Lower Bound Founded Logic of Here-and-There

Pedro Cabalar[1], Jorge Fandinno[2], Torsten Schaub[3], and Sebastian Schellhorn[3]

[1] University of Corunna, Spain
cabalar@udc.es
[2] University of Toulouse, France
jorge.fandinno@irit.fr
[3] 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 of looking 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 by Aziz et al 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 with lower bound founded variables along with its equilibrium models and elaborate upon its formal properties. Finally, we compare our approach with related ones and sketch future work.

1 Motivation

A distinguishing feature of Answer Set Programming (ASP; [6]) 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; [18]), whose equilibrium models correspond to stable models [23]. One way of looking 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. Thus, in analogy to [25, 1] foundedness in ASP can be understood by minimizing values of Boolean variables. This idea was generalized in [2] 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. Note that ASP follows the rationality principle, which says that one shall only believe in things one is forced to [16]. While this principle amounts to foundedness in the propositional case, there are at least two views for statements like $x \geq 42$. First, one may accept any value greater or equal than 42 for x. Second, one may only consider value 42 for x, unless there is a reason for a greater value. The latter one corresponds to the idea of foundedness in ASP.

The ASP literature contains several approaches dealing with atoms containing variables over non-Boolean domains [8, 19, 9] but these approaches do not address foundedness in our sense. For instance, approaches to Constraint ASP (CASP) like [8] only allow for atoms with variables over non-Boolean domains in the body of a rule. Thus, these atoms and the values of non-Boolean variables cannot be founded in terms of ASP.

Approaches like [19, 9] focus on foundedness on an atom level and allow for almost any kind of atoms in heads and bodies. These approaches match the view of the rationality principle that accepts any value satisfying a statement like $x \geq 42$. This permits atoms and variables over non-Boolean domains to be founded but the variables are not necessarily assigned to the smallest value that can be justified. The following examples point out the difference of the two views of the rationality principle. Moreover, we show that taking any value satisfying a statement as a rational choice together with separate minimization will not yield foundedness in terms of ASP. Consider the rules

$$x \geq 0 \qquad\qquad y \geq 0 \qquad\qquad x \geq 42 \leftarrow y < 42 \tag{1}$$

The approach presented in [9] produces the following result. The first two rules alone would generate any arbitrary pair of positive values for x and y, but the last rule further restricts $x \geq 42$ when the choice for y satisfies $y < 42$. It is clear that this last rule causes the range of x to depend on the value of y. Unfortunately, this dependence disappears if we try to minimize variable values a posteriori, that is, imposing a Pareto minimality criterion on the solutions. If we do so, we get a first minimal solution with $y \mapsto 0$ and $x \mapsto 42$ which somehow captures the expected intuition: we first decide the minimal value of y (which does not depend on x) assigning 0 to y and then apply the third rule to conclude $x \geq 42$ obtaining the minimal value 42 for x. However, among the solutions of (1) we also get those in which we chose $y \geq 42$ so the third rule is not applicable and $x \geq 0$. Therefore, we get a second Pareto-minimal solution with $y \mapsto 42$ and $x \mapsto 0$ that seems counterintuitive: as y does not depend on x there seems to be no reason to assign a minimal value other than 0 to y. To show that separate minimization on solutions not always yield all (and possibly more) solutions as expected by foundedness, let us consider the rules

$$x \geq 1 \qquad\qquad x \geq 42 \leftarrow \neg(x \leq 1) \tag{2}$$

In this case, depending on the assumption we make for $\neg(x \leq 1)$ we may get two possible founded solutions. If we assume that $x \leq 1$ is feasible, the second rule is disabled and the first rule $x \geq 1$ determines the founded minimal value 1 for x, still compatible with the assumption $x \leq 1$. If, on the contrary, we assume $\neg(x \leq 1)$, then the second rule imposes $x \geq 42$ determining the minimal value 42 for x that, again, confirms the assumption $\neg(x \leq 1)$. In other words, we expect two founded solutions with $x \mapsto 1$ and $x \mapsto 42$, respectively. In contrast, if we first apply [9] and then a Pareto minimization, we lose the solution with $x \mapsto 42$. This is because when assuming $x \leq 1$ we get $x \geq 1$ as before, and the only compatible solution assigns 1 to x, whereas if we assume $\neg(x \leq 1)$ we obtain infinite many values $x \geq 42$ compatible with the assumption. The solutions are then $x \mapsto 1$ plus the infinite sequence $x \mapsto 42$, $x \mapsto 43$ and so on. Thus, the unique Pareto minimal solution assigns 1 to x.

The previous approach addressing foundedness presented in [2] has a counter-intuitive behavior on rules like $p \leftarrow \neg p$. For instance, under [2] semantics, this rule alone yields a solution with p, although no solution is expected in ASP. We present in the following a logical reconstruction 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* ($\mathrm{HT_{LB}}$ for short) along with its equilibrium models. We elaborate upon the formal properties of $\mathrm{HT_{LB}}$ like persistence, negation and strong equivalence.[4] Furthermore, we point out the relation of $\mathrm{HT_{LB}}$ to HT, and show that our approach can alternatively be captured via a Ferraris-style definition of stable models [12] adapted to our setting. Finally, we compare our approach with related work and point out the benefits of $\mathrm{HT_{LB}}$.

2 Background

Let $\mathcal{A}$ be a set of propositional atoms. A formula φ is a combination of atoms by logical connectives $\bot$, $\wedge$, $\vee$, and $\leftarrow$. As usual, we define $\top \stackrel{\text{def}}{=} \bot \rightarrow \bot$ and $\neg\varphi \stackrel{\text{def}}{=} \varphi \rightarrow \bot$. 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, we rely upon denotations for fixing their semantics. A *denotation* of atoms in $\mathcal{A}$ is a function $[\![\cdot]\!]_{\mathcal{A}} : \mathcal{A} \rightarrow 2^{\mathcal{A}}$ mapping atoms in $\mathcal{A}$ to sets of interpretations over $\mathcal{A}$. Accordingly, $[\![p]\!]_{\mathcal{A}} \stackrel{\text{def}}{=} \{I \mid p \in I\}$ represents the set 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 φ a propositional formula over $\mathcal{A}$. Then, $\langle H, T \rangle$ satisfies φ, written $\langle H, T \rangle \models \varphi$, if the following conditions hold:*

1. $\langle H, T \rangle \not\models \bot$
2. $\langle H, T \rangle \models p$ *iff* $H \in [\![p]\!]_{\mathcal{A}}$ *for propositional atom* $p \in \mathcal{A}$
3. $\langle H, T \rangle \models \varphi_1 \wedge \varphi_2$ *iff* $\langle H, T \rangle \models \varphi_1$ *and* $\langle H, T \rangle \models \varphi_2$
4. $\langle H, T \rangle \models \varphi_1 \vee \varphi_2$ *iff* $\langle H, T \rangle \models \varphi_1$ *or* $\langle H, T \rangle \models \varphi_2$
5. $\langle H, T \rangle \models \varphi_1 \rightarrow \varphi_2$ *iff* $\langle I, T \rangle \not\models \varphi_1$ *or* $\langle I, T \rangle \models \varphi_2$ *for both* $I \in \{H, T\}$

As usual, we call $\langle H, T \rangle$ an HT-model of a theory Γ, if $\langle H, T \rangle \models \varphi$ for all φ in Γ. The usual definition of HT satisfaction (cf. [23]) is obtained by replacing Condition *2* above by

[4] Due to space limitations, we provide just a couple of proofs but give an extended version including all proofs at: www.cs.uni-potsdam.de/~seschell/JELIA19-paper-proofs.pdf.

2'. $\langle H, T\rangle \models p$ iff $p \in H$ for propositional atom $p \in \mathcal{A}$

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

Proposition 1. *Let $\langle H, T\rangle$ be an HT-interpretation and φ a formula over $\mathcal{A}$. Then, $\langle H, T\rangle \models \varphi$ iff $\langle H, T\rangle \models \varphi$ by replacing Condition 2 by 2'.*

As usual, an equilibrium model of a theory Γ is a (total) HT-interpretation $\langle T, T\rangle$ such that $\langle T, T\rangle \models \Gamma$ and there is no $H \subset T$ such that $\langle H, T\rangle \models \Gamma$. Then T is also called a stable model of Γ.

Let us recall some characteristic properties of HT. For HT-interpretations $\langle H, T\rangle$ and $\langle T, T\rangle$ and formula φ over $\mathcal{A}$ both $\langle H, T\rangle \models \varphi$ implies $\langle T, T\rangle \models \varphi$ (*persistence*) and $\langle H, T\rangle \models \varphi \to \bot$ iff $\langle T, T\rangle \not\models \varphi$ (*negation*) holds. Furthermore, $\Gamma_1 \cup \Gamma$ and $\Gamma_2 \cup \Gamma$ have the same stable models for theories Γ_1 and Γ_2 and any theory Γ over $\mathcal{A}$ iff Γ_1 and Γ_2 have the same HT-models (*strong equivalence*).

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 its formal properties.

HT_{LB} Properties. The language of HT_{LB} is defined over a set of atoms $\mathcal{A}_\mathcal{X}$ comprising variables, $\mathcal{X}$, and constants over an ordered domain $(\mathcal{D}, \succeq)$. For simplicity, we assume that each element of $\mathcal{D}$ is uniquely represented by a constant and abuse notation by using elements from $\mathcal{D}$ to refer to constants. Similarly, we identify $\succeq$ with its syntactic representative. The specific syntax of atoms is left open but assumed to refer to elements of $\mathcal{X}$ and $\mathcal{D}$. The only requirement is that we assume that an atom depends on a subset of variables in $\mathcal{X}$. An atom can be understood to hold or not once all variables in it are substituted by domain elements. Clearly, variables not occurring in an atom are understood as irrelevant for its evaluation. Examples of ordered domains are $(\{0, 1, 2, 3\}, \geq)$ and $(\mathbb{Z}, \geq)$, respectively; corresponding atoms are $x = y$ and $x \geq 42$. An example of a formula is '$y < 42 \wedge \neg(x = y) \to x \geq 42$'. We let $\mathit{vars}(\varphi) \subseteq \mathcal{X}$ be the set of variables and $\mathit{atoms}(\varphi) \subseteq \mathcal{A}_\mathcal{X}$ the atoms occurring in a formula φ.

For capturing partiality, we introduce a special domain element $\boldsymbol{u}$, standing for *undefined*, and extend $(\mathcal{D}, \succeq)$ to $(\mathcal{D}_{\boldsymbol{u}}, \succeq_{\boldsymbol{u}})$ where $\mathcal{D}_{\boldsymbol{u}} \stackrel{\text{def}}{=} \mathcal{D} \cup \{\boldsymbol{u}\}$ and $\succeq_{\boldsymbol{u}} \stackrel{\text{def}}{=} \succeq \cup \{(c, \boldsymbol{u}) \mid c \in \mathcal{D}\}$. With it, we define a (partial) *valuation* over $\mathcal{X}, \mathcal{D}$ as a function $v : \mathcal{X} \to \mathcal{D}_{\boldsymbol{u}}$ 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 $x \succeq c$ and add its downward closure $(x{\downarrow}c) \stackrel{\text{def}}{=} \{(x, d) \mid c, d \in \mathcal{D}, c \succeq d\}$. Given this, a valuation v is represented by the set $\bigcup_{v(x)=c, x \in \mathcal{X}} (x{\downarrow}c)$.[5] As an example, consider variables x and y over domain $(\{0, 1, 2, 3\}, \geq)$. The valuation $v = \{x \mapsto 2, y \mapsto 0\}$ can be represented by $v = (x \downarrow 2) \cup (y \downarrow 0) = \{(x, 0), (x, 1), (x, 2), (y, 0)\}$. Then,

[5] Note that $(x{\downarrow}\boldsymbol{u}) = \emptyset$, since $\boldsymbol{u} \notin \mathcal{D}$.

$v' = \{x \mapsto 1, y \mapsto \boldsymbol{u}\}$, viz. $\{(x,0),(x,1)\}$ in set notation, can be regarded as "smaller" than v because $v' \subseteq v$. The comparison of two valuations v and v' by set-inclusion $\subseteq$ amounts to a twofold comparison. That is, v and v' are compared regarding the occurrence of variables and their particular values wrt $\succeq$. We let $\mathfrak{V}_{\mathcal{X},\mathcal{D}}$ stand for the set of valuations over $\mathcal{X}$ and $\mathcal{D}$.

We define the satisfaction of formulas over $\mathcal{A}_\mathcal{X}$ wrt *atom denotations* over $\mathcal{X}, \mathcal{D}$, which are functions $[\![\cdot]\!]_{\mathcal{X},\mathcal{D}} : \mathcal{A}_\mathcal{X} \rightarrow 2^{\mathfrak{V}_{\mathcal{X},\mathcal{D}}}$ mapping atoms to sets of valuations. Let a be an atom of $\mathcal{A}_\mathcal{X}$ and $[\![a]\!]_{\mathcal{X},\mathcal{D}}$ its denotation. Then, $[\![a]\!]_{\mathcal{X},\mathcal{D}}$ is the set of valuations making a true. Since a depends on variables $vars(a) \subseteq \mathcal{X}$, we have for each $v \in [\![a]\!]_{\mathcal{X},\mathcal{D}}$ and valuation v' with $v(x) = v'(x)$ for $x \in vars(a)$ that $v' \in [\![a]\!]_{\mathcal{X},\mathcal{D}}$. Intuitively, values of $\mathcal{X} \setminus vars(a)$ can vary freely without changing the membership of a valuation to $[\![a]\!]_{\mathcal{X},\mathcal{D}}$. For simplicity, we drop indices $\mathcal{X}, \mathcal{D}$ 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:

$$[\![x \geq 42]\!] \stackrel{\text{def}}{=} \{v \mid v(x) \geq 42\} \qquad [\![42 \geq 0]\!] \stackrel{\text{def}}{=} \mathfrak{V} \qquad [\![0 \geq 42]\!] \stackrel{\text{def}}{=} \emptyset.$$

$[\![x \geq 42]\!]$ is the set of valuations assigning x to values greater or equal than 42 and all variables in $\mathcal{X} \setminus \{x\}$ take any value in $\mathcal{D}_{\boldsymbol{u}}$, eg $(x{\downarrow}45)$ and $(x{\downarrow}45) \cup (y{\downarrow}0)$ for $y \in \mathcal{X} \setminus \{x\}$ are possible valuations. Interestingly, atoms like $x \succeq x$ with $[\![x \succeq x]\!] = \{v \mid v(x) \neq \boldsymbol{u}\}$ force variables to be defined over $\mathcal{D}$ per definition of $\succeq$. A valuation v is defined for a set of variables $\mathcal{Y} \subseteq \mathcal{X}$ if $v(x) \neq \boldsymbol{u}$ for all $x \in \mathcal{Y}$.

We define an HT_{LB}-valuation over $\mathcal{X}, \mathcal{D}$ as a pair $\langle h, t \rangle$ of valuations over $\mathcal{X}, \mathcal{D}$ with $h \subseteq t$. We define satisfaction of formulas in HT_{LB}.

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

1. $\langle h, t \rangle \not\models \bot$
2. $\langle h, t \rangle \models a$ *iff* $v \in [\![a]\!]_{\mathcal{X},\mathcal{D}}$ *for atom* $a \in \mathcal{A}_\mathcal{X}$ *and for both* $v \in \{h, t\}$
3. $\langle h, t \rangle \models \varphi_1 \wedge \varphi_2$ *iff* $\langle h, t \rangle \models \varphi_1$ *and* $\langle h, t \rangle \models \varphi_2$
4. $\langle h, t \rangle \models \varphi_1 \vee \varphi_2$ *iff* $\langle h, t \rangle \models \varphi_1$ *or* $\langle h, t \rangle \models \varphi_2$
5. $\langle h, t \rangle \models \varphi_1 \rightarrow \varphi_2$ *iff* $\langle v, t \rangle \not\models \varphi_1$ *or* $\langle v, t \rangle \models \varphi_2$ *for both* $v \in \{h, t\}$

As usual, we call $\langle h, t \rangle$ an HT_{LB}-model of a theory Γ, if $\langle h, t \rangle \models \varphi$ for all φ in Γ. For a simple example, consider the theory containing atom $x \geq 42$ only. Then, every HT_{LB}-valuation $\langle h, t \rangle$ 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 later in this work when comparing to a Ferraris-like stable model semantics.

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 $\langle h, t \rangle$ and $\langle t, t \rangle$ be HT_{LB}-valuations over $\mathcal{X}, \mathcal{D}$, and φ be a formula over $\mathcal{A}_\mathcal{X}$. Then,*

1. $\langle h, t \rangle \models \varphi$ *implies* $\langle t, t \rangle \models \varphi$, *and*

2. $\langle h, t\rangle \models \varphi \rightarrow \bot$ *iff* $\langle t, t\rangle \not\models \varphi$.

Persistence implies that all atoms satisfied by $\langle h, t\rangle$ are also satisfied by $\langle t, t\rangle$. To make this precise, let $At(\langle h, t\rangle) \stackrel{\text{def}}{=} \{a \in \mathcal{A}_{\mathcal{X}} \mid h \in [\![\, a \,]\!] \text{ and } t \in [\![\, a \,]\!]\}$ be the set of atoms satisfied by $\langle h, t\rangle$.

Corollary 1. *Let* $\langle h, t\rangle$ *and* $\langle t, t\rangle$ *be* HT_{LB}*-valuations over* $\mathcal{X}, \mathcal{D}$. *Then,* $At(\langle h, t\rangle) \subseteq At(\langle t, t\rangle)$

Finally, we define an equilibrium model in $\mathrm{HT}_{\mathrm{LB}}$.

Definition 3. *An* HT_{LB}*-valuation* $\langle t, t\rangle$ *over* $\mathcal{X}, \mathcal{D}$ *is an* HT_{LB}*-equilibrium model of a theory* Γ *iff* $\langle t, t\rangle \models \Gamma$ *and there is no* $h \subset t$ *such that* $\langle h, t\rangle \models \Gamma$.

We refer to an $\mathrm{HT}_{\mathrm{LB}}$-equilibrium model $\langle t, t\rangle$ of Γ as an $\mathrm{HT}_{\mathrm{LB}}$-stable model t of Γ. Let us reconsider the theory containing atom $x \geq 42$ only. Then, $t = (x{\downarrow}42)$ is an $\mathrm{HT}_{\mathrm{LB}}$-stable model of $x \geq 42$, since $t \in [\![\, x \geq 42 \,]\!]$ and there is no $h \subset t$ with $h \in [\![\, x \geq 42 \,]\!]$. In contrast, neither $\mathrm{HT}_{\mathrm{LB}}$-model $\langle t', t'\rangle$ with $t' = (x{\downarrow}42) \cup (y{\downarrow}0)$ nor $\langle t'', t''\rangle$ with $t'' = (x{\downarrow}53)$ are $\mathrm{HT}_{\mathrm{LB}}$-stable models since t is a proper subset of both and $\langle t, t'\rangle \models x \geq 42$ as well as $\langle t, t''\rangle \models x \geq 42$ holds. Hence, $\mathrm{HT}_{\mathrm{LB}}$-stable models make sure that each variable is assigned to its smallest founded value.

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

Proposition 3 (Strong Equivalence). *Let* Γ_1, Γ_2 *and* Γ *be theories over* $\mathcal{A}_{\mathcal{X}}$. *Then, theories* $\Gamma_1 \cup \Gamma$ *and* $\Gamma_2 \cup \Gamma$ *have the same* HT_{LB}*-stable models for every theory* Γ *iff* Γ_1 *and* Γ_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 $\mathrm{HT}_{\mathrm{LB}}$-valuation that satisfies Γ_1 but not Γ_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 Γ 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 Γ. Let $h = (x{\downarrow}0)$ and $t = (x{\downarrow}2) \cup (y{\downarrow}0)$ be $\mathrm{HT}_{\mathrm{LB}}$-valuation over $\{x, y\}, \{0, 1, 2, 3\}$ with $\langle h, t\rangle \models \Gamma_1$ and $\langle h, t\rangle \not\models \Gamma_2$. For the first case assume that $\langle t, t\rangle \not\models \Gamma_2$. Since t cannot be a model of $\Gamma_2 \cup \Gamma$ by assumption, we construct Γ in a way that t is a stable model of $\Gamma_1 \cup \Gamma$. Hence, let $\Gamma = \{z \succeq c \mid (z, c) \in t\} = \{x \succeq 0, x \succeq 1, x \succeq 2, y \succeq 0\}$ be the theory with the only stable model t. By persistence of $\langle h, t\rangle$ wrt Γ_1 and construction of Γ 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 $\langle t, t\rangle \models \Gamma_2$. Now we construct Γ 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 $\langle h, t\rangle \models \Gamma_1$ and $\langle h, t\rangle \not\models \Gamma_2$ as well as $\langle t, t\rangle \models \Gamma_2$, thus we want to have $\langle h, t\rangle$ and $\langle v, v'\rangle$ with $t \subseteq v \subseteq v'$ as the only models of Γ. Hence, let $\Gamma = \Gamma' \cup \Gamma''$ with $\Gamma' = \{z \succeq c \mid (z, c) \in h\} = \{x \succeq 0\}$ the theory that is satisfied by everything that is greater or equal than h, and $\Gamma'' = \{z \succeq t(z) \rightarrow z' \succeq t(z'),\ z \succeq c \rightarrow z \succeq t(z) \mid (z, c), (z, t(z)), (z', t(z')) \in t \backslash h, z \neq z'\} = \{x \succeq 2 \rightarrow y \succeq 0,\ y \succeq 0 \rightarrow x \succeq 2,\ x \succeq 1 \rightarrow x \succeq 2,\ x \succeq 2 \rightarrow x \succeq 2\}$

the theory which derives values of t for each v'' with $h \subset v'' \subset t$. Since $\langle h, t\rangle \not\models \Gamma_2$ and by construction of Γ we get that t is a stable model of $\Gamma_2 \cup \Gamma$ but not of $\Gamma_1 \cup \Gamma$.

The following result shows that a formula $a \leftarrow \neg a$ has no stable model if a cannot be derived by some other formula.

Proposition 4. *Let Γ be a theory over $\mathcal{A}_{\mathcal{X}}$ containing a formula of form $a \leftarrow \neg a$ and for each HT_{LB}-stable model v of $\Gamma \setminus \{a \leftarrow \neg a\}$ over $\mathcal{X}, \mathcal{D}$ we have that $\langle v, v\rangle \not\models a$.*
Then, Γ has no HT_{LB}-stable model.

This proposition may seem to be trivial but we show in Section 4 that Aziz' original approach does not satisfy this property.

Negation in HT_{LB}. In the following, we elaborate on complements of atoms and their relation to negation, since $\mathcal{A}_{\mathcal{X}}$ may contain atoms like $x \geq 42$ and $x < 42$. Intuitively, complement of an atom holds whenever the atom itself does not hold. This can be easily expressed by using the atom denotation. More formally, the complement $\overline{a}$ of atom a is defined by its denotation $[\![\, \overline{a} \,]\!]_{\mathcal{X},\mathcal{D}} \stackrel{\text{def}}{=} 2^{\mathfrak{V}_{\mathcal{X},\mathcal{D}}} \setminus [\![\, a \,]\!]_{\mathcal{X},\mathcal{D}}$.

To illustrate that the simple complement of an atom is insufficient to yield something similar to strong negation let us take a closer look at propositional atoms in HT_{LB}. For mimicking Boolean truth values, we consider the domain $(\{\mathbf{t}, \mathbf{f}\}, \{\mathbf{t} \succeq \mathbf{f}\})$. Then, the denotation of propositional atoms in HT_{LB} can be defined as follows: $[\![\, p = \mathbf{t} \,]\!]_{\mathcal{A},\{\mathbf{t},\mathbf{f}\}} \stackrel{\text{def}}{=} \{v \mid v(p) = \mathbf{t}\}$ and $[\![\, p = \mathbf{f} \,]\!]_{\mathcal{A},\{\mathbf{t},\mathbf{f}\}} \stackrel{\text{def}}{=} \{v \mid v(p) = \mathbf{f}\}$. Note that $p = \mathbf{t}$ and $p = \mathbf{f}$ are regarded as strong negations of each other, as in the standard case [17]; their weak negations are given by $\neg(p = \mathbf{t})$ and $\neg(p = \mathbf{f})$, respectively. For instance, the complement $\overline{p = \mathbf{t}}$ is characterized by denotation $[\![\, \overline{p = \mathbf{t}} \,]\!]_{\mathcal{A},\{\mathbf{t},\mathbf{f}\}} = 2^{\mathfrak{V}_{\mathcal{A},\{\mathbf{t},\mathbf{f}\}}} \setminus [\![\, p = \mathbf{t} \,]\!]_{\mathcal{A},\{\mathbf{t},\mathbf{f}\}} = \{v \mid v(p) \neq \mathbf{t}\}$. Note that this complement allows for valuations v with $v(p) = \boldsymbol{u}$ which are not in $[\![\, p = \mathbf{f} \,]\!]_{\mathcal{A},\{\mathbf{t},\mathbf{f}\}}$.

Let us define another complement to exclude assigning undefined to variables of an atom. First, we define a denotation $[\![\, a \,]\!]_{\mathcal{X},\mathcal{D}}$ of an atom a as strict if each $v \in [\![\, a \,]\!]_{\mathcal{X},\mathcal{D}}$ is defined for $vars(a)$. Then, we characterize the strict complement $\overline{a}^s$ of atom a by the strict denotation $[\![\, \overline{a}^s \,]\!]_{\mathcal{X},\mathcal{D}} \stackrel{\text{def}}{=} 2^{\mathfrak{V}_{\mathcal{X},\mathcal{D}}} \setminus ([\![\, a \,]\!]_{\mathcal{X},\mathcal{D}} \cup \{v \mid v(x) = \boldsymbol{u}$ 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 $p = \mathbf{f}$ and $p = \mathbf{t}$ are strict complements of each other.

More generally, an atom with a 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 $[\![\, x \geq 42 \,]\!]_{\mathcal{X},\mathcal{D}} = \{v \mid v(x) \geq 42\}$. Then, its strict complement $\overline{x \geq 42}^s$ is defined by $[\![\, \overline{x \geq 42}^s \,]\!]_{\mathcal{X},\mathcal{D}} = \{v \mid \boldsymbol{u} \neq v(x) < 42\}$. As in the Boolean case, the strict complement $\overline{x \geq 42}^s$ can be seen as the strong negation of $x \geq 42$.

To make the relation of complements and negation precise, let us define entailment. A theory Γ over $\mathcal{A}_{\mathcal{X}}$ entails a formula φ over $\mathcal{A}_{\mathcal{X}}$, written $\Gamma \models \varphi$, when all HT_{LB}-models of Γ are HT_{LB}-models of φ. Then, we have the following result.

Proposition 5. *Let a be an atom over $\mathcal{A}_\mathcal{X}$, and $\overline{a}$ and $\overline{a}^s$ its complement and its strict complement over $\mathcal{A}_\mathcal{X}$, respectively.*
Then, $\{\overline{a}^s\} \models \overline{a}$ and $\{\overline{a}\} \models \neg a$.

This implies that the strict complement $\overline{a}^s$ of an atom a implies its negation $\neg a$, just as strong negation implies weak negation in the standard case [23]. To illustrate that in general the negation of an atom does not entail its complement, viz $\{\neg a\} \not\models \overline{a}$, consider atom $x \leq 42$ with strict denotation $[\![x \leq 42]\!]_{\mathcal{X},\mathcal{D}} = \{v \mid \boldsymbol{u} \neq v(x) \leq 42\}$. Then, its complement $\overline{x \leq 42}$ is defined by $[\![\overline{x \leq 42}]\!]_{\mathcal{X},\mathcal{D}} = 2^{\mathfrak{V}_{\mathcal{X},\mathcal{D}}} \setminus [\![x \leq 42]\!]_{\mathcal{X},\mathcal{D}} = \{v \mid v(x) = \boldsymbol{u} \text{ or } v(x) > 42\}$. For valuations $h = (x{\downarrow}42)$ and $t = (x{\downarrow}50)$, we have $\langle h, t\rangle \models \neg(x \leq 42)$ since $(x{\downarrow}50) \notin [\![x \leq 42]\!]_{\mathcal{X},\mathcal{D}}$. In contrast, $\langle h, t\rangle \not\models \overline{x \leq 42}$, since $(x{\downarrow}42) \notin [\![\overline{x \leq 42}]\!]_{\mathcal{X},\mathcal{D}}$. Thus, the complement $\overline{a}$ of an atom a can be seen as a kind of negation in between strong and weak negation.

HT_{LB} versus HT. Analogously to [9], we next show that HT can be seen as a special case of HT_{LB}.

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

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

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 Γ be a theory over $\mathcal{A}_\mathcal{X}$ and $\langle h, t\rangle$ an HT_{LB}-model of Γ over $\mathcal{X}, \mathcal{D}$.*
Then, $\langle At(\langle h, t\rangle), At(\langle t, t\rangle)\rangle$ is an HT-model of Γ over $\mathcal{A}_\mathcal{X}$.

That is, the collected atoms satisfied by an HT_{LB}-model of Γ can be seen as an HT-model of Γ by interpreting $\mathcal{A}_\mathcal{X}$ as propositional atoms. For instance, consider the theory containing only atom $x \neq y$ and its denotation $[\![x \neq y]\!] \stackrel{\text{def}}{=} \{v \mid \boldsymbol{u} \neq v(x) \neq v(y) \neq \boldsymbol{u}\}$. Let $h = (x{\downarrow}0) \cup (y{\downarrow}4)$ and $t = (x{\downarrow}0) \cup (y{\downarrow}42)$ be valuations and hence $At(\langle h, t\rangle) = At(\langle t, t\rangle) = \{x \neq y\}$ interpretations. Then, $\langle h, t\rangle \models x \neq y$ in HT_{LB} and $\langle At(\langle h, t\rangle), At(\langle t, t\rangle)\rangle \models x \neq y$ in HT.

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

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

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

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 $\mathrm{HT}_{\mathrm{LB}}$. To see this, consider an $\mathrm{HT}_{\mathrm{LB}}$-valuation $\langle h, t\rangle$ satisfying atom a in $\mathcal{A}_{\mathcal{X}}$. Hence, by persistence, $\mathrm{HT}_{\mathrm{LB}}$-valuation $\langle t, t\rangle$ satisfies a as well. However, this does not necessarily mean that $\mathrm{HT}_{\mathrm{LB}}$-valuations $\langle v, t\rangle$ with $h \subset v \subset t$ satisfy a. For instance, consider atom $x \neq 42$ with $\llbracket x \neq 42 \rrbracket \stackrel{\text{def}}{=} \{v \mid \boldsymbol{u} \neq v(x) \neq 42\}$ and valuations $h = (x{\downarrow}0)$ and $t = (x{\downarrow}53)$. Then, $\langle h, t\rangle \models x \neq 42$ and $\langle t, t\rangle \models x \neq 42$, but $\langle v, t\rangle \not\models x \neq 42$ for $v = (x{\downarrow}42)$ with $h \subset v \subset t$.

A question that arises now from the above is whether $\mathrm{HT}_{\mathrm{LB}}$ behaves in accordance with stable models semantics. To this end, we give straightforward definitions of classical satisfaction and the reduct by Ferraris [12] 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 φ a formula over $\mathcal{A}_{\mathcal{X}}$. Then, t satisfies φ, written $t \models_{cl} \varphi$, if the following holds:*

1. $t \not\models_{cl} \bot$
2. $t \models_{cl} a$ *iff* $t \in \llbracket a \rrbracket_{\mathcal{X},\mathcal{D}}$ *for atom* $a \in \mathcal{A}_{\mathcal{X}}$
3. $t \models_{cl} \varphi_1 \wedge \varphi_2$ *iff* $t \models_{cl} \varphi_1$ *and* $t \models_{cl} \varphi_2$
4. $t \models_{cl} \varphi_1 \vee \varphi_2$ *iff* $t \models_{cl} \varphi_1$ *or* $t \models_{cl} \varphi_2$
5. $t \models_{cl} \varphi_1 \rightarrow \varphi_2$ *iff* $t \not\models_{cl} \varphi_1$ *or* $t \models_{cl} \varphi_2$.

We call t a classical model of a theory Γ, if $t \models_{cl} \varphi$ for all φ in Γ. We define a Ferraris-like reduct wrt atoms $\mathcal{A}_{\mathcal{X}}$ as follows.

Definition 5. *Let φ be a formula over $\mathcal{A}_{\mathcal{X}}$ and t a valuation over $\mathcal{X}, \mathcal{D}$. Then, the reduct of φ wrt t, written φ^t, is defined by*

$$\varphi^t \stackrel{\text{def}}{=} \begin{cases} \bot & \text{if } t \not\models_{cl} \varphi \\ a & \text{if } t \models_{cl} \varphi \text{ and } \varphi = a \text{ is an atom in } \mathcal{A}_{\mathcal{X}} \\ {\varphi_1}^t \otimes {\varphi_2}^t & \text{if } t \models_{cl} \varphi \text{ and } \varphi = (\varphi_1 \otimes \varphi_2) \text{ for } \otimes \in \{\wedge, \vee, \rightarrow\} \end{cases}$$

For theory Γ and $\mathrm{HT}_{\mathrm{LB}}$-valuation t, we define $\Gamma^t \stackrel{\text{def}}{=} \{\varphi^t \mid \varphi \in \Gamma\}$. Note that in case of propositional atoms our reduct corresponds to Ferraris' reduct. We define a Ferraris-like stable model as expected.

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

Similar to the standard case [12], the next proposition shows that models in $\mathrm{HT}_{\mathrm{LB}}$ can be alternatively characterized in the style of Ferraris:

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

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

Corollary 2. *Let t be a valuation over $\mathcal{X}, \mathcal{D}$ and Γ a theory over $\mathcal{A}_{\mathcal{X}}$. Then, t is an HT_{LB}-stable model of Γ iff t is an Ferraris-like stable model of Γ.*

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

Modelling with Bound Founded Programs. In the following, we define programs over the class of linear constraint atoms to illustrate the modelling capabilities of HT_{LB} on an example.

We define a linear constraint atom over the ordered domain of integers $(\mathbb{Z}, \geq)$ by $\sum_{i=1}^{m} w_i x_i \prec k$ where $w_i, k \in \mathbb{Z}$ are constants, $x_i \in \mathcal{X}$ are distinct variables, and $\prec \in \{\geq, \leq, \neq, =\}$[6] is a binary relation. The denotation of a linear constraint atom is given by $[\![\sum_{i=1}^{m} w_i x_i \prec k]\!] \stackrel{\text{def}}{=} \{v \mid \sum_{i=1}^{m} w_i v(x_i) \prec k,\ v(x_i) \neq \boldsymbol{u}\}$. By $\mathcal{L}_{\mathcal{X}}$ we denote the set of linear constraint atoms over variables $\mathcal{X}$ and domain $(\mathbb{Z}, \geq)$.

A linear constraint atom a and its negation $\neg a$ over $\mathcal{L}_{\mathcal{X}}$ are called literals. A rule is a formula over $\mathcal{L}_{\mathcal{X}}$ of form

$$a_1 \vee \cdots \vee a_n \leftarrow l_1 \wedge \cdots \wedge l_{n'} \tag{3}$$

with a_i linear constraint atoms for $1 \leq i \leq n$ and l_j literals for $1 \leq j \leq n'$. A logic program is a theory over $\mathcal{L}_{\mathcal{X}}$ of rules of form (3).

For an example, consider the dependency of the revolutions per minute (rpm) of the engine of our car to its maximal range. As we know, the maximal range of a car decreases if we go with higher rpm; we need more fuel when choosing a smaller gear which increases the rpm assuming the same conditions like speed. For simplicity, we do not model gears, fuel or speed. Assume that our car needs at least 2000 rpm. Moreover, we know that our car has a range of at least 100 km. If we go by less than 4000 rpm, then our range is at least 200 km. Then, the following program P models the dependency of rpm and range without explicitly using negation or minimization:

$$\begin{aligned} &\text{rpm} \geq 2000 \\ &\text{range} \geq 100 \\ &\text{range} \geq 200 \leftarrow \text{rpm} < 4000 \end{aligned}$$

The HT_{LB}-stable model of P is $(\text{range} \downarrow 200) \cup (\text{rpm} \downarrow 2000)$, since 2000 is the minimal value satisfying rpm $\geq$ 2000 and thus rpm $<$ 4000 holds and make range $\geq$ 200 fire. For instance, if we extend P by the new statement rpm $\geq$ 4000, then we get HT_{LB}-stable model $(\text{range} \downarrow 100) \cup (\text{rpm} \downarrow 4000)$, since the minimal value derived by rpm $\geq$ 4000 does not make range $\geq$ 200 fire any more. Thus, 100 is the minimal value for range derived by range $\geq$ 100. Intuitively, it makes no sense to go by higher rpm and thus decrease the range if one is not forced to.

Note that this example behaves similar to the example in (1).

[6] 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.

4 Related Work

We start by comparing our approach to Aziz' Bound Founded ASP (BFASP; [2]), since both share the motivation to generalize 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. Informally, a constraint is increasing in a variable if the constraint is monotonic in this variable. Note that increasing is defined on constraints instead of atoms. For an example, the constraint $x \leq 42$ is not increasing in x, but the constraint $x \leq 42 \leftarrow y < 0$ is increasing in x over domain $\mathbb{N}$. 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 $\mathrm{HT_{LB}}$ assign variables to their smallest domain value per default. Interestingly, they differ in their understanding of smallest domain values. In $\mathrm{HT_{LB}}$, the smallest domain value is always the value 'undefined' to capture partiality, whereas in BFASP partiality is not considered if undefined is not explicitly part of the domain.

The value of a head variable is derived by the constraint even if it contains no implication. For instance, consider rule $(x + y \geq 42, x)$ over $\mathbb{N}$ in BFASP. Then, BFASP yields one stable model with $x \mapsto 42$ and $y \mapsto 0$. By default the value of y is 0, since y appears nowhere as an head. The value of x is derived from the value of $42 - y$. In contrast, $\mathrm{HT_{LB}}$ results in 43 stable models from $(x{\downarrow}0) \cup (y{\downarrow}42)$ to $(x{\downarrow}42) \cup (y{\downarrow}0)$ for theory $\{x + y \geq 42\}$. In $\mathrm{HT_{LB}}$, the variables of an (head) atom are treated in an equal way instead of an implicatory way by declaring one of them as head.

As already mentioned, BFASP does not satisfy Proposition 4. Rule $p \leftarrow \neg p$ has no stable model in ASP and $\mathrm{HT_{LB}}$, but BFASP yields a stable model containing p, since the BFASP reduct never replaces head variables and produce the rule as is and yield p is the minimal (and only) model of the rule.

Next, we compare $\mathrm{HT_{LB}}$ to the logic of HT with constraints ($\mathrm{HT_C}$; [9]).

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 $\mathrm{HT_C}$ follows the rationality principle by accepting any value satisfying an atom and thus foundedness is focused on atom level. Whereas, foundedness in $\mathrm{HT_{LB}}$ is focused on variable level by following the rationality principle accepting minimal values only. The latter is achieved by additionally comparing models wrt the values assigned to variables to determine equilibrium models. For instance, reconsider the fact $x \geq 42$ over $\{x\}, \mathbb{N}$ and valuations v and v' with $v(x) = 42$ and $v'(x) = 43$. Then, in $\mathrm{HT_C}$ we have $v \neq v'$, whereas in $\mathrm{HT_{LB}}$ we have $v \subset v'$. Hence, v and v' are solutions in $\mathrm{HT_C}$ but only v is a solution in $\mathrm{HT_{LB}}$. The theories in (1) and (2) of the motivation, show that the semantics of $\mathrm{HT_{LB}}$ cannot be obtained by adding separate minimization to $\mathrm{HT_C}$.

On the other hand, both $\mathrm{HT_{LB}}$ and $\mathrm{HT_C}$ define atomic satisfaction in terms of atom denotations. A difference is that in $\mathrm{HT_C}$ denotations need to be closed. 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 $\mathrm{HT_{LB}}$ this cannot be maintained, due to the additional comparison of valuations regarding values. The closure of denotions is significant to satisfy persistence in $\mathrm{HT_C}$. In contrast, in $\mathrm{HT_{LB}}$ persistence is maintained by forcing atomic satisfaction in both h and t, instead of h only as in $\mathrm{HT_C}$. The corresponding benefit is that this allows us to consider atoms in $\mathrm{HT_{LB}}$ which are not allowed in $\mathrm{HT_C}$, like $x \doteq y$ with $[\![\, x \doteq y \,]\!] \stackrel{\text{def}}{=} \{v \mid v(x) = v(y)\}$ which is not closed in $\mathrm{HT_C}$.

The integration of non-Boolean variables into ASP is also studied in the area of ASP modulo Theories [8, 15, 3, 11, 20, 22, 7, 4, 5, 21, 14, 19]. The common idea of these hybrid approaches is to integrate monotone theories, like constraint or linear programming, into the non-monotonic setting of ASP. Similar to $\mathrm{HT_C}$, foundedness is only achieved at the atomic level — if at all. In fact, many approaches avoid this entirely by limiting the occurrence of theory atoms to rule bodies.

Finally, logic programs with linear constraints under $\mathrm{HT_{LB}}$'s semantics amount to a non-monotonic counterpart of Integer Linear Programming (ILP; [24]). As a matter of fact, the monotonicity of ILP makes it hard to model default values and recursive concepts like reachability. It will be interesting future work to see whether $\mathrm{HT_{LB}}$ can provide a non-monotonic alternative to ILP.

5 Conclusion

We presented a new construction of the idea of foundedness over ordered domains in the setting of the logic of Here-and-There. We have shown that important properties like persistence, negation and strong equivalence hold in our approach. Furthermore, we pointed out that HT is a special case of $\mathrm{HT_{LB}}$. Also, we have shown that $\mathrm{HT_{LB}}$-stable models correspond to stable models according to a Ferraris'-like stable model semantics. We illustrated the modeling capabilities of $\mathrm{HT_{LB}}$ by an example representing the dependency of the rpm of a car and its range.

Finally, we compared our approach to related work to point out that foundedness is a non-trivial key feature of $\mathrm{HT_{LB}}$. We showed that $\mathrm{HT_{LB}}$ and BFASP have the same starting motivation but differ in their treatment of partiality. Furthermore, we pointed out that $\mathrm{HT_{LB}}$ can be seen as non-monotonic counterpart of monotonic theories.

We also think that $\mathrm{HT_{LB}}$ offers a new view of aggregates under Ferraris' semantics as atoms. In fact, sum aggregates are related to linear constraint atoms in $\mathrm{HT_{LB}}$. As we will show in a follow-up work, aggregates under Ferraris' semantics [13] can be represented by atoms in $\mathrm{HT_{LB}}$. This is interesting since then aggregates are no longer an extension of an existing approach, but rather integral atomic parts of $\mathrm{HT_{LB}}$. Hence, results shown in this work also apply to aggregates (under Ferraris' semantics) and provide a way to elaborate upon properties and relationships to other conceptions of aggregates. The view on aggregates as atoms provided by $\mathrm{HT_{LB}}$ may thus help us to better understand the differences among various aggregate semantics.

Appendix of Proofs

Proof of Proposition 2 It is enough to prove the proposition for the base case, since the rest follows directly by structural induction for each formula over $\mathcal{A}_{\mathcal{X}}$. Let $\langle h, t\rangle$ an $\mathrm{HT}_{\mathrm{LB}}$-valuation over $\mathcal{X}, \mathcal{D}$ and a atom of $\mathcal{A}_{\mathcal{X}}$.

First, we prove persistence, represented by *1* of the proposition. We have

$$\langle h, t\rangle \models a \Leftrightarrow h \in [\![\, a \,]\!] \wedge t \in [\![\, a \,]\!] \Rightarrow t \in [\![\, a \,]\!] \Leftrightarrow \langle t, t\rangle \models a$$

Subsequently, we prove negation, represented by *2* of the proposition. We have

$\langle h, t\rangle \models a \rightarrow \bot$
$\Leftrightarrow (\langle h, t\rangle \models \bot \vee \langle h, t\rangle \not\models a) \wedge (\langle t, t\rangle \models \bot \vee \langle t, t\rangle \not\models a)$
$\Leftrightarrow \langle h, t\rangle \not\models a \wedge \langle t, t\rangle \not\models a$
$\Leftrightarrow (h \notin [\![\, a \,]\!] \vee t \notin [\![\, a \,]\!]) \wedge (t \notin [\![\, a \,]\!])$
$\Leftrightarrow \langle t, t\rangle \not\models a$ □

Proof of Proposition 5 Let a be an atom over $\mathcal{A}_{\mathcal{X}}$, and $\overline{a}$ and $\overline{a}^s$ its complement and its strict complement over $\mathcal{A}_{\mathcal{X}}$, respectively.

First, we prove $\overline{a}^s \models \overline{a}$. For any $\mathrm{HT}_{\mathrm{LB}}$-valuation $\langle h, t\rangle$ over $\mathcal{X}, \mathcal{D}$ we have
$\langle h, t\rangle \models \overline{a}^s$
$\Leftrightarrow h \in [\![\, \overline{a}^s \,]\!] \wedge t \in [\![\, \overline{a}^s \,]\!]$ with $[\![\, \overline{a}^s \,]\!] = 2^{\mathfrak{V}} \setminus ([\![\, a \,]\!] \cup \{v \mid v(x) = \boldsymbol{u} \text{ for some } x \in \mathit{vars}(a)\})$
$\Rightarrow h \in 2^{\mathfrak{V}} \setminus [\![\, a \,]\!] \wedge t \in 2^{\mathfrak{V}} \setminus [\![\, a \,]\!]$
$\Leftrightarrow \langle h, t\rangle \models \overline{a}$

Secondly, we prove $\overline{a} \models \neg a$. For any $\mathrm{HT}_{\mathrm{LB}}$-valuation $\langle h, t\rangle$ over $\mathcal{X}, \mathcal{D}$ we have
$\langle h, t\rangle \models \overline{a}$
$\Leftrightarrow h \in [\![\, \overline{a} \,]\!] \wedge t \in [\![\, \overline{a} \,]\!]$ with $[\![\, \overline{a} \,]\!] = 2^{\mathfrak{V}} \setminus [\![\, a \,]\!]$
$\Leftrightarrow h \notin [\![\, a \,]\!] \wedge t \notin [\![\, a \,]\!]$
$\Rightarrow t \notin [\![\, a \,]\!]$
$\overset{\text{Proposition 2}}{\Leftrightarrow} \langle h, t\rangle \models \neg a$ □

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 $\mathcal{A}_{\mathcal{X}}$.

Let Γ be a theory over $\mathcal{A}_{\mathcal{X}}$ and $\langle h, t\rangle$ an $\mathrm{HT}_{\mathrm{LB}}$-valuation over $\mathcal{X}, \mathcal{D}$. Then, we have
$h \models_{cl} a^t$
$\Leftrightarrow h \models_{cl} a \wedge t \models_{cl} a$
$\Leftrightarrow h \in [\![\, a \,]\!] \wedge t \in [\![\, a \,]\!]$
$\Leftrightarrow \langle h, t\rangle \models a$ □

References

1. M. Alviano, F. Calimeri, W. Faber, N. Leone, and S. Perri. Unfounded sets and well-founded semantics of answer set programs with aggregates. *Journal of Artificial Intelligence Research*, 42:487–527, 2011.

2. R. Aziz. *Answer Set Programming: Founded Bounds and Model Counting*. PhD thesis, University of Melbourne, 2015.
3. M. Balduccini. Representing constraint satisfaction problems in answer set programming. In W. Faber and J. Lee, editors, *Proceedings of the Second Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP'09)*, pages 16–30, 2009.
4. M. Banbara, M. Gebser, K. Inoue, M. Ostrowski, A. Peano, T. Schaub, T. Soh, N. Tamura, and M. Weise. aspartame: Solving constraint satisfaction problems with answer set programming. In F. Calimeri, G. Ianni, and M. Truszczyński, editors, *Proceedings of the Thirteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'15)*, volume 9345 of *Lecture Notes in Artificial Intelligence*, pages 112–126. Springer-Verlag, 2015.
5. M. Banbara, B. Kaufmann, M. Ostrowski, and T. Schaub. Clingcon: The next generation. *Theory and Practice of Logic Programming*, 17(4):408–461, 2017.
6. C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2003.
7. M. Bartholomew and J. Lee. System aspmt2smt: Computing ASPMT theories by SMT solvers. In E. Fermé and J. Leite, editors, *Proceedings of the Fourteenth European Conference on Logics in Artificial Intelligence (JELIA'14)*, volume 8761 of *Lecture Notes in Artificial Intelligence*, pages 529–542. Springer-Verlag, 2014.
8. S. Baselice, P. Bonatti, and M. Gelfond. Towards an integration of answer set and constraint solving. In M. Gabbrielli and G. Gupta, editors, *Proceedings of the Twenty-first International Conference on Logic Programming (ICLP'05)*, volume 3668 of *Lecture Notes in Computer Science*, pages 52–66. Springer-Verlag, 2005.
9. P. Cabalar, R. Kaminski, M. Ostrowski, and T. Schaub. An ASP semantics for default reasoning with constraints. In R. Kambhampati, editor, *Proceedings of the Twenty-fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*, pages 1015–1021. IJCAI/AAAI Press, 2016.
10. M. Carro and A. King, editors. *Technical Communications of the Thirty-second International Conference on Logic Programming (ICLP'16)*, volume 52. Open Access Series in Informatics (OASIcs), 2016.
11. C. Drescher and T. Walsh. A translational approach to constraint answer set solving. *Theory and Practice of Logic Programming*, 10(4-6):465–480, 2010.
12. P. Ferraris. Answer sets for propositional theories. In C. Baral, G. Greco, N. Leone, and G. Terracina, editors, *Proceedings of the Eighth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'05)*, volume 3662 of *Lecture Notes in Artificial Intelligence*, pages 119–131. Springer-Verlag, 2005.
13. P. Ferraris. Logic programs with propositional connectives and aggregates. *ACM Transactions on Computational Logic*, 12(4):25, 2011.
14. M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, and P. Wanko. Theory solving made easy with clingo 5. In Carro and King [10], pages 2:1–2:15.
15. M. Gebser, M. Ostrowski, and T. Schaub. Constraint answer set solving. In P. Hill and D. Warren, editors, *Proceedings of the Twenty-fifth International Conference on Logic Programming (ICLP'09)*, volume 5649 of *Lecture Notes in Computer Science*, pages 235–249. Springer-Verlag, 2009.
16. M. Gelfond and Y. Kahl. *Knowledge Representation, Reasoning, and the Design of Intelligent Agents: The Answer-Set Programming Approach*. Cambridge University Press, 2014.
17. M. Gelfond and V. Lifschitz. Logic programs with classical negation. In D. Warren and P. Szeredi, editors, *Proceedings of the Seventh International Conference on Logic Programming (ICLP'90)*, pages 579–597. MIT Press, 1990.

18. A. Heyting. Die formalen Regeln der intuitionistischen Logik. In *Sitzungsberichte der Preussischen Akademie der Wissenschaften*, page 42–56. Deutsche Akademie der Wissenschaften zu Berlin, 1930. Reprint in Logik-Texte: Kommentierte Auswahl zur Geschichte der Modernen Logik, Akademie-Verlag, 1986.
19. T. Janhunen, R. Kaminski, M. Ostrowski, T. Schaub, S. Schellhorn, and P. Wanko. Clingo goes linear constraints over reals and integers. *Theory and Practice of Logic Programming*, 17(5-6):872–888, 2017.
20. T. Janhunen, G. Liu, and I. Niemelä. Tight integration of non-ground answer set programming and satisfiability modulo theories. In P. Cabalar, D. Mitchell, D. Pearce, and E. Ternovska, editors, *Proceedings of the First Workshop on Grounding and Transformation for Theories with Variables (GTTV'11)*, pages 1–13, 2011.
21. Y. Lierler and B. Susman. SMT-based constraint answer set solver EZSMT (system description). In Carro and King [10], pages 1:1–1:15.
22. G. Liu, T. Janhunen, and I. Niemelä. Answer set programming via mixed integer programming. In G. Brewka, T. Eiter, and S. McIlraith, editors, *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning (KR'12)*, pages 32–42. AAAI Press, 2012.
23. D. Pearce. Equilibrium logic. *Annals of Mathematics and Artificial Intelligence*, 47(1-2):3–41, 2006.
24. A. Schrijver. *Theory of linear and integer programming*. Discrete mathematics and optimization. John Wiley & sons, 1999.
25. A. Van Gelder, K. Ross, and J. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.