Lower Bound Founded Logic of Here-and-There: A Preliminary Report

Pedro Cabalar
University of Corunna, Spain
cabalar@udc.es

Jorge Fandinno
University of Toulouse, France
jorge.fandinno@irit.fr

Torsten Schaub and Sebastian Schellhorn
University of Potsdam, Germany
{torsten, seschell}@cs.uni-potsdam.de

Abstract

A distinguishing feature of Answer Set Programming is that all atoms belonging to a stable model must be founded. That is, an atom must not only be true but provably true. This can be made precise by means of the constructive logic of Here-and-There, whose equilibrium models correspond to stable models. One way to look at foundedness is to regard Boolean truth values as ordered by letting true be greater than false. Then, each Boolean variable takes the smallest truth values that can be proven for it. This idea was generalized by Aziz to ordered domains and applied to constraint satisfaction problems. As before, the idea is that a, say integer, variable gets only assigned to the smallest integer that can be justified. In this paper, we present a logical reconstruction of Aziz' idea in the setting of the logic of Here-and-There. More precisely, we start by defining the logic of Here-and-There and elaborate upon their formal properties. We then define a logic program fragment dealing with linear constraints over integers and analyze it in terms of concepts from logic programming. Finally, we compare our approach with related ones and sketch future work.

1 Motivation

A distinguishing feature of Answer Set Programming (ASP; Baral 2003) is that all atoms belonging to a stable model must be founded. That is, an atom must not only be true but provably true. This can be made precise by means of the constructive logic of Here-and-There (HT; Heyting 1930), whose equilibrium models correspond to stable models (Pearce 2006). One way to look at foundedness is to regard Boolean truth values as ordered by letting true be greater than false. Then, each Boolean variable takes the smallest truth value that can be proven for it. This idea was generalized in (Aziz 2015) to ordered domains and applied to constraint satisfaction problems. As before, the idea is that a, say integer, variable gets only assigned to the smallest integer that can be justified. We refer to this idea by calling it foundedness. Note that ASP follows the rationality principle, which says that one shall only believe in things one is forced to. In the propositional case this principle amounts to foundedness, whereas for rules like \( x \geq 42 \) there are at least two ways of understanding. First, one believe in any value greater or equal than 42 for \( x \). Second, one believe in value 42 for \( x \) if there is nothing else forcing to believe more than this. The latter one corresponds to our understanding of foundedness.

The literature of ASP contains several approaches dealing with atoms containing variables over non-Boolean domains, among them (Baselice, Bonatti, and Gelfond 2005), (Janhunen et al. 2017) and (Cabalar et al. 2016), but these approaches do not address foundedness in our sense. For instance, Constraint ASP (CASP) approaches like (Baselice, Bonatti, and Gelfond 2005) allow atoms with variables over non-Boolean domains in the body of a rule only. Thus, these atoms and the values of non-Boolean variables cannot be founded in terms of ASP.

Approaches like (Janhunen et al. 2017) and (Cabalar et al. 2016) allow any kind of atoms in heads and bodies. This allows atoms with variables over non-Boolean domains to be founded but their variables are not necessarily assigned to the smallest value that can be justified. Since in the approach of (Cabalar et al. 2016) atoms as well as the values of its variables are founded and defaults are possible, one could think about to use defaults or minimization to achieve foundedness. For instance, \( x = 1 \leftarrow \neg(x \neq 1) \) assigns value 1 to \( x \) by default. If we add fact \( x = 3 \), then we deactivate the default and assign value 3 to \( x \). Similarly, \( x = 0 \leftarrow \neg(x > 0) \) assigns the value 0 by default. However, in general assigning a minimal value by default cannot be done by rules as the above. To point out the difference of foundedness and founded atoms, the following examples illustrate that minimizing assigned values does not restore foundedness either. Consider the rules

\[
\begin{align*}
    x \geq 0 & \quad y \geq 0 & \quad x \geq 42 \leftarrow y < 42 \quad (1)
\end{align*}
\]

The approach of (Cabalar et al. 2016) leads to solutions that assign values greater or equal than 42 to \( x \) and values greater or equal than 0 to \( y \) or vice versa, respectively. Thus, the two solutions with minimal values assign 42 to \( x \) and 0 to \( y \) and the other way around. Note that only the first one respects foundedness, since there is no reason to assign a value greater than 0 to \( y \). Now, consider the rules

\[
\begin{align*}
    x \geq 1 & \quad x \geq 42 \leftarrow \neg(x \leq 1) \quad (2)
\end{align*}
\]

We expect two solutions in terms of foundedness. One assigns the value 1 to \( x \) and the other assigns value 42 to \( x \), since a value greater than 1 forces the derivation of value 42. The
rules of (2) give us no reason to derive a value greater than 42. In contrast, the approach presented in (Cabalar et al. 2016) yield an intuitivo understanding assigning value 1 or a value greater or equal than 42 to x. That is, the corresponding solution with the minimal value assigned to x assigns 1 to x. The second equally founded solution is not obtained.

The existing approach regarding foundedness of (Aziz 2015) behaves counter intuitive. For instance, consider rule \( p \leftarrow \neg p \). Then, Aziz’ approach yields a solution where \( p \) holds instead of no solution as expected in terms of ASP. To this end, we present in the following a logical reconstruc-

![Image](https://via.placeholder.com/150)

tion of Aziz’ idea of foundedness in the setting of the logic of Here-and-There. More precisely, we start by defining the logic of Here-and-There with lower bound founded variables, short \( HT_{LB} \), along with its equilibrium models. We elaborate upon the formal properties of \( HT_{LB} \) regarding persistence, negation and strong equivalence. Furthermore, we point out the relation of \( HT_{LB} \) to \( HT \), and show that our approach corresponds to a straightforward extension of Ferraris’ stable model semantics (Ferraris 2005). We then define a logic pro-
gram fragment dealing with linear constraints over integers and analyze it in terms of concepts from logic programming. Finally, we compare our approach with related ones, to point out the benefits of \( HT_{LB} \) and sketch future work.

2 Background

Let \( \mathcal{A} \) be the set of propositional atoms. A formula \( \varphi \) is a combination of atoms by logical connectives \( \bot, \land, \lor, \) and \( \leftarrow \). As usual, we define \( \top \equiv \bot \rightarrow \bot \) and \( \neg \varphi \equiv \varphi \rightarrow \bot \). A theory is a set of formulas.

We denote an interpretation over \( \mathcal{A} \) by \( I \subseteq \mathcal{A} \) and an \( HT \)-interpretation over \( \mathcal{A} \) by \( \langle H, T \rangle \) where \( H \subseteq T \subseteq \mathcal{A} \) are interpretations. Since we want to abstract from the specific form of atoms in the following sections, we rely upon denotations for fixing their semantics. A denotation of atoms in \( \mathcal{A} \) is a function \( [ \cdot ] : \mathcal{A} \rightarrow 2^\mathcal{A} \) mapping atoms in \( \mathcal{A} \) to sets of interpretations over \( \mathcal{A} \). Accordingly, \( [ p ]_A \equiv \{ I \mid p \in I \} \) represents the sets of interpretations where atom \( p \) holds.

With it, we next define satisfaction of formulas in \( HT \).

Definition 1 Let \( \langle H, T \rangle \) be an \( HT \)-interpretation over \( \mathcal{A} \) and \( \varphi \) a propositional formula over \( \mathcal{A} \). Then, \( \langle H, T \rangle \) satisfies \( \varphi \), written \( \langle H, T \rangle \models \varphi \), if the following conditions hold:

1. \( \langle H, T \rangle \not\models \bot \)
2. \( \langle H, T \rangle \models \varphi \) iff \( H \in [p]_A \) for propositional atom \( p \in \mathcal{A} \)
3. \( \langle H, T \rangle \models \varphi_1 \land \varphi_2 \) iff \( \langle H, T \rangle \models \varphi_1 \) and \( \langle H, T \rangle \models \varphi_2 \)
4. \( \langle H, T \rangle \models \varphi_1 \lor \varphi_2 \) iff \( \langle H, T \rangle \models \varphi_1 \) or \( \langle H, T \rangle \models \varphi_2 \)
5. \( \langle H, T \rangle \models \varphi \) iff \( \langle I, T \rangle \not\models \varphi_1 \) or \( \langle I, T \rangle \not\models \varphi_2 \) for both \( I \in \{ H, T \} \)

As usual, we call \( \langle H, T \rangle \) an \( HT \)-model of a theory \( \Gamma \), if \( \langle H, T \rangle \models \varphi \) for all \( \varphi \) in \( \Gamma \). The usual definition of \( HT \) satisfaction (cf. Pearce 2006) is obtained by replacing Condition 2 above by

\[ \langle H, T \rangle \models p \text{ iff } p \in H \text{ for propositional atom } p \in \mathcal{A} \]

It is easy to see that both definitions of \( HT \) satisfaction coincide.

3 Lower Bound Founded Logic of Here-and-There

In what follows, we introduce the logic of Here-and-There with lower bound founded variables, short \( HT_{LB} \) and elaborate on some formal properties regarding satisfaction. We discuss the relation of complements of atoms regarding negation and we point out the relation between \( HT_{LB} \) and \( HT \) as well as a straightforward extension of Ferraris’ stable model semantics.

3.1 \( HT_{LB} \) and its Properties

The language of \( HT_{LB} \) is defined over a set of atoms \( \mathcal{A}_X \) comprising variables, \( \mathcal{X} \), and constants over an ordered domain \( (D, \geq) \). For simplicity, we assume that each element of \( D \) is uniquely represented by a constant and abuse notation by using \( D \) to refer to the set of constants. Similarly, we identify \( \mathcal{X} \) with its syntactic representative. The specific syntax of atoms is left open but assumed to refer to elements of \( \mathcal{X} \) and \( D \). The only requirement is that we assume that an atom depends on a distinguished subset of variables of \( \mathcal{X} \). An atoms can be understood to hold or not once all variables depending on it are substituted by domain elements. Intuitively, variables not occurring in an atom are understood as irrelevant for the atom evaluation. Examples of ordered domains are \( (\{0,1,2,3\}, \geq) \) and \( (\mathbb{Z}, \geq) \), respectively; corresponding atoms are \( x \geq 42 \) and \( x = y \). A formula \( \varphi \) is a propositional combination of atoms and logical connectives \( \bot, \land, \lor, \rightarrow \). As usual, we define \( \top \equiv \bot \rightarrow \bot \) and \( \neg \varphi \equiv \varphi \rightarrow \bot \). A theory is a set of formulas. For instance, \( \langle y < 42 \land \neg(x = y) \rightarrow x \geq 42 \rangle \) is a formula. Let \( vars(\varphi) \subseteq \mathcal{X} \) be the set of variables and \( atoms(\varphi) \subseteq \mathcal{A}_X \) the atoms occurring in a formula \( \varphi \).

For capturing partiality, we introduce a special domain element \( u \), standing for undefined, and extend \( (D, \geq) \) to \( (\mathcal{D}, \geq) \) where \( \mathcal{D}_u \equiv D \cup \{ u \} \) and \( u \geq \mathcal{D} \equiv \mathcal{D}_u \cup \{ (c, u) \mid c \in \mathcal{D} \} \). With it, we define a (partial) valuation over \( \mathcal{X} \), \( D \) as a function \( v : \mathcal{X} \rightarrow \mathcal{D}_u \) mapping each variable to a domain value or undefined. For comparing valuations by set-based means, we alternatively represent them by subsets of \( \mathcal{X} \times \mathcal{D} \). Basically, any function \( v \) is a set of pairs \( (x, c) \) such that \( v(x) = c \) for \( c \in \mathcal{D} \). In addition, we view a pair \( (x, c) \) as \( x \geq c \) and add its downward closure \( (x, c) \equiv \{(x, d) \mid c, \in \mathcal{D}, c \geq d \} \). Given this, a valuation \( v \) is represented by the set \( \bigcup_{(x, c) \in v} (x, c) \).1 As an example, consider variables \( x \), \( y \) and \( z \) over domain \( \{0,1,2,3\} \cup \{ u \} \geq u \). The valuation \( v = \{ (x, 2), (y, 0) \} \) can be represented by \( v = ((x, 2), (y, 0), (x, 1), (x, 2), (y, 0)) \). Then, \( v' = \{ x \leftarrow 1, y \leftarrow u \} \), viz. \( \{(x, 0), (x, 1)\} \) in set notation, can be regarded as “smaller” than \( v \) because \( v' \subseteq v \). The

1Note that \( (x, u) = \emptyset \), since \( u \not\in \mathcal{D} \).
comparison of two valuations $v$ and $v'$ by their set-based means using $\subseteq$ amounts to a twofold comparison. That is, $v$ and $v'$ are compared regarding the occurrence of variables and their particular values wrt $\preceq$. We let $\forall X \cdot D$ stand for the set of valuations over $X$ and $D$.

We define the satisfaction of formulas over $A_X$ wrt atom denotations over $X$, $D$, which are functions $\llbracket \cdot \rrbracket_{X,D} : A_X \rightarrow 2^{\forall X \cdot D}$ mapping atoms to sets of valuations. Let $a$ be an atom of $A_X$ and $\llbracket a \rrbracket_{X,D}$ its denotation. Then, $\{ a \}_{X,D}$ is the set of valuations $v$ so that $a$ holds. Since $a$ depends on variables $\text{vars}(a) \subseteq X$, we have for each $v \in \llbracket a \rrbracket_{X,D}$ and valuation $v'$ with $v(x) = v'(x)$ for $x \in \text{vars}(a)$ that $v' \in \llbracket a \rrbracket_{X,D}$. Intuitively, values of $X \setminus \text{vars}(a)$ can vary freely without changing the membership of a valuation to $\llbracket a \rrbracket_{X,D}$. For simplicity, we drop indices $X$, $D$ whenever clear from context.

For instance, interpreting the atoms $x \geq 42, 42 \geq 0$ and $0 \geq 42$ over $(\mathbb{Z}, \geq)$ yields the following denotations:

$\llbracket x \geq 42 \rrbracket \equiv \{ v \mid v(x) \geq 42 \}$

$\llbracket 42 \geq 0 \rrbracket \equiv \{ v \mid v(42) \geq 0 \}$

$\llbracket 0 \geq 42 \rrbracket \equiv \emptyset$.

In particular, $\llbracket x \geq 42 \rrbracket$ is the set of valuations where $x$ is assigned to a value greater or equal than 42 and all variables in $X \setminus \text{vars}(x \geq 42)$ take any value of $D_u$, e.g. $(x \leq 45)$ and $(x < y)$ for $y \in X \setminus \text{vars}(x \geq 42)$ are possible valuations. Interestingly, atoms like $x \geq x$ with $\llbracket x \geq x \rrbracket = \{ v \mid v(x) \neq u \}$ force variables to be defined over $D$ per definition of $\preceq$. A valuation $v$ is defined for a set of variables $Y \subseteq X$ if $v(x) \neq u$ for all $x \in Y$.

We define an $HT_{LB}$-valuation over $X \cdot D$ as a pair $(h, t)$ of valuations over $X \cdot D$ with $h \subseteq t$. We define satisfaction of a formula wrt an $HT_{LB}$-valuation as follows.

**Definition 2** Let $(h, t)$ be an $HT_{LB}$-valuation over $X \cdot D$ and $\varphi$ be a formula over $A_X$. Then, $(h, t)$ satisfies $\varphi$, written $\langle h, t \rangle \models \varphi$, if the following holds:

1. $(h, t) \not\models \perp$
2. $(h, t) \models a \iff v \in \llbracket a \rrbracket_{X,D}$ for atom $a \in A_X$ and for both $v \in \{ h, t \}$
3. $(h, t) \models \varphi_1 \land \varphi_2 \iff (h, t) \models \varphi_1$ and $(h, t) \models \varphi_2$
4. $(h, t) \models \varphi_1 \lor \varphi_2 \iff (h, t) \models \varphi_1$ or $(h, t) \models \varphi_2$
5. $(h, t) \models \varphi_1 \rightarrow \varphi_2 \iff (v, v') \not\models \varphi_1$ or $(v, v') \models \varphi_2$ for both $v \in \{ h, t \}$

As usual, we call $(h, t)$ an $HT_{LB}$-model of a theory $\Gamma$, if $(h, t) \models \varphi$ for all $\varphi$ in $\Gamma$. For a simple example, consider the theory containing atom $x \geq 42$ only. Then, every $HT_{LB}$-valuation $(h, t)$ with $h, t \in [x \geq 42]$ is an $HT_{LB}$-model of $x \geq 42$. Note that, different to $HT$, satisfaction of atoms in $HT_{LB}$ forces satisfaction in both $h$ and $t$, instead of $h$ only. We discuss this in detail in Section 3.4.

Our first result shows that the characteristic properties of persistence and negation hold as well when basing satisfaction on valuations and denotations.

**Proposition 2** Let $(h, t)$ and $(t, t)$ be $HT_{LB}$-valuations over $X \cdot D$, and $\varphi$ be a formula over $A_X$. Then:

1. $(h, t) \models \varphi$ implies $(t, t) \models \varphi$, and
2. $(h, t) \models \varphi \rightarrow \perp \iff (t, t) \not\models \varphi$.

Persistence implies that all atoms satisfied by $(h, t)$ are also satisfied by $(t, t)$. To make this precise, let $A(t, h) \models \{ a \mid a \in A_X \land h \in \llbracket a \rrbracket \land t \in \llbracket a \rrbracket \}$ be the set of atoms satisfied by $(h, t)$.

**Proposition 3** Let $(h, t)$ and $(t, t)$ be $HT_{LB}$-valuations over $X \cdot D$. Then, $A((h, t)) \subseteq A(t, h)$

Finally, we define an equilibrium model in $HT_{LB}$.

**Definition 3** An $HT_{LB}$-valuation $(t, t)$ over $X \cdot D$ is an $HT_{LB}$-equilibrium model of a theory $\Gamma$ if $(t, t) \models \Gamma$ and there is no $h \subset t$ such that $(h, t) \models \Gamma$.

We refer an $HT_{LB}$-equilibrium model $(t, t)$ of $\Gamma$ as an $HT_{LB}$-stable model of $\Gamma$. Let us reconsider the theory containing atom $x \geq 42$ only. Then, $t = (x \geq 42)$ is an $HT_{LB}$-stable model of $x \geq 42$, since $t \subseteq [x \geq 42]$ and there is no $h \subset t$ with $h \in [x \geq 42]$. In contrast, neither $HT_{LB}$-model $(t', t')$ with $t' = (x \geq 42) \cup (y \leq 0)$ nor $(t'', t'')$ with $t'' = (x < 45)$ are $HT_{LB}$-stable models since $t$ is a proper subset of both $(t, t') \models x \geq 42$ as well as $(t, t'') \models x \geq 42$ holds. Hence, $HT_{LB}$-stable models make sure that each variable is assigned to its smallest founded value and does not take any value of possible valuations of corresponding denotations.

Note that $HT_{LB}$-equilibrium models induce the non-monotonic counterpart of the monotonic logic of $HT_{LB}$. Following well-known patterns, we show that $HT_{LB}$ allows us to decide strong equivalence wrt $HT_{LB}$-equilibrium models.

**Proposition 4** (Strong Equivalence) Let $\Gamma_1$, $\Gamma_2$ and $\Gamma$ be theories over $A_X$. Then, theories $\Gamma_1 \cup \Gamma$ and $\Gamma_2 \cup \Gamma$ have the same $HT_{LB}$-stable models for every theory $\Gamma$ if $\Gamma \models \Gamma_1$ and $\Gamma_2$ have the same $HT_{LB}$-models.

The idea is to prove the if direction by proving its contraposition, and the only if direction by proving its straightforward implication. The contraposition assumes that there exists an $HT_{LB}$-valuation that satisfies $\Gamma_1$ but not $\Gamma_2$, which implies that the stable models of $\Gamma_1 \cup \Gamma$ and $\Gamma_2 \cup \Gamma$ do not coincide. There are two cases to construct $\Gamma$ in a way that $\Gamma_1 \cup \Gamma$ has a stable model which is not a stable model of $\Gamma_2 \cup \Gamma$ and the other way around, respectively. Let us consider an example to illustrate the idea of the construction of $\Gamma$. Let $h = (x \leq 0)$ and $t = (x \leq 2) \cup (y \leq 1)$ be $HT_{LB}$-valuation over $\{ x, y \}, \{ 0, 1, 2, 3 \}$ with $(h, t) \models \Gamma_1$ and $(h, t) \not\models \Gamma_2$. For the first case assume that $(t, t) \not\models \Gamma_2$. Since $t$ cannot be a model of $\Gamma_2 \cup \Gamma$ by assumption, we construct $\Gamma$ in a way that $t$ is a stable model of $\Gamma_1 \cup \Gamma$. Hence, let $\Gamma = \{ z \leq c \mid (z, c) \in t \} = \{ x \geq 0, x \geq 1, x \geq 2, y \leq 0 \}$ be the theory with the only stable model $t$. By persistence of $(h, t)$ wrt $\Gamma_1$ and construction of $\Gamma$ we get that $t$ is a stable model of $\Gamma_1 \cup \Gamma$ but not of $\Gamma_2 \cup \Gamma$. For the second case we assume that $(t, t) \models \Gamma_2$. Now we construct $\Gamma$ in a way that $t$ is a stable model of $\Gamma_2 \cup \Gamma$ but not of $\Gamma_1 \cup \Gamma$. By assumption we have that $(h, t) \models \Gamma_1$ and $(h, t) \not\models \Gamma_2$ as well as $(t, t) \models \Gamma_2$, thus we want to have $(h, t)$ and $(v, v')$ with $v \subseteq v' \subseteq v''$ as the only models of $\Gamma$. Hence, let $\Gamma = \Gamma' \cup \Gamma''$ with $\Gamma' = \{ z \leq c \mid (z, c) \in h \} = \{ x \geq 0 \}$ the theory that is satisfied by everything that is greater or equal than $h$, and $\Gamma'' = \{ z \geq t(z) \rightarrow z' \geq t(z') \}, z \geq c \rightarrow z \geq t(z) \mid (z, c), (z, t(z)), (z', t(z')) \in t \setminus h \not\models z' \}$
In the following, we elaborate on complements of atoms and its relation to negation, since $\mathcal{A}_X$ may contain atoms like $x \geq 42$ and $x < 42$. Intuitively, one could expect that the strong negation of an atom holds whenever the atom itself does not hold. This can be easily expressed by defining the complement of valuations of an atom denotation. More formally, we characterize the complement $\pi$ of atom $a$ by its denotation $\|a\| \equiv 2^\mathbb{V} \setminus \|a\|$.

To illustrate that the simple complement of an atom is not sufficient to yield something similar to strong negation let us take a closer look on propositional atoms in $HT_{LB}$. For mimicking Boolean truth values, we consider the domain $\{(t, f), \{t \geq f\}\}$. Then, the denotation of propositional atoms in $HT_{LB}$ can be defined as follows: $\|p = t\|_{\mathcal{A}, \{t, f\}} \equiv \{v \mid v(p) = t\}$ and $\|p = f\|_{\mathcal{A}, \{t, f\}} \equiv \{v \mid v(p) = f\}$. Note that $p = t$ and $p = f$ are regarded as strong negations of each other, as in standard case (Gelfond and Lifschitz 1990); its weak negations are given by $\neg(p = t)$ and $\neg(p = f)$, respectively. For instance, the complement $p = t$ is characterized by denotation $\|p = t\| \equiv 2^\mathbb{V} \setminus \|p = t\| \equiv \{v \mid v(p) \neq t\}$. Note that this complement allows valuations $v$ with $v(p) = u$, which does not match $p = t$.

To this end, we define another complement to exclude assigning value undefined to variables of the atom. First, we define a denotation $\|a\|$ of an atom $a$ as strict if each $v \in \|a\|$ is defined for $vars(a)$. Then, we characterize the strict complement $\pi^s$ of atom $a$ by the strict denotation $\|\pi^s\| \equiv 2^\mathbb{V} \setminus \|a\| \cup \{v \mid v(x) = u \text{ for some } x \in vars(a)\}$. Informally, the strict complement of an atom holds whenever all variables are defined and the atom itself does not hold.

That is, atoms $p = f$ and $p = t$ are strict complements of each other.

More generally, an atom with strict denotation and its strict complement can be regarded as being strongly negated to each other. For instance, consider atom $x \geq 42$ and its strict denotation $\|x \geq 42\| = \{v \mid v(x) \geq 42\}$. Then, its strict complement $x \geq 42^s$ is defined by $\|x \geq 42^s\| = \{v \mid u \neq v(x) < 42\}$. As in the Boolean case, the strict complement $x \geq 42^s$ can be seen as the strong negation of $x \geq 42$.

To make the relation of complements and negation precise, let us define entailments. A theory (or a single formula) $\Gamma$ over $\mathcal{A}_X$ entails a formula $\varphi$ over $\mathcal{A}_X$, written $\Gamma \models \varphi$, when all $HT_{LB}$-models of $\Gamma$ are $HT_{LB}$-models of $\varphi$. Then, we have the following result.

**Proposition 5** Let $a$ be an atom over $\mathcal{A}_X$, and $\pi$ and $\pi^s$ its complement and its strict complement over $\mathcal{A}_X$, respectively. Then, $\pi^s = \pi$ and $\pi = \neg a$.

This implies that the strict complement $\pi^s$ of an atom $a$ implies its negation $\neg a$, just as strong negation implies weak negation in the standard case (Pearce 2006). To illustrate that in general the negation of an atom does not entail its complement ($\neg a \not\equiv \pi$), let us consider atom $x \leq 42$ with strict denotation $\{x \leq 42\} = \{v \mid u \neq v(x) \leq 42\}$. Then, the complement $x \leq 42$ is defined by denotation $\{x \leq 42\} = \{v \mid v(x) = u \text{ or } v(x) > 42\}$. For valuations $h = (x \downarrow 42)$ and $t = (x \downarrow 50)$ we have that $(h, t) \models \neg(x \leq 42)$ since $(x \downarrow 50) \not\in \{x \leq 42\}$. In contrast, $(h, t) \models x \leq 42$ does not hold, since $(x \downarrow 42) \not\in \{x \leq 42\}$. Thus, the complement $\pi$ of an atom $a$ can be seen as a kind of negation in between of strong and weak negation.

**3.3 HT_{LB} versus HT**

Analogously to (Cabalar et al. 2016), we next show that HT can be seen as a special case of $HT_{LB}$.

Note that both types of denotations $\|p\|_A$ and $\|p = t\|_{\mathcal{A}, \{t\}}$ of a propositional atom $p$ collect interpretations and valuations assigning true to $p$, respectively. To this end, we define a transformation $\tau$ relating each propositional atom $p$ with corresponding atom $p = t$ by $\tau(p) \equiv p = t$. Let $\Gamma$ be a propositional theory, then $\tau(\Gamma)$ is obtained by substituting each $p \in atoms(\Gamma)$ by $\tau(p)$. Moreover, we extend $\tau$ to interpretations $f$ by $\tau(f) \equiv \{(p, t) \mid p \in I\}$ to obtain a corresponding valuation over $\mathcal{A}, \{t\}$. The next proposition establishes that HT can be seen as a special case of $HT_{LB}$.

**Proposition 6** Let $\Gamma$ be a theory over propositional atoms $\mathcal{A}$ and $(H, T)$ an HT-interpretation over $\mathcal{A}$. Let $\tau(\Gamma)$ be a theory over atoms $\{p = t \mid p \in \mathcal{A}\}$ and $(\tau(H), \tau(T))$ an $HT_{LB}$-valuation over $\mathcal{A}, \{t\}$. Then, $(H, T) \models \tau(\Gamma)$ iff $(\tau(H), \tau(T)) \models \tau(\Gamma)$.

This can be generalized to any arbitrary singleton domain $\{d\}$ and corresponding atoms $p = d$ and the relationship still holds.

We obtain the following results relating $HT_{LB}$ and $HT$:

**Proposition 7** Let $\Gamma$ be a theory over $\mathcal{A}_X$ and $(h, t)$ an $HT_{LB}$-model of $\Gamma$ over $X, D$. Then, $(\mathcal{A}(h, t)), (\mathcal{A}(t, t))$ is an HT-model of $\Gamma$ over $\mathcal{A}_X$.

That is, the collected atoms satisfied by an $HT_{LB}$-model of $\Gamma$ can be seen as an HT-model of $\Gamma$ by interpreting $\mathcal{A}_X$ as propositional atoms. For instance, consider the theory containing only atom $x \neq y$ and its denotation $\{x \neq y\} \equiv \{v \mid u \neq v(x) \neq v(y) \neq u\}$. Let $h = (x \downarrow 0) \cup (y \downarrow 4)$ and $t = (x \downarrow 4) \cup (y \downarrow 42)$ be valuations and hence $\mathcal{A}(h, t) = \mathcal{A}(t, t) = \{x \neq y\}$ interpretations. Then, $(h, t) \models x \neq y$ in $HT_{LB}$ and $(\mathcal{A}(h, t)), (\mathcal{A}(t, t))$ $\models x \neq y$ in $HT$.

Furthermore, we relate tautologies in $HT$ and $HT_{LB}$.

**Proposition 8** Let $\varphi$ be a tautology over $\mathcal{A}$ and $\varphi'$ a formula over $\mathcal{A}_X$ obtained by replacing all atoms in $\varphi$ by atoms of $\mathcal{A}_X$. Then, $\varphi'$ is a tautology in $HT_{LB}$.

That is, tautologies in $HT$ are independent of any form of atoms.

**3.4 HT_{LB}-stable versus Ferraris-style stable models**

As mentioned, in Definition 2 satisfaction of atoms differs from $HT$ by forcing satisfaction in both $h$ and $t$, instead of $h$ only. This is necessary to satisfy persistence in $HT_{LB}$. In fact, let $HT_{LB}$-valuation $(h, t)$ satisfy atom $a$ in $\mathcal{A}_X$, and by persistence $HT_{LB}$-valuation $(t, t)$ satisfies $a$ as well, but
not necessarily each $HT_{LB}$-valuation $(v, t)$ with $h \subset v \subset t$ satisfies $a$. For instance, consider atom $x \neq 42$ with $[x \neq 42] = \{ v \mid u \neq v(x) \neq 42 \}$. Let $h = (x \mid 0)$ and $t = (x \mid 53)$ be valuations. Then, $(h, t) \models x \neq 42$ and $(t, t) \models x \neq 42$, but for $v = (x \mid 42)$ with $h \subset v \subset t$ we have $(v, t) \models x \neq 42$.

A question that arises now from the above is whether $HT_{LB}$ behaves as expected in terms of stable models semantics. To this end, we give a straightforward definition of classical satisfaction and of the reduct put by Ferraris in (Ferraris 2005) in our setting and show that equilibrium models correspond to stable models according to the resulting Ferraris’-like stable model semantics. We define the counterpart of classical satisfaction as follows.

**Definition 4** Let $t$ be a valuation over $\mathcal{X}, \mathcal{D}$ and $\varphi$ a formula over $A_X$. Then, $t$ satisfies $\varphi$, written $t \models_{cl} \varphi$, if the following holds:

1. $t \not\models_{cl} \bot$
2. $t \models_{cl} a$ iff $a \in \mathcal{D}$ for atom $a \in A_X$
3. $t \models_{cl} \varphi_1 \land \varphi_2$ iff $t \models_{cl} \varphi_1$ and $t \models_{cl} \varphi_2$
4. $t \models_{cl} \varphi_1 \lor \varphi_2$ iff $t \models_{cl} \varphi_1$ or $t \models_{cl} \varphi_2$
5. $t \models_{cl} \neg \varphi_1 \rightarrow \neg \varphi_2$ iff $t \models_{cl} \neg \varphi_1$ or $t \models_{cl} \varphi_2$.

We call $t$ a classical model of a theory $\Gamma$, if $t \models_{cl} \varphi$ for all $\varphi$ in $\Gamma$. We define a Ferraris-like reduct, short $F$-reduct, wrt atoms $A_X$ as follows.

**Definition 5** Let $\varphi$ be a formula over $A_X$ and $t$ a valuation over $\mathcal{X}, \mathcal{D}$. Then, the $F$-reduct of $\varphi$ over $t$, written $\varphi^t$, is given by

$$\varphi^t \equiv \begin{cases} \bot & \text{if } t \not\models_{cl} \varphi \\ a & \text{if } t \models_{cl} \varphi \text{ and } \varphi = a \text{ atom of } A_X \\ \varphi_1^t \land \varphi_2^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \land \varphi_2) \\ \varphi_1^t \lor \varphi_2^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \lor \varphi_2) \\ \neg \varphi_1^t \rightarrow \neg \varphi_2^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \rightarrow \varphi_2) \\ \varphi_1^t \lor \varphi_2^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \lor \varphi_2) \\ \varphi_1^t \land \varphi_2^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \land \varphi_2) \\ \varphi_1^t \rightarrow \varphi_2^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \rightarrow \varphi_2) \\ \varphi_1^t \lor \varphi_2^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \lor \varphi_2) \\ \end{cases}$$

For theory $\Gamma$ and $HT_{LB}$-valuation $t$, we define $\Gamma^t \equiv \{ \varphi^t \mid \varphi \in \Gamma \}$. Note that in case of propositional atoms the $F$-reduct corresponds to Ferraris’ reduct.

We define an $F$-stable model as expected according to classical satisfaction and the $F$-reduct above.

**Definition 6** A valuation $t$ over $\mathcal{X}, \mathcal{D}$ is an $F$-stable model of theory $\Gamma$ over $A_X$ iff $t \models_{cl} \Gamma^t$ and there is no $h \subset t$ such that $h \models_{cl} \Gamma^t$.

The next propositions shows that models in $HT_{LB}$ can be alternatively characterized in the style of Ferraris, rephrasing (Ferraris 2005, Lemma 1):

**Proposition 9** Let $(h, t)$ be an $HT_{LB}$-valuation over $\mathcal{X}, \mathcal{D}$ and $\Gamma$ a theory over $A_X$. Then, $h \models_{cl} \Gamma^t$ iff $(h, t) \models \Gamma$.

As a special case, we obtain that every $HT_{LB}$-stable model corresponds to an $F$-stable model and vice versa.

**Corollary 1** Let $t$ be a valuation over $\mathcal{X}, \mathcal{D}$ and $\Gamma$ a theory over $A_X$. Then, $t$ is an $HT_{LB}$-stable model of $\Gamma$ iff $t$ is an $F$-stable model of $\Gamma$.

The last two results have shown that our logic follows well known patterns wrt different representations of stable models.

### 4 Bound Founded Programs with Linear Constraints

In this section, we focus on atoms representing linear constraints over integers and analyze them in terms of concepts known from ASP. Due to space limitations, we present proofs and some preliminaries needed for the following results in an extended version of this work. We illustrate the modelling capabilities of this fragment of $HT_{LB}$ on an example of error diagnosis.

#### 4.1 Programs and its Properties

Reconsider the ordered domain of integers $(\mathbb{Z}, \geq)$. We define a linear constraint atom as

$$\sum_{i=1}^{m} w_i x_i < k$$

where $w_i, k \in \mathbb{Z}$ are constants, $x_i \in \mathcal{X}$ are distinct variables, and $\{\geq, \leq, \neq, =\}$ is a binary relation. By $\mathcal{L}_X$ we denote the set of linear constraint atoms wrt $\mathcal{X}$ and $\mathbb{Z}$. The denotation of a linear constraint atom is given by

$$[\sum_{i=1}^{m} w_i x_i < k] = \{ v \mid \sum_{i=1}^{m} w_i v(x_i) < k, v(x_i) \neq u \}.$$

A linear constraint atom $\alpha$ and its negation $\neg \alpha$ are called linear constraint literals. In the following, we just say atoms and literals.

We define logic programs as follows.

**Definition 7** A formula over $\mathcal{L}_X$ is called a rule if it is of form

$$a_1 \lor \cdots \lor a_n \leftarrow l_1 \land \cdots \land l_{n'}$$

where $a_i$ is an atom for $1 \leq i \leq n$ and $l_j$ is a literal for $1 \leq j \leq n'$ both over $\mathcal{L}_X$.

A logic program is a theory of rules of form (3). Following logic programming syntax, we use ‘$\land$’ and ‘$\lor$’ as alternative representations of $\land$ and $\lor$, respectively. Moreover, in this context we write $\varphi_1 \rightarrow \varphi_2$ for $\varphi_2 \leftarrow \varphi_1$ for formulas $\varphi_1$ and $\varphi_2$. Examples of programs over $\mathcal{L}_X$ are given in the introduction.

Let $r$ be a rule of form (3). Then, we define by $\text{head}(r) \equiv \{ a_i \mid 1 \leq i \leq n \}$ and $\text{body}(r) \equiv \{ l_j \mid 1 \leq j \leq n' \}$ the set of literals of the left and right hand side of $r$, respectively. Whenever $\text{body}(r) = \emptyset$, then we drop $\leftarrow$ and call $r$ fact. If $\text{head}(r) = \emptyset$ we write $\leftarrow l_1, \ldots, l_{n'}$. Rules of latter form are called integrity constraints; they eliminate all models satisfying their body. The following result is related to integrity constraints.

**Proposition 10** Let $P$ be a program over $\mathcal{L}_X$ containing a rule of form $a \leftarrow \neg \alpha$ and for each $HT_{LB}$-stable model $v$ of $P \setminus \{ a \leftarrow \neg \alpha \}$ over $\mathcal{X}, \mathcal{Z}$ we have that $(v, v) \neq a$.

Then, $P$ has no $HT_{LB}$-stable model.

---

As usual, $w_1 x_1 + \cdots + w_n x_n < k$ and $w_1 x_1 + \cdots + w_n x_n > k$ can be expressed by $w_1 x_1 + \cdots + w_n x_n \leq k - 1$ and $w_1 x_1 + \cdots + w_n x_n \geq k + 1$, respectively.
We define an atom \( a \) as monotonic (resp. anti-monotonic) wrt variable \( x \) if \( v \models a \) implies \( v' \models a \) for every valuation \( v' \) with \( v' \subseteq v \) and \( v'(x) \neq u \), where \( v(y) = v'(y) \) for all \( y \in \text{vars}(a) \setminus \{x\} \). We define an atom \( a \) as monotonic (resp. anti-monotonic) if it is monotonic (resp. anti-monotonic) wrt all variables in \( \text{vars}(a) \), and non-monotonic otherwise. Analogously, a program \( P \) is monotonic (resp. anti-monotonic) if all atoms occurring in it are monotonic (resp. anti-monotonic). We call a program \( P \) directed if no atom in it is non-monotonic.

Thus, we define normal programs as follows.

Definition 8 A rule over \( \mathcal{L}_X \) is normal if it is of form
\[
\begin{align*}
& a_0 \leftarrow a_1, \ldots, a_n, \neg a_{n+1}, \ldots, \neg a_{n'} \quad (4) \\
\text{where } |\text{vars}(a_0)| = 1 \text{ and each atom } a_i \text{ is monotonic for } n+1 \leq i \leq n'.
\end{align*}
\]

A normal program is a set of rules of form (4). As the program in (2) illustrates, programs containing rule bodies with non monotonic atoms in the scope of negation, like \( \neg(x \leq 1) \), may lead to stable models which are not subset minimal. As in \( \text{ASP} \), we have that stable models of normal programs are subset minimal.

The fact that stable models are subset minimal is also known as anti-chain property.

Note that our definition of monotonicity of atoms differs from Aziz’ ones (Aziz 2015), due to different concepts of valuations.

Proposition 11 Let \( P \) be a normal program over \( \mathcal{L}_X \). Then, each \( \mathcal{HT}_{LB} \)-stable model of \( P \) over \( X, Z \) is subset minimal.

To elaborate more on the influence of atomic monotonicity on programs, let us consider the following example. Let \( \tilde{P} \) be a directed program, in which no atom occurs in the scope of negation:
\[
\begin{align*}
& x \geq 0 \\
& y \geq 0
\end{align*}
\]

Then, \( \tilde{P} \) has the two stable models \( (x \geq 42) \cup (y \leq 42) \) and \( (x \leq 42) \cup (y \leq 42) \). Compare this with the \( \text{ASP} \) program \( \{ a \leftarrow \neg b \}, \{ b \leftarrow \neg a \} \) formulating an “even loop” yielding stable models \( \{ a \} \) and \( \{ b \} \). Both programs behave similarly, since assigning \( x \) (or \( y \)) to 42 disables the foundedness of 42 for \( y \) (or \( x \)) in the same way as assigning \( a \) (or \( b \)) to true disables the foundedness of true for \( b \) (or \( a \)). That is, not monotonic atoms implicitly involve negation.

The previous example motivates us to define positive programs. To this end, we first define the positive and negative body of a rule. Let \( r \) be a normal rule of form (4), then we define the positive body of \( r \) as \( \text{body}^+(r) = \{ a_i | 1 \leq i \leq n, a_i \text{ monotonic} \} \) and its negative body as \( \text{body}^-(r) = \text{body}(r) \setminus \text{body}^+(r) \). That is, atoms like \( x < 42 \) not occurring in the scope of negation belong to the negative body, since they are not monotonic.

Then, we define positive programs as follows.

Definition 9 A normal rule \( r \) over \( \mathcal{L}_X \) is positive if head \( (r) \) is monotonic and body \( ^-(r) = \emptyset \).

A positive program is a set of positive rules.

The following result shows that a positive program has a unique stable model, just as in \( \text{ASP} \) (Apt, Blair, and Walker 1987).

Proposition 12 Let \( P \) be a positive program over \( \mathcal{L}_X \). Then, \( P \) has exactly one \( \mathcal{HT}_{LB} \)-stable model over \( X, Z \).

The proof follows the well-known idea of applying a fix point calculation using a continuous and monotonic operator.

In \( \text{ASP} \), a program is stratified if it is free of recursion through negation (Apt, Blair, and Walker 1987), also referred to “negative loops”. This idea remains the same in case of \( \mathcal{HT}_{LB} \). Note that we drop in this work the preliminaries needed for the following results, due to space limitations. That is, we give the definitions of dependency graph, loop, stratification and splitting set in terms of \( \mathcal{HT}_{LB} \) in an extended work of this version.

The next results generalize the calculation of a stable model to stratified programs.

Proposition 13 Let \( P \) be a stratified program over \( \mathcal{L}_X \) with monotonic heads only. Then, \( P \) has exactly one \( \mathcal{HT}_{LB} \)-stable model over \( X, Z \).

Interestingly, allowing not monotonic atoms in the head may eliminate stable models but it does not produce further stable models. That is, if we drop the additional condition on heads, then we can still apply a fix point calculation and get the following result.

Proposition 14 Let \( P \) be a stratified program over \( \mathcal{L}_X \). Then, \( P \) has at most one \( \mathcal{HT}_{LB} \)-stable model over \( X, Z \).
For instance, the program consisting of facts \( x \geq 42 \) and \( x < 42 \) only has no \( HT_{LB} \)-stable model.

### 4.2 Modelling Capabilities

In this section, we go into an example of error diagnosis to illustrate some modelling features of \( HT_{LB} \) in terms of programs. In particular, the following example illustrates foundedness and default valuations.

Let \( N = \{1, \ldots, n\} \subseteq \mathbb{Z} \) be an index set. We represent events by constants \( e_i \) and identify them with value \( i \) for \( i \) in \( N \). Consider program \( P \),

\[
\begin{align*}
\text{HT} & \quad \text{BFASP} \\
\text{5.1} & \quad \text{HT}_{LB} \\
\end{align*}
\]

This is beyond some threshold value \( 4 \). If the value of \( \text{temperature} \) falls below 42 degrees, then event \( e_4 \) occurs, modelled by Rule (7). Rule (8) sets the default value of \( \text{temperature} \) to 60 degrees.

To illustrate the behaviour of \( P_{err} \) let us consider the specific instance \( I_{err} \) containing fact \( \text{occur}(e_3) = 1 \). Then, we get the single stable model \( (\text{temperature} \geq 60) \cup (\text{error} \downarrow 3) \cup (\text{occur}(e_3) \downarrow 1) \) of \( P_{err} \cup I_{err} \). The minimal founded value of \( \text{temperature} \) is the default value 60. Since \( e_3 \) is the only event that occurs, by (5) we derive \( \text{error} \geq e_3 \) and thus the minimal founded value for \( \text{error} \) is 3.

Let us extend \( I_{err} \) to \( I'_{err} \) by adding \( \text{temperature} \geq 42 \). Then we get stable models \( (\text{temperature} \downarrow 60) \cup (\text{error} \downarrow 3) \cup (\text{occur}(e_3) \downarrow 1) \) and \( (\text{temperature} \downarrow 42) \cup (\text{error} \downarrow 9) \cup (\text{occur}(e_2) \downarrow 1) \cup (\text{occur}(e_3) \downarrow 1) \cup (\text{occur}(e_4) \downarrow 1) \) of \( P_{err} \cup I_{err} \). Note that for one stable model the default valuation of \( \text{temperature} \) is founded and for the other one not, due to non-monotonic atom \( \text{temperature} \neq 60 \) in the scope of negation. Hence, we derive \( \text{error} \geq e_3 \) and \( \text{error} \geq e_2 + e_3 + e_4 \), respectively.

### 5 Related Work

In this section, we compare \( HT_{LB} \) to existing formalisms.

#### 5.1 BFASP

First, let us compare \( HT_{LB} \) to Aziz’ bound founded \( ASP \) (BFASP; Aziz 2015), since both share the same motivation to generalize the idea of foundedness to ordered domains.

Let us point out some differences of both approaches. In \( BFASP \) an arbitrary formula is called constraint and a rule is defined as a pair of a constraint and a variable called head. The constraint needs to be increasing wrt its head variable. A constraint is increasing in one of its variables if the constraint holds for a substitution of its variables by domain values and it holds for each substitution where the value of the particular variable is increased and rest stays the same as before.\(^6\) Note that the definition of increasing is made for constraints and does not differentiate between the monotonicity of atoms and logic connectives. In case of atoms Aziz’ definitions of increasing and ours of monotonic coincide. Stable models are defined in \( BFASP \) via a reduct depending on the monotonicity of constraints wrt their variables and by applying a fix point operation.

Both, \( BFASP \) and \( HT_{LB} \) assign variables to their smallest domain value per default. Interestingly, they differ in their understanding of smallest domain values. In \( HT_{LB} \), the smallest domain value is always the value undefined to capture partiality, whereas in \( BFASP \) partiality is not considered if the value undefined is not explicitly part of a given domain.

However, the value of the head variable is derived by the constraint even if it contains no implication. For instance, let \( Z x^\ast \) be the variable domain of positive integers 0 and \( (x + y \geq 42, x) \) a rule in \( BFASP \). Then, \( BFASP \) yields one stable model assigning \( x \) to 42 and \( y \) to 0. The value of \( x \) is derived from the value of \( 42 - y \), obtained by the smallest value of \( y \). Per default the value of \( y \) is 0, since \( y \) appears never as an head. This is different from \( HT_{LB} \) where the fact \( x + y \geq 42 \) results in two stable models \( (x \downarrow 0) \cup (y \downarrow 42) \) and \( (x \downarrow 42) \cup (y \downarrow 0) \). In \( HT_{LB} \), the variables of a fact are treated in an equal way instead of an implicative way by declaring one of them as head.

Now, we show that \( BFASP \) does not satisfy the same well-known properties as \( HT_{LB} \). In particular, \( BFASP \) does not satisfying Proposition 10 in its turn. That is, in \( BFASP \) we may get unintuitive stable models. For instance, consider \( ASP \) rule \( p \leftarrow \lnot p \). This rule has no stable model in \( ASP \) and \( HT_{LB} \), since if \( p \) holds then we cannot derive \( p \) any more and if \( p \) not holds then we need to derive \( p \). In contrast, \( BFASP \) yields the stable model assigning \( p \) to true, since the reduct will never replace head variables and produce the rule as it is. Hence, \( BFASP \) yields the stable model assigning \( p \) to true, since it is the minimal (and only) model of the rule.

#### 5.2 \( HT_C \)

Next, we compare our approach to the logic of Here-and-There with constraints \( HT_C \); Cabalar et al. (2016).

First, note that both are based on \( HT \) and capture theories over (constraint) atoms in a non-monotonic setting and can express default values. The difference is that \( HT_{LB} \) inherently minimizes valuations wrt foundedness. This is achieved by additionally comparing valuations wrt the values assigned to the variables. Hence, we represent valuations by sets of downward closed tuples regarding the assignments to yield a comparison of values in a set based mean using standard subset relation. For instance, consider the fact \( x \geq 42 \) over \( \{x\} \), \( Z \) and valuations \( v \) and \( v' \) with \( v(x) = 42 \) and \( v'(x) = 43 \). Then, in \( HT_C \) we have \( v \neq v' \), whereas in \( HT_{LB} \) we have \( v \subseteq v' \). Hence, \( v \) and \( v' \) are stable models in \( HT_C \) but only the first one is \( HT_{LB} \)-stable model wrt foundedness.

\(^5\)Note that (5) leads to exponentially many rules; it is also possible to write this in a more compact way using nested expressions, what we not do in this work for reasons of simplicity.

\(^6\)For more details see (Aziz 2015).
On a first look, $HT_{LB}$ seems like $HT_C$ with value minimization on top. However, this is insufficient, since it does not yield foundedness. Recall program $P$ in (2) with $HT_{LB}$-stable models $(x \downarrow 1)$ and $(x \downarrow 42)$. In contrast, the minimal stable model in $HT_C$ assigns $x$ to 1. This eliminates the second $HT_{LB}$-stable model. Moreover, program $P$ in (1) has the sole $HT_{LB}$-stable model $(x \downarrow 42) \cup (y \downarrow 0)$. Whereas in $HT_C$, we get two stable models with minimal values: one assigns $x$ to 42 and $y$ to 0, and the other $x$ to 0 and $y$ to 42.

However, both $HT_{LB}$ and $HT_C$ define atomic satisfaction in terms of atom denotations. A difference is that in $HT_C$ denotations need to be closed.\(^7\) Informally, a denotation is closed if for each valuation of the denotation every valuation which is a superset is in the denotation as well. For $HT_{LB}$ this cannot be maintained, due to the additional comparison of valuations regarding values. For instance, consider atom $x \neq 42$ with $\| x \neq 42 \| = \{ v \mid u \neq v(x) \neq 42 \} \cup \{ x \}, \mathbb{Z}$. Then, valuations $v$ and $v'$ with $v(x) = 0$ and $v'(x) = 99$ are part of the denotation, but $v''$ with $v''(x) = 42$ and $v \subseteq v'' \subseteq v'$ is not. The reason to be closed or not is that $v$, $v'$ and $v''$ are different in $HT_C$ but subsets in $HT_{LB}$, respectively.

The closure of denotations is significant to satisfy persistence in $HT_C$. In contrast, in $HT_{LB}$ persistence is maintained by forcing atomic satisfaction in both $h$ and $i$, instead of $h$ only as in $HT_C$. The corresponding benefit is that this allows us to consider atoms in $HT_{LB}$ which are not allowed in $HT_C$, like $x = y$ with $\| x = y \| = \{ v \mid v(x) = v(y) \}$ which is not closed in $HT_C$ as well.

With $HT_C$ and $HT_{LB}$ we have two different paradigms, where one is maybe better suited than the other for a particular application area. We plan to further elaborate on possible application areas and the relation of $HT_C$ and $HT_{LB}$.

### 5.3 Other Formalisms

**ILP**  Let us compare Integer Linear Programming (ILP; Schrijver 1999) with $HT_{LB}$.

Note that ILP is a monotone theory. Hence, compared to ASP it is not intuitive to model recursion like reachability using ILP. For instance, in (Liu, Janhunen, and Niemelä 2012) it is mentioned that it is not easy to represent loop formulas in ILP which are needed for this purpose.

To overcome this shortcoming, approaches like $HT_{LB}$ and $HT_C$ tried to integrate monotone theories as ILP in a non-monotonic setting. In other words, these approaches can be seen as non-monotonic counterparts of ILP which support an intuitive modelling of reachability and thus recursion, like in ASP. That is, the benefit of an intuitive modelling is a key difference of $HT_{LB}$ to ILP.

As we will show in an extended version of this work, aggregates under Ferraris’ semantics (Ferraris 2011) can be represented by atoms in $HT_{LB}$. To this end, we restrict conditions of aggregates to propositional atoms. Note that this is not a very limiting restriction, since these atoms can be seen as auxiliaries for arbitrary formulas.

This is interesting, since it means that aggregates are no longer an extension of an existing approach, instead aggregates under Ferraris’ semantics are now already integrated as atoms of an approach. Hence, the results shown in this work allow us to view aggregates in a new setting and give us a possibly better way to elaborate on their properties like monotonicity. Maybe the view on aggregates as atoms in context of $HT_{LB}$ helps us to better understand the existing discussion of different aggregate semantics and their properties.

### 6 Conclusion

We presented the idea of foundedness for minimal values of variables over ordered domains in the setting of the logic of Here-and-There. We elaborated on important properties like persistence, negation and strong equivalence and showed that they hold in our approach. Furthermore, we pointed out that the base logic $HT$ can be seen as a special case of $HT_{LB}$. To prove if our approach follows well-known patterns, we showed that $HT_{LB}$-stable models correspond to stable models according to a Ferraris’-like stable model semantics.

---

\(^7\)Please see (Cabalar et al. 2016) for more details.
To elaborate on our approach in terms of logic programming and modelling, we isolated a fragment dealing with linear constraints over integers. In this context, we analyzed the influence of monotonicity of atoms on programs and concepts like normal, stratified and positive. Moreover, we illustrated the features of foundedness and defaults with the example of error diagnosis.

Finally, we compared our approach to related ones and showed that foundedness is a non-trivial key feature of $HT_{LB}$. We showed that $HT_{LB}$ and BFASP have the same starting motivation but differ in their treatments of undefined and monotonicity. Furthermore, we pointed out that $HT_{LB}$ can be seen as non-monotonic counterpart of monotonic theories. We also mentioned that $HT_{LB}$ offers a new view of aggregates under Ferraris’ semantics as atoms with its corresponding monotonic properties. Thus, aggregates are integrated in $HT_{LB}$ instead of being an extension of an existing approach.

In an extended version we plan to present a fix point operator, dependency graph, (odd and even) loops, stratification, splitting sets, and the relation to aggregates in detail.

References


Appendix of Proofs

Proof of Proposition 1 To prove that $\langle H, T \rangle \models \varphi$ holds under Definition 1 iff it holds when replacing Condition 2 by 2* for $\langle H, T \rangle$ $HT$-interpretation over $\mathcal{A}$ and $\varphi$ a propositional formula over $\mathcal{A}$, it is enough to prove equivalence of base cases 2 and 2*, since the rest follows directly by structural induction. Per definition of denotation we have for propositional atom $p \in \mathcal{A}$ that

$$H \in \downarrow p \iff H \in \{ I \mid p \in I \} \iff p \in H$$

$\Box$
Proof of Proposition 2  It is enough to prove the proposition for the case base, since the rest follows directly by structural induction. Let \( \langle h, t \rangle \) an \( H_{LB} \) -valuation over \( \mathcal{X} \) and an atom of \( \mathcal{X} \).

First, we prove persistence, represented by \( I \) of the proposition. We have
\[
\langle h, t \rangle \models a \iff a \in \{ a \} \\land t \in \{ a \} \implies t \in \{ a \} \iff \langle t, t \rangle \models a
\]

Subsequently, we prove negation, represented by \( 2 \) of the proposition. We have
\[
\langle h, t \rangle \models a \iff a \rightarrow \bot
\]
\[
\iff \langle h, t \rangle \models \bot \lor \langle h, t \rangle \not\models a \quad \land \quad \langle t, t \rangle \not\models \bot \lor \langle t, t \rangle \not\models a
\]
\[
\iff \langle h, t \rangle \not\models a \lor \langle t, t \rangle \not\models a
\]
\[
\iff \langle h, t \rangle \not\models a \land \langle t, t \rangle \not\models a
\]

Proof of Proposition 3  For any \( a \in \mathcal{A}(\langle h, t \rangle) = \{ a \in \mathcal{A} | h \in \{ a \} \text{ and } t \in \{ a \} \} \), we have \( h \in \{ a \} \) and \( t \in \{ a \} \). Thus, we conclude \( a \in \mathcal{A}(\langle t, t \rangle) = \{ a \in \mathcal{A} | t \in \{ a \} \}. \)

Proof of Proposition 4  Let \( \Gamma_1, \Gamma_2 \) and \( \Gamma \) be theories over \( \mathcal{A} \). First, we prove \( \forall \) of the proposition. For each \( H_{LB} \)-valuation \( \langle h, t \rangle \) over \( \mathcal{X} \), \( D \) we have \( \langle h, t \rangle \models \Gamma_1 \iff \langle h, t \rangle \models \Gamma_2 \). This implies that \( \langle h, t \rangle \models \Gamma_1 \cup \Gamma \iff \langle h, t \rangle \models \Gamma_2 \cup \Gamma \) for any \( \Gamma \). Hence, \( \Gamma_1 \cup \Gamma \) and \( \Gamma_2 \cup \Gamma \) have the same \( H_{LB} \)-stable models for every \( \Gamma \).

Secondly, we prove \( \exists \) by contradiction. Without loss of generality, assume that \( \langle h, t \rangle \) is \( H_{LB} \)-valuation over \( \mathcal{X} \), \( D \) with \( \langle h, t \rangle \models \Gamma_1 \) and \( \langle h, t \rangle \not\models \Gamma_2 \). Then, we differ two cases.

Case 1: Let \( \langle h, t \rangle \not\models \Gamma_2 \). We have \( \langle h, t \rangle \models \Gamma_1 \) and thus by persistence (Proposition 2) \( \langle t, t \rangle \models \Gamma_1 \). Let \( \Gamma = \{ x \geq c \mid \langle x, c \rangle \in t \} \). Then, \( \langle t, t \rangle \models \Gamma_1 \cup \Gamma \) is \( H_{LB} \)-stable model but \( \langle t, t \rangle \not\models \Gamma_2 \cup \Gamma \) by assumption.

Case 2: Let \( \langle h, t \rangle \models \Gamma_2 \). Moreover, let \( \Gamma = \Gamma_1 \cup \Gamma' \) with \( \Gamma' = \{ x \geq c \mid \langle x, c \rangle \in h \} \) and \( \Gamma'' = \{ x \geq t(x) \rightarrow y \geq t(y), x \geq c \rightarrow x \geq t(x), \langle x, c \rangle, \langle x, t(x) \rangle, (y, t(y)) \in t \} \). Then, \( \langle t, t \rangle \models \Gamma_2 \cup \Gamma \) by assumption and \( h \subseteq t \). Note there is no \( v \in t \subseteq \{ x \} \subseteq \{ v \} \subseteq \{ a \} \) and \( v \not\in \{ a \} \). Hence, \( \langle v, , t \rangle \not\models \Gamma \) for \( h \subseteq v \subseteq t \). Thus, \( \langle t, t \rangle \) is \( H_{LB} \)-stable model of \( \Gamma_2 \cup \Gamma \) by assumption and construction, we have \( \langle h, t \rangle \models \Gamma_1 \) and \( \langle h, t \rangle \models \Gamma_1 \cup \Gamma \), respectively. Moreover, we have that \( \langle h, t \rangle \not\models a \) for every \( a \in \text{atoms}(\Gamma'') \). Hence, \( \langle h, t \rangle \models \Gamma'' \) and thus \( \langle h, t \rangle \models \Gamma_1 \cup \Gamma \). Note that since \( \langle h, t \rangle \not\models \Gamma_2 \) and \( \langle t, t \rangle \models \Gamma_2 \), we have \( \langle h, t \rangle \not\models \Gamma \), which implies \( h \subseteq t \). Finally, \( \langle t, t \rangle \) is no \( H_{LB} \)-stable model of \( \Gamma_1 \cup \Gamma \).

Proof of Proposition 5  Let \( a \) be an atom over \( \mathcal{A} \), and \( \mathcal{X} \) and \( \mathcal{X}' \) its complement and its strict complement over \( \mathcal{A} \), respectively.

First, we prove \( \mathcal{X}' \models a \). For any \( H_{LB} \)-valuation \( \langle h, t \rangle \) over \( \mathcal{X} \), \( D \) we have
\[
\Rightarrow h \in \{ \mathcal{X}' \} \land t \in \{ \mathcal{X}' \} \iff | \{ \mathcal{X}' \} | = 2^{\mathcal{X}} \setminus \{ a \} \cup \{ \forall \mid v(x) = u \text{ for some } x \in \text{vars}(a) \} \]

\[
\Rightarrow (h, t) \models \mathcal{X}'
\]

Secondly, we prove \( \mathcal{X} \models \neg a \). For any \( H_{LB} \)-valuation \( \langle h, t \rangle \) over \( \mathcal{X} \), \( D \) we have
\[
\Rightarrow h \in \{ \mathcal{X} \} \land t \in \{ \mathcal{X} \} \iff | \{ \mathcal{X} \} | = 2^{\mathcal{X}} \setminus \{ a \}
\]

\[
\Rightarrow (h, t) \models \neg a
\]

Proof of Proposition 6  It is enough to prove the proposition for the base case, since the rest follows directly by structural induction for each theory over \( \mathcal{A} \).

Let \( \Gamma \) be a theory over propositional atoms \( \mathcal{A} \) and \( \langle H, T \rangle \) an \( H_{LB} \) -interpretation over \( \mathcal{A} \). Let \( \tau(\Gamma) \) be a theory over atoms \( \{ p = t \mid p \in \mathcal{A} \} \) and \( \tau(H), \tau(T) \) and \( H_{LB} \)-valuation over \( \mathcal{A}, \{ t \} \). Then we have
\[
\langle H, T \rangle \models p
\]
\[
\iff \mathcal{X} \subseteq \{ p \} \mathcal{X} \land \mathcal{T} \subseteq \{ p \} \mathcal{T}
\]

\[
\tau(H) \in \{ p = t \} \mathcal{A} \land \tau(T) \in \{ p = t \} \mathcal{A}
\]

\[
\tau(H), \tau(T) \models p = t
\]

Proof of Proposition 7  It is enough to prove the proposition for the base case, since the rest follows directly by structural induction for each theory over \( \mathcal{A} \).

First, note that the pair \( \langle H, T \rangle \) over \( \mathcal{A} \), \( H = \mathcal{A}(\langle h, t \rangle) \) and \( T = \mathcal{A}(\langle t, t \rangle) \) is a well-formed \( H_{LB} \) -interpretation, since \( H \subseteq T \) holds by \( h \subseteq t \) and Proposition 3. Then we have
\[
\langle H, T \rangle \models a
\]

Proof of Proposition 8  Let \( \phi \) over \( \mathcal{A} \) be an arbitrary tautology in \( H_{LB} \). This means that for every \( H_{LB} \)-interpretation \( \langle H, T \rangle \) over \( \mathcal{A} \) holds \( (H, T) \models \phi \). Thus, we conclude that formula \( \phi' \) over \( \mathcal{A} \), \( \phi' \) obtained by replacing \( \text{atoms}(\phi) \) in a formula \( \phi \) by atoms of \( \mathcal{A} \) is a tautology as well. Since every \( H_{LB} \)-valuation \( \langle h, t \rangle \) over \( \mathcal{A} \), \( \mathcal{D} \) holds \( (h, t) \models \phi' \), since the semantics of the atoms may change the truth value of a single atom but can not affect the truth of the formula itself.

Proof of Proposition 9  It is enough to prove the proposition for the base case, since the rest follows directly by structural induction for each theory over \( \mathcal{A} \).

Let \( \Gamma \) be a theory over \( \mathcal{A} \), \( \langle h, t \rangle \) an \( H_{LB} \)-valuation over \( \mathcal{X}, \mathcal{D} \). Then, we have
\[
\langle h \rangle \models a
\]
\[
\iff \langle h \rangle \models a \land \langle t \rangle \models a
\]
\[
\iff \langle h \rangle \models a
\]

\[
\iff \langle h, t \rangle \models a
\]