq \|p\|_{I'}^\bot$.

We write $I \prec I'$ if $I \preceq I'$ and $I' \not\preceq I$ for $\preceq \in \{\sqsubseteq, \sqsubseteq_k\}$.

We proceed with a generalization of the notion of reduct to programs with intensional atoms.

Definition 6. *Let* $\mathcal{A}$ *be set of atoms, and* $\mathfrak{F} = \langle W, N\rangle$ *a frame. The reduct of a program* $\mathcal{P}$ *at* $w \in W$ *w.r.t. an interpretation* I*,* $\mathcal{P}/I_w$*, contains for each* $r \in \mathcal{P}$ *of the form (1):*

- $A \leftarrow A_1, \ldots, A_n$ *if* $w \notin \bigcup_{i\le m} \|B_i\|^u$
- $A \leftarrow A_1, \ldots, A_n, \mathsf{u}$ *if* $w \in \bigcup_{i\le m} \|B_i\|^u \setminus \bigcup_{i\le m} \|B_i\|^\top$

Intuitively, for each rule r of $\mathcal{P}$, the reduct $\mathcal{P}/I_w$ contains either a rule of the first form, if all negated program atoms in the body of r are false at w (or the body does not have negated atoms), or a rule of the second form, if none of the negated program atoms in the body of r are true at w, but some of these are undefined at w, or none, otherwise. This also explains why the reduct is defined at w: truth and undefinedness vary for different worlds. The special atom u is applied to ensure that rules for the second case cannot impose the truth of the head in the notion of satisfaction for positive programs.

Note that the reduct of a program is a positive program, for which we can define a notion of satisfaction as follows.

Definition 7. *Let* $\mathcal{A}$ *be a set of atoms, and* $\mathfrak{F} = \langle W, N\rangle$ *a frame. An interpretation* I *satisfies a positive program* $\mathcal{P}$ *at* $w \in W$ *iff for each* $r \in \mathcal{P}$ *of the form (1), we have that* $w \in \bigcap_{i\le n} \|A_i\|^\dagger$ *implies* $w \in \|A\|^\dagger$ *(for any* $\dagger \in \{\top, \mathsf{u}\}$*)* [5].

Stable models can now be defined by imposing minimality w.r.t. the truth ordering on the corresponding reduct.

Definition 8. *Let* $\mathcal{A}$ *be set of atoms, and* $\mathfrak{F} = \langle W, N\rangle$ *a frame. An interpretation* I *is* a stable model *of a program* $\mathcal{P}$ *if:*

- *for every* $w \in W$*,* I *satisfies* $\mathcal{P}/I_w$ *at* w*, and*
- *there is no interpretation* I' *such that* $I' \sqsubset I$ *and, for each* $w \in W$*,* I' *satisfies* $\mathcal{P}/I_w$ *at* w*.*

Example 5. *Recall* $\mathcal{P}$ *from in Ex. 2. For simplicity of presentation suppose that the set of constants* C *only contains* p*, resulting in the following grounded program.* [6]

$$\begin{aligned}
\mathtt{called}(p) &\leftarrow \mathsf{O}\mathtt{move}(p), \sim \mathtt{move}(p)\\
\mathsf{O}\mathtt{move}(p) &\leftarrow \sim @_\gamma \mathtt{is}(p)\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\beta}\mathtt{is}(p), @_{t,\alpha}\mathtt{is}(p)\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\gamma}\mathtt{is}(p), @_{t,\beta}\mathtt{is}(p)\\
@_{t,\beta}\mathtt{is}(p) &\leftarrow \triangleleft_t\mathtt{passed}_a(p), \sim \triangleleft_t\mathtt{passed}_b(p)\\
@_{t,\gamma}\mathtt{is}(p) &\leftarrow \triangleleft_t\mathtt{passed}_b(p)\\
\Box_1\mathtt{passed}_a(p) &\leftarrow
\end{aligned}$$

Consider $\mathfrak{F} = \langle W_1, \mathcal{O}_1\rangle$ *as in Ex. 3 and the total interpretation* I_1 *defined by:*

$$\begin{aligned}
\|\mathtt{passed}_a(p)\|_{I_1}^\top &= \{(1,\ell,\mathbb{A}) \mid \ell \in \{\alpha,\beta,\gamma\}\}\\
\|\mathtt{passed}_b(p)\|_{I_1}^\top &= \{\}\\
\|\mathtt{is}(p)\|_{I_1}^\top &= \{(1,\beta,\mathbb{A})\}\\
\|\mathtt{move}(p)\|_{I_1}^\top &= \{(t,\ell,\mathbb{I}) \mid t \in T; \ell \in \{\alpha,\beta,\gamma\}\}\\
\|\mathtt{called}(p)\|_{I_1}^\top &= \{(t,\ell,\mathbb{A}) \mid t \in T; \ell \in \{\alpha,\beta,\gamma\}\}
\end{aligned}$$

We see that for any $(t',\ell,\mathbb{A}) \in W_1$*,* $\mathcal{P}/(I_1)_w$ *consists of the following rules occuring in* $\mathcal{P}$*.*

$$\begin{aligned}
\mathtt{called}(p) &\leftarrow \mathsf{O}\mathtt{move}(p)\\
\mathsf{O}\mathtt{move}(p) &\leftarrow\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\beta}\mathtt{is}(p), @_{t,\alpha}\mathtt{is}(p)\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\gamma}\mathtt{is}(p), @_{t,\beta}\mathtt{is}(p)\\
@_{t,\beta}\mathtt{is}(p) &\leftarrow \triangleleft_t\mathtt{passed}_a(p)\\
@_{t,\gamma}\mathtt{is}(p) &\leftarrow \triangleleft_t\mathtt{passed}_b(p)\\
\Box_1\mathtt{passed}_a(p) &\leftarrow
\end{aligned}$$

[5] Since the intersection of an empty sequence of subsets of a set is the entire set, then, for n=0, i.e., when the body of the rule is empty, the satisfaction condition is just $w \in \|A\|^\dagger$ for any $\dagger \in \{\top, \mathsf{u}\}$.

[6] To ease the presentation, we still use t to represent all possible values.

Whereas for any $(t', \ell, \mathbb{I}) \in W_1$, $\mathcal{P}/(I_1)_w$ consists of:

$$\begin{aligned}
\texttt{Omove}(p) &\leftarrow \\
\Box_t\texttt{move}(p) &\leftarrow @_{t+1,\beta}\texttt{is}(p), @_{t,\alpha}\texttt{is}(p) \\
\Box_t\texttt{move}(p) &\leftarrow @_{t+1,\gamma}\texttt{is}(p), @_{t,\beta}\texttt{is}(p) \\
@_{t,\beta}\texttt{is}(p) &\leftarrow \triangleleft_t\texttt{passed}_a(p) \\
@_{t,\gamma}\texttt{is}(p) &\leftarrow \triangleleft_t\texttt{passed}_b(p) \\
\Box_1\texttt{passed}_a(p) &\leftarrow
\end{aligned}$$

It can be checked that I_1 satisfies minimality and is therefore a stable model of $\mathcal{P}$.

Consider now the total interpetation I_2 identical to I_1 except for $\|\texttt{passed}_b(p)\|^{\top}_{I_2} = \{(2, \ell, \mathbb{A}) \mid \ell \in \{\alpha, \beta, \gamma\}\}$. Then for e.g. $(2, \alpha, \mathbb{A})$, the reduct $\mathcal{P}/(I_2)_{(2,\alpha,\mathbb{A})}$ is:

$$\begin{aligned}
\texttt{called}(p) &\leftarrow \texttt{Omove}(p) \\
\texttt{Omove}(p) &\leftarrow \\
\Box_t\texttt{move}(p) &\leftarrow @_{t+1,\beta}\texttt{is}(p), @_{t,\alpha}\texttt{is}(p) \\
\Box_t\texttt{move}(p) &\leftarrow @_{t+1,\gamma}\texttt{is}(p), @_{t,\beta}\texttt{is}(p) \\
@_{t,\gamma}\texttt{is}(p) &\leftarrow \triangleleft_t\texttt{passed}_b(p) \\
\Box_1\texttt{passed}_a(p) &\leftarrow
\end{aligned}$$

Since $(2, \alpha, \mathbb{A}) \notin \|@_{2,\gamma}\texttt{is}(p)\| = \emptyset$ even though $@_{2,\gamma}\texttt{is}(p) \leftarrow \triangleleft_2\texttt{passed}_b(p) \in \mathcal{P}/(I_2)_{(2,\alpha,\mathbb{A})}$ and $(2, \alpha, \mathbb{A}) \in \| \triangleleft_2 \texttt{passed}_b(p)\|^{\top}_{I_2} = W_1$, we see that I_2 does not satisfy $\mathcal{P}/(I_2)_{(2,\alpha,\mathbb{A})}$ and therefore is not a stable model.

We can show that our model notion is faithful to partial stable models of normal logic programs (Przymusinski 1991), i.e., if we consider a program without intensional atoms, then its semantics corresponds to that of partial stable models.

Proposition 1. *Let $\mathcal{A}$ be set of atoms, $\mathfrak{F}$ a frame, and $\mathcal{P}$ a program with no intensional atoms. Then, there is a one-to-one correspondence between the stable models of $\mathcal{P}$ and the partial stable models of the normal logic program $\mathcal{P}$.*

While partial stable models are indeed truth-minimal, this turns out not to be the case for intensional programs, due to non-monotonic intensional operators.

Example 6. *Consider the operator $|j, k|^{\gamma}$ representing that an atom is true at γ at all time points in $[j, k]$, and not in any interval properly containing $[j, k]$. This operator has the following neighborhood (given W_1 from Ex. 3): $\theta_{|j,k|}(w) = \{W' \subseteq W_1 \mid \{(j, \gamma, \mathbb{A}), (j + 1, \gamma, \mathbb{A}), \ldots, (k, \gamma, \mathbb{A})\} \subseteq W'$ and $(j - 1, \gamma, \mathbb{A}), (k + 1, \gamma, \mathbb{A}) \notin W'\}$. Consider the following program $\mathcal{P}$ consisting of:*

$$\begin{aligned}
@_{1,\gamma}\texttt{is}(p) &\leftarrow \\
@_{2,\gamma}\texttt{is}(p) &\leftarrow \\
@_{3,\gamma}\texttt{is}(p) &\leftarrow\sim |1, 2|^{\gamma}\texttt{is}(p)
\end{aligned}$$

This program has two stable models, of which one is not minimal. In more detail, the following interpretations are stable: I_1 with $\|\mathsf{is}(p)\|^{\top}_{I_1} = \|\mathsf{is}(p)\|^{\mathsf{u}}_{I_1} = \{(1, \gamma, \mathbb{A}), (2, \gamma, \mathbb{A})\}$ and I_2 with $\|\mathsf{is}(p)\|^{\top}_{I_2} = \|\mathsf{is}(p)\|^{\mathsf{u}}_{I_2} = \{(1, \gamma, \mathbb{A}), (2, \gamma, \mathbb{A}), (3, \gamma, \mathbb{A})\}$. To see that I_2 is stable, observe first that since $\{(1, \gamma, \mathbb{A}), (2, \gamma, \mathbb{A}), (3, \gamma, \mathbb{A})\} \notin \theta_{|1,2|^{\gamma}}(w)$ for any $w \in W_1$, $\||1, 2|^{\gamma}\texttt{is}(p)\|^{\top}_{I_2} = \emptyset$, which means that $\mathcal{P}/I_2 = \{@_{1,\gamma}\texttt{is}(p) \leftarrow; @_{2,\gamma}\texttt{is}(p) \leftarrow; @_{3,\gamma}\texttt{is}(p) \leftarrow\}$. Clearly, I_2 is the $\sqsubset$-minimal interpretation that satisfies $\mathcal{P}/I_2$. However, $I_1 \sqsubset I_2$ and thus, I_2 is not a truth-minimal stable model.

To counter that, we consider monotonic operators. Formally, given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F}$, an intensional operator ∇ is said to be *monotonic in* $\mathfrak{F}$ if, for any two interpretations I and I' such that $I \sqsubseteq I'$, we have that $\|\nabla p\|^{\dagger}_I \subseteq \|\nabla p\|^{\dagger}_{I'}$ for every $p \in \mathcal{A}$ and $\dagger \in \{\top, u\}$.

If all intensional operators in a frame are monotonic, then truth-minimality of stable models is guaranteed.

Proposition 2. *Let $\mathcal{A}$ be set of atoms, and $\mathfrak{F}$ a frame in which all intensional operators are monotonic. If I is a stable model of $\mathcal{P}$, then there is no stable model I' of $\mathcal{P}$ such that $I' \sqsubset I$.*

Regarding support, recall that the stable models semantics of normal logic programs satisfies the support property, in the sense that for every atom of a stable model there is a rule that justifies it. In other words, if we remove an atom p from a stable model some rule becomes false in the resulting model. Such rule can be seen as a justification for p being true at the stable model. In the case of intensional logic programs we say that an interpretation $I = \langle W, N, V\rangle$ is supported for a program $\mathcal{P}$ if, for every $p \in \mathcal{A}$ and $w \in W$, if $w \in \|p\|^{\top}$, then there is a rule $r \in \mathcal{P}/I_w$ that is not satisfied by I' at w, where $I' = \langle W, N, V'\rangle$ is such that $V'(q) = V(q)$ for $q \neq p$, and $V'(p) = \langle W^{\top} \setminus \{w\}, W^u\rangle$ where $V(p) = \langle W^{\top}, W^u\rangle$.

This notion of supportedness is desirable for intensional logic programs since we also want a justification why each atom is true at each world in a stable model. The following results show that this is indeed the case.

Proposition 3. *Let $\mathcal{A}$ be set of atoms, and $\mathfrak{F}$ a frame. Then, every stable model of a program $\mathcal{P}$ is supported.*

In general, the existence and uniqueness of stable models of a program is not guaranteed, not even for positive programs and/or under the restriction of all operators being monotonic.

Example 7. *Let $\mathcal{O} = \{\oplus\}$, $\mathcal{A} = \{p\}$ and $\mathfrak{F} = \langle\{1, 2\}, \{\theta_{\oplus}\}\rangle$ where $\theta_{\oplus}(1) = \theta_{\oplus}(2) = \{\{1\}, \{2\}\}$. Let $\mathcal{P} = \{\oplus p \leftarrow\}$. This program has two stable models:*

- *I_1 with $V_1(p) = (\{1\}, \{1\})$;*
- *I_2 with $V_2(p) = (\{2\}, \{2\})$.*

The existence of two stable models of the above positive program is caused by the non-determinism introduced by the intensional operator in the head of the rule. Formally, an operator θ of a frame $\mathfrak{F} = \langle W, N\rangle$ is *deterministic* if $\bigcap \theta(w) \in \theta(w)$ for every $w \in W$. A program $\mathcal{P}$ is *deterministic in the head* if, for every rule $r \in \mathcal{P}$ of the form (1), if $A = \nabla p$, then θ_{∇} is deterministic.

We can show that every positive program that is deterministic in the head and only considers monotonic operators has a single minimal model.

Proposition 4. *Given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F}$, if $\mathcal{P}$ is a positive program that is deterministic in the head and*

every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$, then it has a unique stable model.

Due to this result, in what follows, we focus on monotonic operators and programs that are deterministic in the head, as this is important for several of the results we obtain subsequently. This does not mean that non-montonic intensional operators cannot be used in our framework. In fact, we can take advantage of the default negation operator $\sim$ to *define* non-monotonic formulas on the basis of monotonic operators and default negation. As an example, consider again the operator $|j,k|$ from example 6. We can use the following rule to define $|j,k|p$ for some atom $p \in \mathcal{A}$: $|j,k|p \leftarrow @_j p, @_{j+1} p, \ldots, @_{k-1} p, @_k p, \sim @_{j-1} p, \sim @_{k+1} p$.

Among the stable models of a program, we can distinguish the *well-founded models* as those that are minimal in terms of the knowledge order.

Definition 9. *Given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F}$, an interpretation $I = \langle W, N, V \rangle$ is a* well-founded model of a program $\mathcal{P}$ *if it is a stable model of $\mathcal{P}$, and, for every stable model I' of $\mathcal{P}$, it holds that $I \sqsubseteq_k I'$.*

Example 8 (Example 5 continued)**.** *Since I_2 is in fact the unique stable model, it is therefore the well-founded model.*

Given our assumptions about monotonicity and determinism in the head, we can also show that the well-founded model of an intensional program exists and is unique.

Theorem 1. *Given a set of atoms $\mathcal{A}$, and a frame $\mathfrak{F}$, every program $\mathcal{P}$ has a unique well-founded model.*

4 ALTERNATING FIXPOINT

In this section, we show how the well-founded model can be efficiently computed. Essentially, we extend the idea of the alternating fixpoint developed for logic programs (Gelder 1989), that builds on computing, in an alternating manner, underestimates of what is necessarily true, and overestimates of what is not false, with the mechanisms to handle intensional inferences. Namely, we first define an operator for inferring consequences from positive programs, and make it applicable in general using the reduct. Based on an iterative application of these, the alternating fixpoint provides the well-founded model.

First, since different pieces of knowledge are inferable in different worlds, we need a way to distinguish between these. Therefore, we introduce labels referring to worlds and apply them to formulas of a given language as well as programs.

Given a language $\mathcal{L}$, a frame $\mathfrak{F} = \langle W, N \rangle$, and a program $\mathcal{P}$, we define the *language labelled by* W, $\mathcal{L}_W$, as $\{w : A \mid w \in W \text{ and } A \in \mathcal{L}\}$, and the *program labelled by* W, $\mathcal{P}_W$, as $\{w{:}r \mid r \in \mathcal{P}, w \in W\}$. $\mathcal{P}_W$ is *positive* iff $\mathcal{P}$ is. This allows us to say that some (program) atom is true (or undefined) at world w, which can, e.g., be used for inferences using rules labelled with w. This way, we can also compute inferences for several worlds simultaneously, since the labels allow us to avoid any potential overlaps.

We now proceed to define an operator for positive labelled programs for computing inferences given a set of labelled atoms. This operator is composed of three operators to incorporate reasoning over rules as well as over intensional atoms.

We first define an *immediate consequence operator* applied to sets of labelled program atoms. To ease notation, here and in the following, we use $\mathcal{L}_W$ to represent $(\mathcal{L}^{\mathcal{A}}_{\mathcal{O}})_W$.

Definition 10. *Given a frame $\mathfrak{F}$, a set of $\mathcal{L}_W$-formulas Δ, and a positive program $\mathcal{P}_W$, we define $T_{\mathcal{P}_W}(\Delta)$ as follows:*

$$T_{\mathcal{P}_W}(\Delta) = \{w{:}A \mid w{:}A \leftarrow A_1, \ldots A_n \in \mathcal{P}_W, \\ w{:}A_1, \ldots, w{:}A_n \in \Delta\}$$

Example 9. *Let $\mathcal{P}_W = \{(2, \alpha, \mathbb{A}) : @_1 \texttt{passed}_a(p) \leftarrow\}$. Then $T_{\mathcal{P}_W}(\emptyset) = \{(2, \alpha, \mathbb{A}){:}@_1 \texttt{passed}_a(p)\}$.*

The result of $T_{\mathcal{P}_W}$ may contain labelled intensional atoms, such as in Ex. 9, which implies that $\texttt{passed}_a(p)$ holds at $(1, \alpha, \mathbb{A})$.

The next operator, the *intensional extraction operator* IE_∇ allows us to derive such labelled atoms from labelled intensional atoms.

Definition 11. *Given a frame $\mathfrak{F}$, a set of $\mathcal{L}_W$-formulas Δ, and $\nabla \in \mathcal{O}$, we define $IE_\nabla(\Delta)$ as follows:*

$$IE_\nabla(\Delta) = \{w{:}A \mid w'{:}\nabla A \in \Delta, w \in \bigcap \theta_\nabla(w')\}$$

As IE_∇ is intended to be applied to results of $T_{\mathcal{P}_W}$, and these only contain intensional atoms occurring in the head of some rule in a given program $\mathcal{P}$, we restrict IE_∇ to this set of intensional operators, which we denote by $\mathcal{O}^{\mathcal{P}}$. Since $\mathcal{P}$ is deterministic in the head, this also ensures that IE_∇ has a unique outcome. Also, since programs do not contain nested operators, we can consider the union of IE_∇ for all $\nabla \in \mathcal{O}^{\mathcal{P}}$.

Finally, the *intensional consequence operator* IC_∇ maps atoms to intensional atoms that are implied by the former, i.e., it maps $w_1{:}A, \ldots, w_n{:}A$ to $w{:}\nabla A$ if $\{w_1, \ldots, w_n\} \in \theta_\nabla(w)$.

Definition 12. *Given a frame $\mathfrak{F} = \langle W, N \rangle$, a set of $\mathcal{L}_W$-formulas Δ, and $\theta_\nabla \in N$, we define $IC_\nabla(\Delta)$ as follows:*

$$IC_\nabla(\Delta) = \{w : \nabla A \in \mathcal{L}_W \mid \{w' \mid w'{:}A \in \Delta\} \in \theta_\nabla(w), \\ w \in W\}$$

Here, we can also simply consider the union for all $\theta_\nabla \in N$.

Example 10. *Consider the frame $\mathfrak{F} = \langle W_1, \{\theta_{@_1}, \theta_{\triangleleft_1}\}\rangle$ as defined in Ex. 3. Let $\Delta = \{(2, \alpha, \mathbb{A}) : @_1 \texttt{passed}_a(p)\}$. Since $(2, \alpha, \mathbb{A}) : @_1 \texttt{passed}_a(p) \in \Delta$ and $(1, l, \mathbb{A}) \in \bigcap \theta_{@_1}((2, \alpha, \mathbb{A}))$, $IE_{@_1}(\Delta) = \{(1, \ell, \mathbb{A}) : \texttt{passed}_a(p) \mid \ell \in \{\alpha, \beta, \gamma\}\}$. Informally, to believe Δ, and thus to believe that Petra passed gate a at time 1, $\texttt{passed}_a(p)$ has to be true at every actual world with time stamp 1.*

Also, $IC_{\triangleleft_1}(IE_{@_1}(\Delta)) = \{w{:}\triangleleft_1 \texttt{passed}_a(p) \mid w \in W_1\}$, since $\{(1, \ell, \mathbb{A}) \mid \ell \in \{\alpha, \beta, \gamma\}\} \in \theta_{\triangleleft_1}(w)$ for any $w \in W_1$ and $(1, \ell, \mathbb{A}) : \texttt{passed}_a(p) \in IE_{@_1}(\Delta)$ for any $\ell \in \{\alpha, \beta, \gamma\}$. Informally, since we know that $\texttt{passed}_a(p)$ is the case at time 1, we derive that $\texttt{passed}_a(p)$ is the case at or before time 1 (formally: $\triangleleft_1 \texttt{passed}_a(p)$).

We are now ready to define the closure operator as the least fixpoint of the composition of $T_{\mathcal{P}_W}$, IE_∇ and IC_∇.

Definition 13. *Given a frame $\mathfrak{F} = \langle W, N\rangle$ and a positive program $\mathcal{P}_W$, $Cn(\mathcal{P}_W)$ is defined as the least fixpoint* [7] *of*

$$\bigcup_{\nabla\in\mathcal{O}} IC_\nabla\left(\bigcup_{\nabla\in\mathcal{O}^\mathcal{P}} IE_\nabla(T_{\mathcal{P}_W})\right).$$

The consequence operator defined above is adequate in the sense that it calculates the minimal (total) model of a positive program. In more detail, we define an interpretation $I(Cn)$: $W_p = \{w \in W \mid w : p \in Cn(\mathcal{P}_W)\}$, $I(Cn) = \langle W, N, V\rangle$ is a minimal model with $V(p) = (W_p, W_p)$ for every $p \in \mathcal{A}$.

Proposition 5. *Given a frame $\mathfrak{F} = \langle W, N\rangle$, and a positive program $\mathcal{P}$, if every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$ and $\mathcal{P}$ is deterministic in the head, then $I(Cn)$ is the unique (and total) stable model of $\mathcal{P}$.*

This result can be generalized to arbitary programs relying on the reduct and on the alternating fixpoint (Gelder 1989). For the former, we adapt the reduct from Def. 6 to labelled languages, with the benefit that we can define a single reduct for all worlds $w \in W$.

Given a frame $\mathfrak{F} = \langle W, N\rangle$, a set of $\mathcal{L}_W$-formulas Δ and a program $\mathcal{P}_W$, *the reduct*, $\mathcal{P}_W/\Delta = \{w : A \leftarrow A_1, \ldots, A_n \mid r \in \mathcal{P}$ of the form (1) and, $\forall i \leqslant m, w : B_i \notin \Delta\}$.

The idea of the alternating fixpoint can be summarized as follows. We create two sequences, that are meant to represent an underestimation of what is true (P^i) and an overestimation of what is not false (N^i). Each iteration is meant to further increase the elements in P^i and further decrease the elements in N^i using Cn over reducts obtained from the labelled program and the results from the previous iteration.

Given a frame $\mathfrak{F} = \langle W, N\rangle$ and a program $\mathcal{P}$, we define:

$$P^0 = \emptyset \qquad N^0 = \mathcal{L}_W$$
$$P^{i+1} = Cn(\mathcal{P}_W/N^i) \qquad N^{i+1} = Cn(\mathcal{P}_W/P^i)$$
$$P^\omega = \bigcup_i P^i \qquad N^\omega = \bigcap_i N^i$$

We can show that P^i is increasing, and that N^i is decreasing, and both sequences reach a fixpoint because the operator for determining Cn is monotonic and the reduct is antitonic.

Proposition 6. *Given a set of atoms $\mathcal{A}$, a frame $\mathfrak{F} = \langle W, N\rangle$, and a positive program $\mathcal{P}$, if every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$ and $\mathcal{P}$ is deterministic in the head, then there are $i, j \in \mathbb{N}$ s.t. $P^i = P^{i+1}$ and $N^j = N^{j+1}$.*

Example 11. *Consider the frame $\mathfrak{F} = \langle W_1, \{\theta_{@_1}, \theta_{@_{1,\beta}}\}\rangle$ as defined in Ex. 3. Let $\mathcal{P}_W = \{w : @_1\texttt{passed}_a(p) \leftarrow; w : @_{1,\beta}\texttt{is}(p) \leftarrow \texttt{passed}_a(p), \sim \texttt{passed}_b(p) \mid w \in W_1\}$. The alternating fixpoint construction is carried out as follows: We start with $P^0 = \emptyset$ and $N^0 = \mathcal{L}_W$. Then $\mathcal{P}_W/N^0 = \{w : @_1\texttt{passed}_a(p) \leftarrow\mid w \in W_1\}$ and $\mathcal{P}_W/P^0 = \{w : @_1\texttt{passed}_a(p) \leftarrow; w : @_{1,\beta}\texttt{is}(p) \leftarrow \texttt{passed}_a(p) \mid w \in W_1\}$, which implies $P^1 = \{w : @_1\texttt{passed}_a(p); (1, \ell, \mathbb{A}) : \texttt{passed}_a(p) \mid w \in W_1, \ell \in \{\alpha, \beta, \gamma\}\}$ and $N^1 = P^1 \cup \{(1, \ell, \mathbb{A}) : @_{1,\beta}\texttt{is}(p) \mid \ell \in \{\alpha, \beta, \gamma\}\}$. From this we derive $\mathcal{P}_W/P^1 = \mathcal{P}_W/N^1 = \{w : @_1\texttt{passed}_a(p) \leftarrow; w : @_{1,\beta}\texttt{is}(p) \leftarrow \texttt{passed}_a(p) \mid w \in W_1\}$, which allows us to calculate $P^2 = N^2 = N^1$. Notice that a fixpoint is reached and thus $P^\omega = N^\omega = N^1$.*

Given a frame $\mathfrak{F}$, for which any $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$, the alternating fixpoint construction defined above offers a characterization of the well-founded model for programs that are deterministic in the head. In more detail, given a pair $\langle\Delta, \Theta\rangle$ of sets of $\mathcal{L}_W$-formulas, we define a partial interpretation $I(\langle\Delta, \Theta\rangle) = (W, N, V)$ on the basis of Δ as follows: for every $A \in \mathcal{A}$, $V(A) = (\{w \in W \mid w : A \in \Delta\}, \{w \in W \mid w : A \in \Theta\})$. We can then show this correspondence.

Theorem 2. *Given a frame $\mathfrak{F} = \langle W, N\rangle$, and a program $\mathcal{P}$ s.t. every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$ and $\mathcal{P}$ is deterministic in the head, then $I(\langle P^\omega, N^\omega\rangle)$ is the well-founded model of $\mathcal{P}$.*

Thus the result of the alternating fixpoint operator is a precise representation of the well-founded model of the considered intensional program.

5 COMPUTATIONAL COMPLEXITY

In this section, we study the computational complexity of several of the problems considered. We recall that the problem of satisfiability under neighborhood semantics has been studied for a variety of epistemic structures (Vardi 1989). Here, we consider the problem of determining models for the two notions we established, stable models and the well-founded model, and we focus on the propositional case.[8]

We assume familiarity with standard complexity concepts, including oracles and the polynomial hierarchy.

We first provide a result in the spirit of model-checking for programs $\mathcal{P}$. As we do not impose any semantic properties on the neighborhood frames we consider, determining a model for a frame that can be arbitrarily chosen is not meaningful. Thus, in the remainder, we assume a fixed frame $\mathfrak{F}$, fixing the worlds and the semantics of the intensional operators.[9]

Proposition 7. *Given a program $\mathcal{P}$ and an interpretation I, deciding whether I is a stable model of $\mathcal{P}$ is in* coNP.

This result is due to the minimization of stable models, i.e., we need to check for satisfaction and verify that there is no other interpretation which is smaller (cf. Def. 8). This also impacts on the complexity of finding a stable model given a fixed frame.

Theorem 3. *Given a program $\mathcal{P}$, deciding whether there is a stable model of $\mathcal{P}$ is in Σ_2^P.*

[7] Recall that, given an operator T over lattice $(L, \leqslant)$ and an ordinal α, $T \uparrow \alpha$ is defined as: $T \uparrow 0 = \emptyset$, $T \uparrow \alpha = T(T \uparrow \alpha - 1)$ for successor ordinals, and $T \uparrow \alpha = \bigcup_\alpha T \uparrow \alpha$ for limit ordinals.

[8] Corresponding results for the data complexity of this problem for programs with variables can then be achieved in the usual way (Dantsin et al. 2001).

[9] This also aligns well with related work, e.g., for reasoning with time, such as stream reasoning where often a finite timeline is assumed, and avoids the exponential explosion on the number of worlds for satisfiability for some epistemic structures (Vardi 1989).

Note that these results do not require that intensional operators be monotonic or deterministic in the head. In fact, if intensional operators are monotonic, we obtain the following improved results on Prop. 7 and Thm. 3 from Prop. 2.

Corollary 1. *Given a program $\mathcal{P}$ such that all operators ocurring in $\mathcal{P}$ are monotonic, and an interpretation I, deciding whether I is a stable model of $\mathcal{P}$ is in* P.

Corollary 2. *Given a program $\mathcal{P}$, deciding whether there is a stable model of $\mathcal{P}$ is in* NP.

Thus, if all operators are monotonic the complexity results do coincide with that of normal logic programs (without intensional atoms) (Dantsin et al. 2001), which indicates that monotonic operators do not add an additional burden in terms of computational complexity.

Now, if we in addition consider programs that are deterministic in the head, then we know that there exists the unique well-founded model (cf. Thm. 1). As we have shown, this model can be computed efficiently (cf. Thm. 2), and we obtain the following result in terms of computational complexity.

Theorem 4. *Given a program $\mathcal{P}$ that is deterministic in the head and all operators occurring in $\mathcal{P}$ are monotonic, computing the well-founded model of $\mathcal{P}$ is* P*-complete.*

Note that this result is indeed crucial in contexts were reasoning with a variety of intensional concepts needs to be highly efficient.

6 RELATED WORK

In this section, we discuss related work establishing relations to relevant formalisms in the literature.

Intensional logic programs were first defined by Orgun and Wadge [1992] focussing on the existence of models in function of the properties of the intensional operators. Only positive programs are considered, but nesting of intensional operators is allowed. The latter however can be covered in our approach by introducing corresponding additional operators that represent each nesting occurring in such a program. This allows us to show that our approach covers the previous work.

Proposition 8. *Let $\mathcal{P}$ be program as in (Orgun and Wadge 1992). Then there is a positive intensional program $\mathcal{P}'$ such that there is a one-to-one correspondence between the models of $\mathcal{P}$ and the total stable models of $\mathcal{P}'$.*

The contrary is not true already for programs without intensional operators. We could use a non-monotonic intensional operator for representing default negation, but these are not considered in (Orgun and Wadge 1992) confirming that our work is indeed an extension of the previous approach.

Since (Orgun and Wadge 1992) covers classical approaches for intensional reasoning, such as TempLog (Abadi and Manna 1989) and MoLog (del Cerro 1986), our work applies to these as well.

It also relates to more recent work with intensional operators, and we first discuss two prominent approaches in the area of stream reasoning.

LARS (Beck, Dao-Tran, and Eiter 2018) assumes a set of atoms $\mathcal{A}$ and a stream $S = (T, v)$, where T is a closed interval of the natural numbers and v is an evaluation function that defines which atoms are true at each time point of T. Several temporal operators are defined, including expressive window operators, and answer streams, a generalization of FLP-semantics, are employed for reasoning. A number of related approaches are covered including CQL (Arasu, Babu, and Widom 2006), C-SPARQL (Barbieri et al. 2010), and CQELS (Phuoc et al. 2011). Among the implementations exists LASER (Bazoobandi, Beck, and Urbani 2017), which focuses on a considerable fragment, called plain LARS.

We can represent a plain LARS program $\mathcal{P}$ for stream S as a program $\mathcal{P}_S$, encoding S using the $@_t$ operator. This allows us to show the following result relating the answer streams of plain LARS to the total stable models of such a program $\mathcal{P}_S$.

Proposition 9. *Given a plain LARS program $\mathcal{P}$ for stream S and $\mathcal{P}_S$, there is a one-to-one correspondence between answer streams of $\mathcal{P}$ for S and total stable models of $\mathcal{P}_S$.*

In addition, for such an encoding of plain LARS programs into intensional programs, we can apply our well-founded semantics, since the operators applied in plain LARS are monotonic and deterministic. Hence, our work also provides a well-founded semantics for plain LARS, i.e., we allow the usage of unrestricted default negation while preserving polynomial reasoning.

ETALIS (Anicic et al. 2012) aims at *complex event processing*. It assumes as input atomic events with a time stamp and uses *complex events*, based on Allen's interval algebra (Allen 1990), that are associated with a time *interval*, and is therefore considerably different from LARS (which considers time points). It contains no negation in the traditional sense, but allows for a negated pattern among the events. Many of the complex event patterns from ETALIS can be captured as neighborhood functions in our framework. However, ETALIS also makes use of some event patterns that would result in a non-monotonic operator, such as the negated pattern $\texttt{not}(p)[q, r]$ which expresses that p is not the case in the interval between the end time of q and the starting time of r. We conjecture that such a negation can be modelled with a combination of the default negation $\sim$ and an operator $[q, r]p$ which expresses that p is the case in the interval between the end time of q and the starting time of r, which in turn can to be defined using rules such as: $[q, r]p \leftarrow [t, t']p, @_t q, @_{t'} r, \sim @_{t+1} q, \sim @_{t'-1} r$. Defining a transformation that converts a set of ETALIS rules into an intensional logic program is left for future work.

Deontic logic programs of (Gonçalves and Alferes 2012) are similar in spirit to our work as they extend logic programs with deontic logic formulas under stable model semantics. Although complex deontic formulas can appear in the rules, the deontic operators are restricted to those of Standard Deontic Logic (SDL), and computational aspects are not considered.

Answer Set Programming Modulo Theories extended to the Qualitative Spatial Domain (in short, ASPMT(QS))

(Walega, Schultz, and Bhatt 2017) allows for the systematic modelling of dynamic spatial systems. It is based on logic programs over first-order formulas (with function symbols), which are not yet integrated in our approach. On the other hand, this work does only conside spatial reasoning. An interesting option for future work would be considering an extension incorporating such formulas.

7 CONCLUSIONS

Building on work by Orgun and Wadge (1992), we have introduced intensional logic programs that allow defeasible reasoning with intensional concepts, such as time, space, and obligations, and with streams of data. Relying on the neighborhood semantics (Pacuit 2017), we have introduced a novel three-valued semantics based on ideas adapted from partial stable models (Przymusinski 1991). Due to the expressivity of the intensional operators, stable models may not be minimal nor deterministic even for programs without default negation. Hence, we have studied the characteristics of our semantics for monotonic intensional operators and programs that only admit deterministic operators in the heads of the rules, and shown that a unique minimal model, the well-founded model, exists and can be computed with an alternating fixpoint construction. We have studied the computational complexity of checking for existence of models and computation of models and established that the well-founded model can be computed in polynomial time. Finally, we have discussed related work and shown that several relevant approaches in the literature can be covered.

In terms of future work, we want to investigate in more detail the exact relations to existing approaches in the literature, that are not formally covered in this paper. Furthermore, this work can be generalized in several directions, for example, by allowing for first-order formulas instead of essentially propositional formulas (which is what programs with constants and variables over a finite instantiation domain amount to) and nested, non-deterministic, and non-monotonic intensional operators. Furthermore, we may want to consider intensional operators with multiple minimal neighborhoods, by defining IE_∇ as a non-determistic operator that extracts a minimal neighborhood $W' \in \theta_\nabla(w')$. In that case, of course, the alternating fixpoint construction as it is defined now might not result in a unique well-founded model. However, the occurence of non-deterministic operators in the heads of rules is very similar to disjunctive logic programs, where the truth of a head of a rule can also be guaranteed by a choice of different atoms (the disjuncts) being made true. Therefore, we plan to look at techniques from disjunctive logic programming to generate unique well-founded extensions (cf. references in (Knorr and Hitzler 2007)). Finally, the integration with taxonomic knowledge in the form of description logic ontologies (Baader et al. 2007) may also be worth pursuing as applications sometimes require both (see e.g. (Alberti et al. 2011; Alberti et al. 2012; Kasalica et al. 2019)). Hybrid MKNF knowledge bases (Motik and Rosati 2010) are a more prominent approach among the existing approaches for combining non-monotonic rules and such ontologies, and the well-founded semantics for these (Knorr, Alferes, and Hitzler 2011), also based on an alternating fixpoint construction, together with its efficient implementation (Kasalica et al. 2020) may prove fruitful for such an endeavour.

Acknowledgments The authors are indebted to the anonymous reviewers of this paper for helpful feedback. The authors were partially supported by FCT project RIVER (PTDC/CCI-COM/30952/2017) and by FCT project NOVA LINCS (UIDB/04516/2020). J. Heyninck was also supported by the German National Science Foundation under the DFG-project CAR (Conditional Argumentative Reasoning) KE-1413/11-1.

References

Abadi, M., and Manna, Z. 1989. Temporal logic programming. *J. Symb. Comput.* 8(3):277–295.

Alberti, M.; Gomes, A. S.; Gonçalves, R.; Leite, J.; and Slota, M. 2011. Normative systems represented as hybrid knowledge bases. In *CLIMA*, volume 6814 of *LNCS*, 330–346. Springer.

Alberti, M.; Knorr, M.; Gomes, A. S.; Leite, J.; Gonçalves, R.; and Slota, M. 2012. Normative systems require hybrid knowledge bases. In *AAMAS*, 1425–1426. IFAAMAS.

Allen, J. F. 1990. Maintaining knowledge about temporal intervals. In *Readings in qualitative reasoning about physical systems*. Elsevier. 361–372.

Anicic, D.; Rudolph, S.; Fodor, P.; and Stojanovic, N. 2012. Stream reasoning and complex event processing in ETALIS. *Semantic Web* 3(4):397–407.

Arasu, A.; Babu, S.; and Widom, J. 2006. The CQL continuous query language: semantic foundations and query execution. *VLDB J.* 15(2):121–142.

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2007. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2nd edition.

Barbieri, D. F.; Braga, D.; Ceri, S.; Valle, E. D.; and Grossniklaus, M. 2010. C-SPARQL: a continuous query language for RDF data streams. *Int. J. Semantic Computing* 4(1):3–25.

Bazoobandi, H. R.; Beck, H.; and Urbani, J. 2017. Expressive stream reasoning with LASER. In *Procs. of ISWC*, volume 10587 of *LNCS*, 87–103. Springer.

Beck, H.; Dao-Tran, M.; and Eiter, T. 2018. LARS: A logic-based framework for analytic reasoning over streams. *Artif. Intell.* 261:16–70.

Beirlaen, M.; Heyninck, J.; and Straßer, C. 2019. Structured argumentation with prioritized conditional obligations and permissions. *Journal of Logic and Computation* 29(2):187–214.

Brandt, S.; Kalayci, E. G.; Ryzhikov, V.; Xiao, G.; and Zakharyaschev, M. 2018. Querying log data with metric temporal logic. *J. Artif. Intell. Res.* 62:829–877.

Brenton, C.; Faber, W.; and Batsakis, S. 2016. Answer set programming for qualitative spatio-temporal reasoning:

Methods and experiments. In *Technical Communications of ICLP*, volume 52 of *OASICS*, 4:1–4:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Brewka, G.; Ellmauthaler, S.; Gonçalves, R.; Knorr, M.; Leite, J.; and Pührer, J. 2018. Reactive multi-context systems: Heterogeneous reasoning in dynamic environments. *Artif. Intell.* 256:68–104.

Caminada, M.; Sá, S.; Alcântara, J.; and Dvořák, W. 2015. On the equivalence between logic programming semantics and argumentation semantics. *International Journal of Approximate Reasoning* 58:87–111.

Chellas, B. F. 1980. *Modal Logic: An Introduction*. Cambridge University Press.

Chen, Y.; Wan, H.; Zhang, Y.; and Zhou, Y. 2010. dl2asp: implementing default logic via answer set programming. In *European Workshop on Logics in Artificial Intelligence*, 104–116. Springer.

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv.* 33(3):374–425.

del Cerro, L. F. 1986. MOLOG: A system that extends PROLOG with modal logic. *New Generation Comput.* 4(1):35–50.

Gelder, A. V.; Ross, K. A.; and Schlipf, J. S. 1991. The well-founded semantics for general logic programs. *J. ACM* 38(3):620–650.

Gelder, A. V. 1989. The alternating fixpoint of logic programs with negation. In *Procs. of SIGACT-SIGMOD-SIGART*, 1–10. ACM Press.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Comput.* 9(3-4):365–385.

Gelfond, M. 2008. Answer sets. In *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*. Elsevier. 285–316.

Gonçalves, R., and Alferes, J. J. 2012. Specifying and reasoning about normative systems in deontic logic programming. In *Procs. of AAMAS*, 1423–1424. IFAAMAS.

Gonçalves, R.; Knorr, M.; and Leite, J. 2014. Evolving multi-context systems. In *ECAI*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, 375–380. IOS Press.

Governatori, G.; Rotolo, A.; and Riveret, R. 2018. A deontic argumentation framework based on deontic defeasible logic. In *International Conference on Principles and Practice of Multi-Agent Systems*, 484–492. Springer.

Izmirlioglu, Y., and Erdem, E. 2018. Qualitative reasoning about cardinal directions using answer set programming. In *Procs. of AAAI*, 1880–1887. AAAI Press.

Kasalica, V.; Gerochristos, I.; Alferes, J. J.; Gomes, A. S.; Knorr, M.; and Leite, J. 2019. Telco network inventory validation with nohr. In *LPNMR*, volume 11481 of *LNCS*, 18–31. Springer.

Kasalica, V.; Knorr, M.; Leite, J.; and Lopes, C. 2020. NoHR: An overview. *Künstl Intell*.

Knorr, M.; Alferes, J. J.; and Hitzler, P. 2011. Local closed world reasoning with description logics under the well-founded semantics. *Artif. Intell.* 175(9-10):1528–1554.

Knorr, M., and Hitzler, P. 2007. A comparison of disjunctive well-founded semantics. In *FAInt*, volume 277 of *CEUR Workshop Proceedings*. CEUR-WS.org.

McNamara, P. 2019. Deontic logic. In Zalta, E. N., ed., *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2019 edition.

Motik, B., and Rosati, R. 2010. Reconciling description logics and rules. *J. ACM* 57(5):30:1–30:62.

Orgun, M. A., and Wadge, W. W. 1992. Towards a unified theory of intensional logic programming. *The Journal of Logic Programming* 13(4):413–440.

Pacuit, E. 2017. *Neighborhood semantics for modal logic*. Springer.

Panagiotidi, S.; Nieves, J. C.; and Vázquez-Salceda, J. 2009. A framework to model norm dynamics in answer set programming. In *MALLOW*.

Phuoc, D. L.; Dao-Tran, M.; Parreira, J. X.; and Hauswirth, M. 2011. A native and adaptive approach for unified processing of linked streams and linked data. In *Procs. of ISWC*, volume 7031 of *LNCS*, 370–388. Springer.

Przymusinski, T. C. 1991. Stable semantics for disjunctive programs. *New Generation Comput.* 9(3/4):401–424.

Suchan, J.; Bhatt, M.; Walega, P. A.; and Schultz, C. P. L. 2018. Visual explanation by high-level abduction: On answer-set programming driven reasoning about moving objects. In *Procs. of AAAI*, 1965–1972. AAAI Press.

Vardi, M. Y. 1989. On the complexity of epistemic reasoning. In *Procs. of LICS*, 243–252. IEEE Computer Society.

Walega, P. A.; Kaminski, M.; and Grau, B. C. 2019. Reasoning over streaming data in metric temporal datalog. In *Procs. of AAAI*, 3092–3099. AAAI Press.

Walega, P. A.; Schultz, C. P. L.; and Bhatt, M. 2017. Nonmonotonic spatial reasoning with answer set programming modulo theories. *TPLP* 17(2):205–225.