Recursive Aggregates as Intensional Functions in Answer Set Programming: Semantics and Strong Equivalence

Jorge Fandinno and Zachary Hansen

Abstract

This paper shows that the semantics of programs with aggregates implemented by the solvers `clingo` and `dlv` can be characterized as extended First-Order formulas with intensional functions in the logic of Here-and-There. Furthermore, this characterization can be used to study the strong equivalence of programs with aggregates under either semantics. We also present a transformation that reduces the task of checking strong equivalence to reasoning in classical First-Order logic, which serves as a foundation for automating this procedure.

Introduction

Answer set programming (ASP) is a form of declarative programming well-suited for solving knowledge-intensive search and optimization problems (Lifschitz 2019). Its success relies on the combination of a rich knowledge representation language with efficient solvers. Some of the its most useful constructs are *aggregates*: intuitively, these are functions that apply to sets. The semantics of aggregates have been extensively studied in the literature (Simons, Niemelä, and Soininen 2002; Dovier, Pontelli, and Rossi 2003; Pelov, Denecker, and Bruynooghe 2007; Son and Pontelli 2007; Ferraris 2011; Faber, Pfeifer, and Leone 2011; Gelfond and Zhang 2014, 2019; Cabalar et al. 2019). In most cases, they rely on the idea of *grounding*—a process in which all variables are replaced by variable-free terms. This makes reasoning about programs with variables cumbersome and it does not allow the use of First-Order (FO) theorem provers for verifying properties (Fandinno et al. 2020).

Though several approaches describe the semantics of aggregates bypassing the need for grounding, most of these approaches only allow a restricted class of aggregates (Lee, Lifschitz, and Palla 2008; Lifschitz 2022) or use some extension of the logical language (Bartholomew, Lee, and Meng 2011; Lee and Meng 2012; Asuncion et al. 2015; Cabalar et al. 2018). Recently, Fandinno, Hansen, and Lierler (2022) showed how to translate logic programs with aggregates into first-formulas, which after the application of the SM operator (Ferraris, Lee, and Lifschitz 2011), capture the `ASP-Core-2` semantics. Though most practical problems can be represented within the restrictions of the `ASP-Core-2` semantics, some notable exceptions are more naturally represented using recursive aggregates, which is not allowed by `ASP-Core-2`. One of these examples is the *Company Control problem*, which consists of finding companies that control other companies by (directly or indirectly) owning a majority of their shares. This problem has been encoded in the literature using the following logic program (Pelov, Denecker, and Bruynooghe 2007; Faber, Pfeifer, and Leone 2011; Mumick, Pirahesh, and Ramakrishnan 1990; Kemp and Stuckey 1991; Ross and Sagiv 1997):

```
ctrStk(C1,C1,C2,P) :- ownsStk(C1,C2,P).                (1)
ctrStk(C1,C2,C3,P) :- controls(C1,C2),
                      ownsStk(C2,C3,P).                (2)
controls(C1,C3) :- company(C1), company(C3),
      #sum{P,C2 : ctrStk(C1,C2,C3,P)} > 50.            (3)
```

where atom `ownsStk(C1,C2,P)` means that company `C1` directly owns `P%` of the shares of company `C2`; `ctrStk(C1,C2,C3,P)` means that company `C1` controls `P%` of the shares of company `C3` through company `C2` that it controls; and `controls(C1,C3)` means that company `C1` controls company `C3`. Another area where allowing recursive aggregates is important is in the study of *strong equivalence* (Lifschitz, Pearce, and Valverde 2001, 2007). The strong equivalence problem consists of determining whether two programs have the same behavior in any context. Even if the programs we are analyzing do not contain recursion, adding some context may introduce it.

In this paper, we show that the translation introduced by Fandinno, Hansen, and Lierler can also be used for programs with recursive aggregates if we interpret functions in an intensional way (Lin and Wang 2008; Cabalar 2011; Lifschitz 2012; Balduccini 2013; Bartholomew and Lee 2019). We focus on the Abstract Gringo (Gebser et al. 2015) generalization of the semantics by Ferraris (2011), which is used in the answer set solver `clingo`, and the semantics by Faber, Pfeifer, and Leone (2011), which is used in the answer set solver `dlv`. We prove that the translation introduced by Fandinno, Hansen, and Lierler coincides with the Abstract Gringo semantics when we interpret the function symbols representing sets according to the semantics for intensional functions by Bartholomew and Lee. For `dlv`, we introduce a similar translation, which uses a second form of negation. We show that we can use these translations to state the strong equivalence of the two programs and how to reduce this problem to reasoning in classical FO logic.

Preliminaries

We start by reviewing the syntax of programs with aggregates and presenting an extension of the logic of Quantified Here-and-There (Pearce and Valverde 2008) with intensional functions that is suited for programs with aggregates.

Syntax of programs with aggregates. We follow here the presentation by Fandinno, Hansen, and Lierler (2022). We assume a *(program) signature* with three countably infinite sets of symbols: *numerals*, *symbolic constants* and *program variables*. We also assume a 1-to-1 correspondence between numerals and integers; the numeral corresponding to an integer n is denoted by $\overline{n}$. *Program terms* are either numerals, symbolic constants, variables or one of the special symbols *inf* and *sup*. A program term (or any other expression) is *ground* if it contains no variables. We assume that a total order on ground terms is chosen such that

- *inf* is its least element and *sup* is its greatest element,
- for any integers m and n, $\overline{m} < \overline{n}$ iff $m < n$. , and
- for any integer n and any symbolic constant c, $\overline{n} < c$.

An *atom* is an expression of the form $p(\mathbf{t})$, where p is a symbolic constant and $\mathbf{t}$ is a list of program terms. A *comparison* is an expression of the form $t \prec t'$, where t and t' are program terms and $\prec$ is one of the *comparison symbols*:

$$= \quad \neq \quad < \quad > \quad \leq \quad \geq \tag{4}$$

An *atomic formula* is either an atom or a comparison. A *basic literal* is an atomic formula possibly preceded by one or two occurrences of *not*. An *aggregate element* has the form

$$t_1, \ldots, t_k : l_1, \ldots, l_m \tag{5}$$

where each t_i ($1 \leq i \leq k$) is a program term and each l_i ($1 \leq i \leq m$) is a basic literal. An *aggregate atom* is of form $\#\texttt{op}\{E\} \prec u$ where $\texttt{op}$ is an operation name, E is an aggregate element, $\prec$ is one of the comparison symbols in (1), and u is a program term, called *guard*. We consider operation names `count` and `sum`. For example, the expression

$$\#\texttt{sum}\{\texttt{P,C2 : ctrStk(C1,C2,C3,P)}\} > \texttt{50}$$

in the body of rule (3) is an aggregate atom. An *aggregate literal* is an aggregate atom possibly preceded by one or two occurrences of *not*. A *literal* is either a basic literal or an aggregate literal. A *rule* is an expression of the form

$$\mathit{Head} \text{ :- } B_1, \ldots, B_n, \tag{6}$$

where

- *Head* is either an atom or symbol $\bot$; we often omit symbol $\bot$ which results in an empty head;
- each B_i ($1 \leq i \leq n$) is a literal.

We call the symbol :- a *rule operator*. We call the left hand side of the rule operator the *head*, the right hand side of the rule operator the *body*. When the head of the rule is an atom we call the rule *normal*, and when it is the symbol $\bot$ we call it a *constraint*. When the body of a normal rule is empty, we call the rule a *fact*. A *program* is a set of rules.

Each operation name $\texttt{op}$ is associated with a function $\widehat{\texttt{op}}$ that maps every set of tuples of ground terms to a ground term. If the first member of a tuple $\mathbf{t}$ is a numeral $\overline{n}$ then we say that integer n is the weight of $\mathbf{t}$, otherwise the weight of $\mathbf{t}$ is 0. For any set Δ of tuples of ground terms,

- $\widehat{\texttt{count}}(\Delta)$ is the numeral corresponding to the cardinality of Δ, if Δ is finite; and *sup* otherwise.
- $\widehat{\texttt{sum}}(\Delta)$ is the numeral corresponding to the sum of the weights of all tuples in Δ, if Δ contains finitely many tuples with non-zero weights; and 0 otherwise.[1] If Δ is empty, then $\widehat{\texttt{sum}}(\Delta) = 0$.

Many-sorted logic and extended FO formulas. A many-sorted signature consists of symbols of three kinds—*sorts*, *function constants*, and *predicate constants*. A reflexive and transitive *subsort* relation is defined on the set of sorts. A tuple $s_1, \ldots, s_n$ ($n \geq 0$) of *argument sorts* is assigned to every function constant and to every predicate constant; in addition, a *value sort* is assigned to every function constant. Function constants with $n = 0$ are called *object constants*. For every sort, we assume an infinite sequence of *object variables* of that sort is chosen. *Terms* and *atomic formulas* over a (many-sorted) signature σ are defined as usual with the additional restriction that the sort of a term is a subsort of the sort of the function or predicate constant it is an argument of. *Extended First-Order formulas* over σ are formed from atomic formulas and the 0-place connective $\bot$ (falsity) using the unary connective $\llcorner$, the binary connectives $\wedge$, $\vee$, $\rightarrow$ and the quantifiers $\forall$, $\exists$. We define the usual abbreviations: $\neg F$ stands for $F \rightarrow \bot$ and $F \leftrightarrow G$ stands for $(F \rightarrow G) \wedge (G \rightarrow F)$. We have two negation symbols ($\neg$ and $\llcorner$) and both correspond to classical negation in the context of classical FO logic. The symbol $\neg$ is used to represent the standard negation in the logic of Here-and-There and corresponds to default negation in logic programs under the `clingo` semantics, while symbol $\llcorner$ is a new connective and it is used to capture default negation under the semantics of `dlv`. *Interpretation*, *sentence*, and *theory*, *satisfaction* and *models* are defined as usual with the additional condtion that $I \models \llcorner F$ iff $I \not\models F$. A *standard* FO formula (resp. sentence, theory) is a formula (resp. sentence, theory) without the new operator $\llcorner$.

Stable Model Semantics with Intensional Functions. Let I and H be two interpretations of a signature σ and $\mathcal{P}$ and $\mathcal{F}$ respectively be sets of predicate and function constants of σ. We write $H \leq^{\mathcal{PF}} I$ if

- H and I have the same universe for all sorts;
- $p^H \subseteq p^I$ for every predicate constant p in $\mathcal{P}$ and $p^H = p^I$ for every predicate constant p not in $\mathcal{P}$; and
- $f^H = f^I$ for every function constant f not in $\mathcal{F}$.

[1]The sum of a set of integers is not always defined. We could choose a special symbol to denote this case, we chose to use 0 following the description of Abstract Gringo (Gebser et al. 2015).

Sets $\mathcal{P}$ and $\mathcal{F}$ respectively contain predicate and function constant of σ, that we consider *intensional*, that is, those that are defined by the theory. An *ht-interpretation* of σ is a pair $\langle H, I\rangle$, where H and I are interpretation of σ such that $H \leq^{\mathcal{PF}} I$. (In terms of many-sorted Kripke models, I is the there-world, and H is the here-world). The satisfaction relation $\models_{ht}$ between HT-interpretation $\langle H, I\rangle$ of σ and a sentence F over σ^I is defined recursively as follows:

- $\langle H, I\rangle \models_{ht} p(\mathbf{t})$, if $I \models p(\mathbf{t})$ and $H \models p(\mathbf{t})$;
- $\langle H, I\rangle \models_{ht} t_1 = t_2$ if $t_1^I = t_2^I$ and $t_1^H = t_2^H$;
- $\langle H, I\rangle \models_{ht} \neg F$ if both $I \not\models F$ and $H \not\models F$;
- $\langle H, I\rangle \models_{ht} F \wedge G$ if $\langle H, I\rangle \models_{ht} F$ and $\langle H, I\rangle \models_{ht} G$;
- $\langle H, I\rangle \models_{ht} F \vee G$ if $\langle H, I\rangle \models_{ht} F$ or $\langle H, I\rangle \models_{ht} G$;
- $\langle H, I\rangle \models_{ht} F \rightarrow G$ if $I \models F \rightarrow G$, and $\langle H, I\rangle \not\models_{ht} F$ or $\langle H, I\rangle \models_{ht} G$,
- $\langle H, I\rangle \models_{ht} \forall X\, F(X)$ if $\langle H, I\rangle \models_{ht} F(d^*)$ for each $d \in |I|^s$, where s is the sort of X;
- $\langle H, I\rangle \models_{ht} \exists X\, F(X)$ if $\langle H, I\rangle \models_{ht} F(d^*)$ for some $d \in |I|^s$, where s is the sort of X.

If $\langle H, I\rangle \models_{ht} F$ holds, we say that $\langle H, I\rangle$ *ht-satisfies* F and that $\langle H, I\rangle$ is an *ht-model* of F. If it is clear from the context that the $\models_{ht}$ entailment relation is referred to, we will simply say that $\langle H, I\rangle$ *satisfies* F. We say that an ht-interpretation $\langle H, I\rangle$ *satisfies* a set of sentences Γ if it satisfies every sentence F in Γ.

We write $H <^{\mathcal{PF}} I$ if $H \leq^{\mathcal{PF}} I$ and $H \neq I$. A model I of a set of sentences Γ is called *stable* if there is no $H <^{\mathcal{PF}} I$ such that $\langle H, I\rangle$ satisfies Γ. If we consider finite standard theories, this definition of stable models coincides with the definition of stable models by Bartholomew and Lee (2019). If we consider (possibly infinite) standard theories without intensional functions each stable model I corresponds to the equilibrium model $\langle I, I\rangle$ by Pearce and Valverde (2008).

Logic Programs With Aggregates as Extended Many-Sorted First-Order Sentences

We present here translations τ^{cli} and τ^{dlv} that turn a program into extended FO sentences with equality over a signature $\sigma(\mathcal{P}, \mathcal{S})$ of *three sorts*; $\mathcal{P}$ and $\mathcal{S}$ are sets of predicate and *set symbols*, respectively. Superscripts *cli* and *dlv* refer to the semantics of `clingo` and `dlv`, respectively.

Target Signature. A *set symbol* is a pair $E/\mathbf{X}$, where E is an aggregate element and $\mathbf{X}$ is a list of variables occurring in E. For brevity's sake, each set symbol $E/\mathbf{X}$ is assigned a short name $|E/\mathbf{X}|$. The target signature is of three sorts. The first sort is called the *general sort* (denoted s_{gen}); all program terms are of this sort. The second sort is called the *tuple sort* (denoted s_{tuple}); it contains entities that are *tuples* of objects of the general sort. The third sort is called the *set sort* (denoted s_{set}); it contains entities that are *sets* of elements of the second sort, that is, tuples of objects of the general sort. Signature $\sigma(\mathcal{P}, \mathcal{S})$ contains:

1. all ground terms as object constants of the general sort;
2. all predicate symbols in $\mathcal{P}$ with all arguments of general sort;
3. comparison symbols other than equality as binary predicate symbols whose arguments are of the general sort;
4. predicate constant $\in/2$ with the first argument of the sort tuple and the second argument of the sort set;
5. function constant $tuple/k$ with arguments of the general sort and value of the tuple sort for each set symbol $E/\mathbf{X}$ in $\mathcal{S}$ with E of the form of (5);
6. unary function constants $count$ and sum whose argument is of the set sort and its value is of the general sort;
7. for each set symbol $E/\mathbf{X}$ in $\mathcal{S}$ where n is the number of variables in $\mathbf{X}$, function constants $s^{cli}_{|E/\mathbf{X}|}$ and $s^{dlv}_{|E/\mathbf{X}|}$ with n arguments of the general sort and whose value is of the set sort;

We use infix notation in constructing atoms that utilize predicate symbols of comparisons ($>, \geq, <, \leq, \neq$) and the set membership predicate $\in$. Function $tuple(t_1, \ldots, t_k)$ is a constructor for the k-tuple containing program terms $t_1, \ldots, t_k$. Function constants $s^{cli}_{|E/\mathbf{X}|}$ and $s^{dlv}_{E/\mathbf{X}}$ are used to represent sets occurring in aggregates for the `clingo` and `dlv` semantics, respectively. Each of these function constants maps an n-tuple of ground terms $\mathbf{x}$ to the set of tuples represented[2] by $E^{\mathbf{X}}_{\mathbf{x}}$. The result of $count$ is the cardinality of the set passed as an argument; the result of sum is the sum of the weights of all elements in the set passed as an argument. These claims are formalized below.

About a predicate symbol p/n, we say that is *occurs* in a program Π if there is an atom of the form $p(t_1, \ldots, t_n)$ in Π. For set symbols, we need to introduce first the concepts of global variable and set symbol. A variable is said to be *global* in a rule if

1. it occurs in any non-aggregate literal, or
2. it occurs in a guard of any aggregate literal.

We say that set symbol $E/\mathbf{X}$ occurs in rule R if this rule contains an aggregate literal with the aggregate element E and $\mathbf{X}$ is the list of all variables in E that are global in R. We say that $E/\mathbf{X}$ occurs in a program Π if $E/\mathbf{X}$ occurs in some rule of the program. For instance, in rule (3) the global variables are `C1` and `C3`. Set symbol $E_{ctr}/\mathbf{X}_{ctr}$ occurs in this rule where E_{ctr} stands for the aggregate element `P,C2 : ctrStk(C1,C2,C3,P)` and $\mathbf{X}_{ctr}$ is the list of variables `C1,C3`. In the following, we denote by $s^{cli}_{ctr}/2$ and $s^{dlv}_{ctr}/2$ the function symbols associated with this set symbol for the `clingo` and `dlv` semantics, respectively.

When discussing a program Π, we assume a signature $\sigma(\mathcal{P}, \mathcal{S})$ such that $\mathcal{P}$ and $\mathcal{S}$ are the sets that contain all predicate symbols and all set symbols occurring in Π, respectively. When clear by the context, we write σ instead of $\sigma(\mathcal{P}, \mathcal{S})$. We also assume that $\mathcal{P}$ is the set of intensional predicates and that the set of intensional functions is the set of all function symbols corresponding to set symbols in $\mathcal{S}$.

Translations. We now describe translations that convert a program into a set of extended FO sentences. We use $\tau^x_{\mathbf{Z}}$

[2]For a tuple $\mathbf{X}$ of distinct variables, a tuple $\mathbf{x}$ of ground terms of the same length as $\mathbf{X}$, and an expression α, by $\alpha^{\mathbf{X}}_{\mathbf{x}}$ we denote the expression obtained from α by substituting $\mathbf{x}$ for $\mathbf{X}$.

and τ^x to denote the rules that are common when x is replaced by either *cli* or *dlv*. Given a list $\mathbf{Z}$ of global variables in some rule R, we define $\tau^{cli}_{\mathbf{Z}}$ and $\tau^{dlv}_{\mathbf{Z}}$ for all elements of R as follows.

1. for every atomic formula A occurring outside of an aggregate literal, its translation $\tau^x_{\mathbf{Z}} A$ is A itself; $\tau^x_{\mathbf{Z}}\bot$ is $\bot$;
2. for an aggregate atom A of form `#count`$\{E\} \prec u$ or `#sum`$\{E\} \prec u$, its translation $\tau^x_{\mathbf{Z}}$ is the atom
$$count(s^x_{|E/\mathbf{X}|}(\mathbf{X})) \prec u \text{ or } sum(s^x_{|E/\mathbf{X}|}(\mathbf{X})) \prec u$$
respectively, where $\mathbf{X}$ is the list of variables in $\mathbf{Z}$ occurring in E;
3. for every (basic or aggregate) literal of the form *not* A its translation are $\tau^{cli}_{\mathbf{Z}}(not\ A)$ is $\neg\tau^{cli}_{\mathbf{Z}} A$ and $\tau^{dlv}_{\mathbf{Z}}(not\ A)$ is $\llcorner\tau^{cli}_{\mathbf{Z}} A$; for every literal of the form *not not* A its translations $\tau^{cli}_{\mathbf{Z}}(not\ not\ A)$ is $\neg\neg\tau^{cli}_{\mathbf{Z}} A$ and its translation $\tau^{dlv}_{\mathbf{Z}}(not\ not\ A)$ is $\llcorner\llcorner\tau^{dlv}_{\mathbf{Z}} A$.

We now define the translation τ^x as follows:

4. for every rule R of form (4), its translation $\tau^x R$ is the universal closure of
$$\tau^x_{\mathbf{Z}} B_1 \wedge \cdots \wedge \tau^x_{\mathbf{Z}} B_n \rightarrow \tau^x_{\mathbf{Z}} Head,$$
where $\mathbf{Z}$ is the list of the global variables of R.
5. for every program Π, its translation $\tau^x\Pi$ is the theory containing $\tau^x R$ for each rule R in Π.

Note that τ^{cli} and τ^{dlv} only differ in the translation of negation and in the assignment of different function constants to set symbols. For instance, rule (3) is translated into the universal closure of

$$\begin{aligned} &company(C_1) \wedge company(C_3) \\ &\quad \wedge sum(s^x_{ctr}(C_1, C_3)) > 50 \rightarrow controls(C_1, C_3) \end{aligned} \tag{7}$$

where variables C_1 and C_3 are of general sort, and x is either *cli* or *dlv* depending on the semantics considered.

Standard interpretations. A *standard interpretation* I is an interpretation of $\sigma(\mathcal{P}, \mathcal{S})$ that satisfies the following *conditions*:

1. universe $|I|^{s_{gen}}$ is the set containing all ground terms of the general sort;
2. universe $|I|^{s_{tuple}}$ is the set of all tuples of form $\langle d_1, \ldots, d_k\rangle$ with $d_i \in |I|^{s_{gen}}$ for each set symbol $E/\mathbf{X}$ in $\mathcal{S}$ with E of the form of (5);
3. the elements of $|I|^{s_{set}}$ are sets of elements from $|I|^{s_{tuple}}$;
4. I interprets each ground program term as itself;
5. I interprets predicate symbols $>, \geq, <, \leq$ according to the total order chosen earlier;
6. I interprets each tuple term of form $tuple(t_1, \ldots, t_k)$ as the tuple $\langle t_1^I, \ldots, t_k^I\rangle$;
7. $\in^I$ is the set of pairs (t, s) such that tuple t belongs to set s;
8. for term t_{set} of sort s_{set}, $count(t_{set})^I$ is $\widehat{\texttt{count}}(t^I_{set})$;
9. for term t_{set} of sort s_{set}, $sum(t_{set})^I$ is $\widehat{\texttt{sum}}(t^I_{set})$;

An *agg-interpretation* is a standard interpretation I satisfying, for every set symbol $E/\mathbf{X}$ in $\mathcal{S}$ with E of the form of (5) and for all $x \in \{cli, dlv\}$, that $s^x_{|E/\mathbf{X}|}(\mathbf{x})^I$ is the set of all tuples of the form $\langle (t_1)^{\mathbf{XY}}_{\mathbf{xy}}, \ldots, (t_k)^{\mathbf{XY}}_{\mathbf{xy}}\rangle$ such that I satisfies $\tau^x(l_1)^{\mathbf{XY}}_{\mathbf{xy}} \wedge \cdots \wedge \tau^x(l_m)^{\mathbf{XY}}_{\mathbf{xy}}$.

For instance, the program representing the Company Control problem has a unique set symbol that is associated with the function symbols $s^x_{ctr}/2$ $(x \in \{cli, dlv\})$. If I is a agg-interpretation such that $ctrStk^I$ is the set containing $(c_1, c_2, c_3, 10)$ and $(c_1, c_4, c_3, 20)$, it follows that $s^x_{ctr}(c_1, c_3)^I$ (with $x \in \{cli, dlv\}$) is the set containing tuples $\langle 10, c_2\rangle$ and $\langle 20, c_4\rangle$.

An ht-interpretation $\langle H, I\rangle$ is said to be *standard* if both H and I are standard. An *agg-ht-interpretation* is a standard ht-interpretation $\langle H, I\rangle$ satisfying that I is an agg-interpretation and the following conditions for every set symbol $E/\mathbf{X}$ in $\mathcal{S}$ with E of the form of (5) :

- $s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})^H$ is the set of all tuples of form $\langle (t_1)^{\mathbf{XY}}_{\mathbf{xy}}, \ldots, (t_k)^{\mathbf{XY}}_{\mathbf{xy}}\rangle$ such that $\langle H, I\rangle$ satisfies $\tau^{cli}(l_1)^{\mathbf{XY}}_{\mathbf{xy}} \wedge \cdots \wedge \tau^{cli}(l_m)^{\mathbf{XY}}_{\mathbf{xy}}$; and
- $s^{dlv}_{|E/\mathbf{X}|}(\mathbf{x})^H$ is the set of all tuples of form $\langle (t_1)^{\mathbf{XY}}_{\mathbf{xy}}, \ldots, (t_k)^{\mathbf{XY}}_{\mathbf{xy}}\rangle$ such that H satisfies $\tau^{dlv}(l_1)^{\mathbf{XY}}_{\mathbf{xy}} \wedge \cdots \wedge \tau^{dlv}(l_m)^{\mathbf{XY}}_{\mathbf{xy}}$.

Let us consider now an ht-interpretation $\langle H, I\rangle$ where I is as described above and $ctrStk^H$ is the set containing $(c_1, c_2, c_3, 10)$. Then, $s^x_{ctr}(c_1, c_3)^H$ is the set containing tuples $\langle 10, c_2\rangle$ (with $x \in \{cli, dlv\}$). In this example, there is no difference between the semantics of `clingo` and `dlv`. As an example of where these semantics differ, consider an ht-interpretation $\langle H, I\rangle$ with $p^H = r^H = \emptyset$ and $p^I = q^I = q^H = r^I = \{1\}$, and rule

```
p(1) :- #sum{X : q(X), not r(X)} < 1.        (8)
```

This rule is translated into the sentences

$$sum(s^{cli}_{qr}) < 1 \rightarrow p(1) \tag{9}$$
$$sum(s^{dlv}_{qr}) < 1 \rightarrow p(1) \tag{10}$$

for the `clingo` and `dlv` semantics, respectively. It is clear that I satisfies both rules because $1 \in p^I$. However, when considering the ht-interpretation $\langle H, I\rangle$, only the second rule is satisfied. On the one hand, $(s^x_{qr})^I$ (with $x \in \{cli, dlv\}$) is the empty set. Furthermore, $(s^{cli}_{qr})^H$ is also the empty set because $\langle H, I\rangle \not\models \neg r(1)$, and the antecedent of (9) is satisfied. Then, the rule is not satisfied because the consequent is not satisfied due to $1 \notin p^H$. On the other hand, $(s^{dlv}_{qr})^H$ is the set containing 1 because $H \models q(1) \wedge \llcorner r(1)$. Hence, $\langle H, I\rangle$ does not satisfy the antecedent of (10) and the rule is satisfied.

We now define stable models for programs with aggregates. A model that is also an agg-interpretation is called an *agg-model* and an agg-ht-interpretation that satisfies a formula or theory is called an *agg-ht-model*.

Definition 1. *An agg-model I is an* agg-stable model *if there is no agg-ht-model $\langle H, I\rangle$ with $H <^{\mathcal{PF}} I$.*

Correspondence with `clingo` and `dlv`

We establish now the correspondence between the semantics of programs with aggregates introduced in the previous section and the semantics of the solver `clingo`, named Abstract Gringo (Gebser et al. 2015), and the solver `dlv`, which is based on the FLP-reduct (Faber, Pfeifer, and Leone 2011). These semantics are stated in terms of infinitary formulas following the work by Harrison and Lifschitz (2019).

Infinitary Formulas. We extend the definitions of infinitary logic by Truszczyński (2012) to formulas with intensional functions and the $\llcorner$ connective. For every nonnegative integer r, *infinitary ground formulas of rank* r are defined recursively:

- every ground atom in σ is a formula of rank 0,
- if Γ is a set of formulas, and r is the smallest nonnegative integer that is greater than the ranks of all elements of Γ, then $\Gamma^\wedge$ and $\Gamma^\vee$ are formulas of rank r,
- if F and G are formulas, and r is the smallest nonnegative integer that is greater than the ranks of F and G, then $F \to G$ is a formula of rank r,
- if F is a formula, and r is the smallest nonnegative integer that is greater than the rank F, then $\llcorner F$ is a formula of rank r.

We write $\{F,G\}^\wedge$ as $F \wedge G$, $\{F,G\}^\vee$ as $F \vee G$, and $\emptyset^\vee$ as $\bot$.

We extend the satisfaction relation for ht-interpretations to infinitary formulas by adding the following two conditions to the definition for FO formulas:

- $\langle H, I\rangle \models_{ht} \Gamma^\wedge$ if for every formula F in Γ, $\langle H, I\rangle \models_{ht} F$,
- $\langle H, I\rangle \models_{ht} \Gamma^\vee$ if there is a formula F in Γ such that $\langle H, I\rangle \models_{ht} F$,

We write $I \models F$ if $\langle I, I\rangle \models_{ht} F$.

Truszczyński (2012) defines the satisfaction of infinitary formulas with respect to sets of ground atoms instead of FO interpretations. We can define such satisfaction relation for infinitary formulas when we have no intensional functions. We say that an infinitary ground formula is *propositional* if it does not contain intensional functions. For a signature σ, by σ^p we denote the set of all ground atoms over σ that do not contain intensional functions. Subsets of a propositional signature σ^p are called *propositional interpretations*. The satisfaction relation between a propositional interpretation $\mathcal{A}$ and an infinitary propositional formula is defined recursively:

- for every ground atom A from σ, $\mathcal{A} \models A$ if A belongs to $\mathcal{A}$,
- $\mathcal{A} \models \Gamma^\wedge$ if for every formula F in Γ, $\mathcal{A} \models F$,
- $\mathcal{A} \models \Gamma^\vee$ if there is a formula F in Γ such that $\mathcal{A} \models F$,
- $\mathcal{A} \models F \to G$ if $\mathcal{A} \not\models F$ or $\mathcal{A} \models G$,
- $\mathcal{A} \models \llcorner F$ if $\mathcal{A} \not\models F$.

In the following, if I is an interpretation, then $\mathcal{I}$ denotes the set of atomic formulas of σ^p satisfied by I. With this notation, the following result is easily proved by induction.

Proposition 1. *Let F be an infinitary propositional formula. Then, $I \models F$ iff $\mathcal{I} \models F$.*

Grounding. The *grounding* of a FO sentence allows us to replace quantifiers with infinitary conjunctions and disjunctions. Formally, the *grounding of a First-Order sentence F with respect to an interpretation I and sets $\mathcal{P}$ and $\mathcal{F}$ of intensional predicate and function symbols* is defined as follows:

- $gr_I^{\mathcal{PF}}(\bot)$ is $\bot$;
- $gr_I^{\mathcal{PF}}(p(\mathbf{t}))$ is $p(\mathbf{t})$ if $p(\mathbf{t})$ contains intensional symbols;
- $gr_I^{\mathcal{PF}}(p(\mathbf{t}))$ is $\top$ if $p(\mathbf{t})$ does not contain intensional symbols and $I \models p(\mathbf{t})$; and $gr_I^{\mathcal{PF}}(p(\mathbf{t}))$ is $\bot$ otherwise;
- $gr_I^{\mathcal{PF}}(t_1 = t_2)$ is $(t_1 = t_2)$ if t_1 or t_2 contain intensional symbols;
- $gr_I^{\mathcal{PF}}(t_1 = t_2)$ is $\top$ if t_1 and t_2 do not contain intensional symbols and $t_1^I = t_2^I$ and $\bot$ otherwise;
- $gr_I^{\mathcal{PF}}(\llcorner F)$ is $\llcorner gr_I^{\mathcal{PF}}(F)$;
- $gr_I^{\mathcal{PF}}(F \otimes G)$ is $gr_I^{\mathcal{PF}}(F) \otimes gr_I^{\mathcal{PF}}(G)$ if $\otimes$ is $\wedge$, $\vee$, or $\to$;
- $gr_I^{\mathcal{PF}}(\exists X\, F(X))$ is $\{gr_I^{\mathcal{PF}}(F(u)) \mid u \in |I|^s\}^\vee$ if X is a variable of sort s;
- $gr_I^{\mathcal{PF}}(\forall X\, F(X))$ is $\{gr_I^{\mathcal{PF}}(F(u)) \mid u \in |I|^s\}^\wedge$ if X is a variable of sort s;

For a first-order theory Γ, we define $gr_I^{\mathcal{PF}}(\Gamma) = \{gr_I^{\mathcal{PF}}(F) \mid F \in \Gamma\}^\wedge$. For any first-order theory Γ, $gr_I^{\mathcal{PF}}(\Gamma)$ is a infinitary formula, which may contain intensional functions or the $\llcorner$ connective. We write $gr_I(\cdot)$ instead of $gr_I^{\mathcal{PF}}(\cdot)$ when clear by the context.

Proposition 2. $\langle H, I\rangle \models_{ht} F$ *iff* $\langle H, I\rangle \models_{ht} gr_I(F)$.

Standard formulas and minimal models. We say that an infinitary propositional formula is *standard* if it does not contain the $\llcorner$ connective. The definitions of the semantics of `clingo` and `dlv` only use standard infinitary formulas and rely on the notion of minimal models. A propositional interpretation $\mathcal{A}$ satisfies a set Γ of formulas, in symbols $\mathcal{A} \models \Gamma$, if it satisfies every formula in Γ. We say that a set $\mathcal{A}$ of atoms is a $\subseteq$-minimal model of a set of infinitary formulas Γ, if $\mathcal{A} \models \Gamma$ and there is no $\mathcal{B}$ satisfying $\mathcal{B} \models \Gamma$ and $\mathcal{B} \subset \mathcal{A}$.

Clingo. The *FT-reduct* $F^{\mathcal{A}}$ of a standard infinitary formula F with respect to a propositional interpretation $\mathcal{A}$ is defined recursively. If $\mathcal{A} \not\models F$ then $F^{\mathcal{A}}$ is $\bot$; otherwise,

- for every ground atom A, $A^{\mathcal{A}}$ is A
- $(\Gamma^\wedge)^{\mathcal{A}} = \{G^{\mathcal{A}} \mid G \in \Gamma\}^\wedge$,
- $(\Gamma^\vee)^{\mathcal{A}} = \{G^{\mathcal{A}} \mid G \in \Gamma\}^\vee$,
- $(G \to H)^{\mathcal{A}}$ is $G^{\mathcal{A}} \to H^{\mathcal{A}}$,

We say that a propositional interpretation $\mathcal{A}$ is a *FT-stable model* of a formula F if it is a $\subseteq$-minimal model of $F^{\mathcal{A}}$. We say that a set $\mathcal{A}$ of ground atoms is a *clingo answer set* of a program Π if $\mathcal{A}$ a FT-stable model of $\tau\Pi$ where τ is the translation from logic programs to infinitary formulas defined by Gebser et al. (2015). The following result states that the usual relation between ht-interpretations and the FT-reduct is satisfied in our settings.

Proposition 3. *Let F be a standard infinitary formula of σ^p. Then, $\langle H, I\rangle \models_{ht} F$ iff $\mathcal{H} \models F^{\mathcal{I}}$.*

For agg-ht-interpretations we can state the relation $<^{\mathcal{PF}}$ in terms of the atomic formulas satisfied by it as follows:

Proposition 4. *Let $\langle H, I\rangle$ be an agg-interpretation. Then, $H <^{\mathcal{PF}} I$ iff $\mathcal{H} \subset \mathcal{I}$.*

Using Propositions 1-4, we can prove the relation between clingo answer sets and agg-stable models of the corresponding FO theory. Note that clingo answer sets are propositional interpretations while agg-stable models of FO theories are FO interpretations. To fill this gap, we introduce the following notation. If I is an agg-stable model of $\tau^{cli}\Pi$, we say that $\mathcal{I}$ is a *fo-clingo answer set* of Π.

Theorem 5. *The fo-clingo answer sets of any program coincide with its clingo answer sets.*

Proof sketch. The core of the proof consists of showing that $\langle H, I\rangle \models_{ht} \tau^{cli}\Pi$ iff $\mathcal{H} \models (\tau\Pi)^{\mathcal{I}}$ holds. By Proposition 2, we get $\langle H, I\rangle \models_{ht} \tau^{cli}\Pi$ iff $\langle H, I\rangle \models_{ht} gr_I(\tau^{cli}\Pi)$. Note that $gr_I(\tau^{cli}\Pi)$ is not an infinitary propositional formula, because it may contain intensional functions. Thus, we cannot apply Proposition 3 directly. However, we can prove that $\langle H, I\rangle \models_{ht} gr_I(\tau^{cli}\Pi)$ iff $\langle H, I\rangle \models_{ht} \tau\Pi$ holds and use Proposition 3 to prove the stated result. Finally, Proposition 4 is used to state the correspondence between stable models of $\tau\Pi$ and agg-stable models of $\tau^{cli}\Pi$. □

The `dlv` semantics. Similarly to the Abstract Gringo semantics, the `dlv` semantics can be stated in terms of the same translation τ to infinitary formulas, but using a different reduct (Harrison and Lifschitz 2019). Let F be implication $F_1 \to F_2$. Then, the *FLP-reduct* $FLP(F, \mathcal{A})$ of F w.r.t. a propositional interpretation $\mathcal{A}$ is F if $\mathcal{A} \models F_1$, and $\top$ otherwise. For a conjunction of implications $\mathcal{F}^\wedge$, we define

$$FLP(\mathcal{F}^\wedge, \mathcal{A}) \;=\; \{FLP(F, \mathcal{A}) \mid F \in \mathcal{F}\}^\wedge$$

A set $\mathcal{A}$ of ground atoms is an *FLP-stable model* of F if it is a $\subseteq$-minimal model of $FLP(F, \mathcal{A})$. We say that a set $\mathcal{A}$ of ground atoms is a *dlv answer set* of a program Π if $\mathcal{A}$ is an FLP-stable model of $\tau\Pi$. If I is an agg-stable model of $\tau^{dlv}\Pi$, we say that $\mathcal{I}$ is a *fo-dlv answer set* of Π.

Theorem 6. *The fo-dlv answer sets of any program coincide with its dlv answer sets.*

Proof sketch. The structure of the proof is analogous to the one of Theorem 5. Here, the key step of the proof consists of showing $\langle H, I\rangle \models_{ht} \tau^{dlv}\Pi$ iff both $\mathcal{I} \models \tau\Pi$ and $\mathcal{H} \models FLP(\tau\Pi, \mathcal{I})$. □

Strong Equivalence

We say that two programs Π_1 and Π_2 are *strongly equivalent for `clingo`* if program $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have the same `clingo` answer sets for any program Δ.

We assume a signature $\sigma(\mathcal{P}, \mathcal{S})$ where $\mathcal{P}$ and $\mathcal{S}$ are the sets that respectively contain all predicate and all set symbols occurring in $\Pi_1 \cup \Pi_2$.

Theorem 7. *The following conditions are equivalent:*

- Π_1 *and* Π_2 *are strongly equivalent for `clingo`;*
- $\tau^{cli}(\Pi_1)$ *and* $\tau^{cli}(\Pi_2)$ *have the same agg-ht-models.*

Let us consider the program formed by rule (8). This program has $\{p(1)\}$ as its unique clingo answer set. Similarly, the program formed by the rule

```
p(1) :- not #sum{X : q(X), not r(X)} >= 1.        (11)
```

also has $\{p(1)\}$ as its unique clingo answer set. However, these two programs are not strongly equivalent under the `clingo` semantics. To illustrate this claim, consider an agg-ht-interpretation $\langle H, I\rangle$ with $p^H = r^H = r^I = \emptyset$, and $p^I = q^H = \{1\}$, and $q^I = \{1, -1\}$. One the one hand, $\langle H, I\rangle$ satisfies (9) because its antecedent is not satisfied as we have $sum(s_{qr}^{cli})^H = 1$. One the other hand, $\langle H, I\rangle$ does satisfies the formula

$$\neg sum(s_{qr}^{cli}) \geq 1 \to p(1) \tag{12}$$

obtained by applying τ^{cli} to (11). Note that in the scope of negation, we only look at the value in I and we have $sum(s_{qr}^{cli})^I = 0$. By Theorem 7, this implies that the two programs are not strongly equivalent under the `clingo` semantics. This affirmation can confirmed by adding context

```
q(1).   q(-X) :- p(X).   :- not p(1).   (13)
```

When added to rule (8), the resulting program has no clingo answer sets, but when added to rule (11), the resulting program has $\{q(1),\, q(-1),\, p(1)\}$ as its unique answer set.

Similarly, we say that two programs Π_1 and Π_2 are *strongly equivalent for `dlv`* if programs $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have the same `dlv` answer sets for any program Δ.

Theorem 8. *The following conditions are equivalent.*

- Π_1 *and* Π_2 *are strongly equivalent for `dlv`;*
- $\tau^{dlv}(\Pi_1)$ *and* $\tau^{dlv}(\Pi_2)$ *have the same agg-ht-models.*

Though programs containing rules (8) and (11) are not strongly equivalent for `clingo`, they are strongly equivalent for `dlv`. Applying τ^{dlv} to rule (11) yields formula

$$\llcorner sum(s_{qr}^{dlv}) \geq 1 \to p(1) \tag{14}$$

and any ht-interpretation $\langle H, I\rangle$ satisfies the antecedent of (14) iff $H \not\models sum(s_{qr}^{dlv}) \geq 1$ and $I \not\models sum(s_{qr}^{dlv}) \geq 1$ iff $H \models sum(s_{qr}^{dlv}) < 1$ and $I \models sum(s_{qr}^{dlv}) < 1$ iff $\langle H, I\rangle$ satisfies $sum(s_{qr}^{dlv}) < 1$, which is the antecedent of (10). By Theorem 8, this implies that the two programs are strongly equivalent for `dlv`.

We can also use these translations to establish strong equivalence results across the two semantics.

Theorem 9. *The following conditions are equivalent.*

- $\tau^{cli}(\Pi_1)$ *and* $\tau^{dlv}(\Pi_2)$ *have the same agg-ht-models.*
- *the clingo answer sets of* $\Pi_1 \cup \Delta$ *coincide with the dlv answer sets of* $\Pi_2 \cup \Delta$ *for every set* Δ *of rules such that* $\tau^{cli}(\Delta)$ *and* $\tau^{dlv}(\Delta)$ *have the same agg-ht-models.*

In particular, note that $\tau^{cli}(\Delta)$ and $\tau^{dlv}(\Delta)$ are equivalent for any set of rules without aggregates nor double negation. The restriction on the agg-models of $\tau^{cli}(\Delta)$ and $\tau^{dlv}(\Delta)$ is necessary because the same added rules may have different behavior under the two semantics. As an example, consider program Π_1 formed by rule (8) plus constraint

```
:- q(X), r(X).        (15)
```

and program Π_2 formed by rule (11) plus constraint (15). Recall that $\tau^{cli}(8)$ is sentence (9) and $\tau^{dlv}(11)$ is sentence (14). As we discussed above, any ht-interpretation satisfies the antecedent of (14) iff it satisfies $sum(s^{dlv}_{qr}) < 1$. Hence, it is enough to show that $(s^{cli}_{qr})^H = (s^{dlv}_{qr})^H$ for every agg-interpretation $\langle H, I\rangle$ that satisfies $\forall X \neg(q(X) \wedge r(X))$. Every $\langle H, I\rangle$ that satisfies this sentence, satisfies that $H \models q(c)$ implies $H \not\models r(c)$ for every object constant c of general sort. Hence $\langle H, I\rangle$ satisfies $q(c) \wedge \neg r(c)$ iff H satisfies $q(c) \wedge \llcorner r(c)$ for every object constant c of general sort. This implies $(s^{cli}_{qr})^H = (s^{dlv}_{qr})^H$. It is well-known that for programs where all aggregate literals are positive, the `clingo` and `dlv` semantics coincide (Ferraris 2011; Harrison and Lifschitz 2018). As illustrated by this example, Theorem 9 give us tools to prove this correspond for some programs where some aggregates are not positive.[3]

Strong Equivalence using Classical Logic

In this section, we show how an additional syntactic transformation γ allows us to replace the logic of Here-and-There by classical FO theory. This also allows us to remove the non-standard negation $\llcorner$ and replace the semantic condition that characterizes agg-interpretation in favor of axiom schemata. This is a generalization of the transformation by Fandinno and Lifschitz (2023) and it is similar to the one used by Bartholomew and Lee (2019) to define the SM operator for FO formulas with intensional functions.

We define a signature $\hat{\sigma}$ that is obtained from the signature σ by adding, for every predicate symbol p other than comparison symbols (4), a new predicate symbol $\hat{p}$ of the same arity; and for every function symbol $s^x_{|E/\mathbf{X}|}$ with $x \in \{cli, dlv\}$, a new function symbol $\hat{s}^x_{|E/\mathbf{X}|}$ of the same arity.

For any expression E of signature σ, by $\hat{E}$ we denote the expression of $\hat{\sigma}$ obtained from E by replacing every occurrence of every predicate symbol p by $\hat{p}$ and every occurrence of function symbol $s^x_{|E/\mathbf{X}|}$ by $\hat{s}^x_{|E/\mathbf{X}|}$. The translation γ, which relates the logic of here-and-there to classical logic, maps formulas over σ to formulas over $\hat{\sigma}$. It is defined recursively:

- $\gamma F = F \wedge \hat{F}$ if F is atomic,
- $\gamma(\neg F) = \neg \hat{F}$,
- $\gamma(\llcorner F) = \neg \hat{F} \wedge \neg n(F)$,
- $\gamma(F \otimes G) = \gamma F \otimes \gamma G$ with $\otimes \in \{\wedge, \vee\}$.
- $\gamma(F \rightarrow G) = (\gamma F \rightarrow \gamma G) \wedge (\hat{F} \rightarrow \hat{G})$,
- $\gamma(\forall X\, F) = \forall X\, \gamma F$,
- $\gamma(\exists X\, F) = \exists X\, \gamma F$.

where $n(F)$ is the result of replacing all occurrences of $\llcorner$ by $\neg$ in F. To apply γ to a set of formulas means to apply γ to each of its members. Note that $\gamma\Gamma$ is always a standard FO theory (without the $\llcorner$ operator) over the signature $\hat{\sigma}$.

[3]Recall that an aggregate literal is called *positive* if it is not in the scope of negation and negation does not occur within its scope (Harrison and Lifschitz 2018).

For any ht-interpretation $\langle H, I\rangle$ of σ, I^H stands for the interpretation of $\hat{\sigma}$ that has the same domain as I, interprets extensional symbols in the same way as I, interprets the other function symbols f, $\hat{f}$ as $f^{I^H} = f^H$ and $\hat{f}^{I^H} = f^I$, and other predicate constants p, $\hat{p}$ as follows:

$$I^H \models p(\mathbf{d}^*) \text{ iff } H \models p(\mathbf{d}^*); \qquad I^H \models \hat{p}(\mathbf{d}^*) \text{ iff } I \models p(\mathbf{d}^*).$$

Proposition 10. $\langle H, I\rangle \models_{ht} \Gamma$ *iff* $I^H \models \gamma\Gamma$.

The following set of formulas characterizes which interpretations of the signature $\hat{\sigma}$ correspond to ht-interpretations. By HT we denote the set of all formulas of the form $\forall \mathbf{X}(p(\mathbf{X}) \rightarrow \hat{p}(\mathbf{X}))$ for every intensional predicate p. By AGG we denote the set of all formulas of the form

$$\forall \mathbf{X} T\big(T \in \hat{s}^x_{|E/\mathbf{X}|}(\mathbf{X}) \leftrightarrow \exists \mathbf{Y} \hat{F}^x\big) \tag{16}$$

$$\forall \mathbf{X} T\big(T \in s^{cli}_{|E/\mathbf{X}|}(\mathbf{X}) \leftrightarrow \exists \mathbf{Y} \gamma(F^{cli})\big) \tag{17}$$

$$\forall \mathbf{X} T\big(T \in s^{dlv}_{|E/\mathbf{X}|}(\mathbf{X}) \leftrightarrow \exists \mathbf{Y} n(F^{dlv})\big) \tag{18}$$

for every $E/\mathbf{X}$ in $\mathcal{S}$ with E of the form of (5) and where

$$F^{cli} \text{ is } T = tuple(t_1, \dots, t_k) \wedge \tau^{cli}(l_1) \wedge \cdots \wedge \tau^{cli}(l_m)$$
$$F^{dlv} \text{ is } T = tuple(t_1, \dots, t_k) \wedge \tau^{dlv}(l_1) \wedge \cdots \wedge \tau^{dlv}(l_m)$$

Proposition 11. *An interpretation of the signature $\hat{\sigma}$ satisfies HT and AGG iff it can be represented in the form I^H for some agg-ht-interpretation $\langle H, I\rangle$.*

We are ready to state the main result of this section showing that we can use classical FO logic to reason about strong equivalence under the `clingo` and `dlv` semantics.

Theorem 12. *Finite programs Π_1 and Π_2 are strongly equivalent under the `clingo` semantics iff all standard interpretations of $\hat{\sigma}$ satisfy the sentence*

$$\bigwedge HT \wedge \bigwedge AGG \rightarrow (F_1 \leftrightarrow F_2)$$

where F_i is the conjunction of all sentences in $\gamma\tau^{cli}\Pi_i$. The same holds if we replace `clingo` and τ^{cli} by `dlv` and τ^{dlv}.

Discussion and Conclusions

In this paper, we provided a characterization of the semantics of logic programs with aggregates which bypasses grounding. We focus on the semantics for recursive aggregates of used by ASP solvers `clingo` and `dlv`. Our characterization reflects the intuition that aggregates are functions that apply to sets, usually missing in most formal characterizations of aggregates, which treat them as monolithic constructs. To achieve that, we translate logic programs with aggregates into First-Order formulas with intensional functions, establishing a connection between these two extensions of logic programs. We also show that this characterization can be used to study the strong equivalence of programs with aggregates under either semantics. Finally, we show how to reduce the task of checking strong equivalence to reasoning in classical First-Order logic, which serves as a foundation for automating this procedure. We also axiomatize the meaning of the symbols used to represent sets. The axiomatization of the symbols representing aggregate operations (`sum` and `count`) developed by Fandinno, Hansen, and Lierler (2022) for non-recursive aggregates also applies to recursive aggregates because these function symbols stay non-intensional. Immediate future work includes the integration of this characterization of aggregates with the formalization of arithmetics used by the verification tool ANTHEM (Fandinno et al. 2020, 2023) and the implementation of a new verification tool that can handle programs with aggregates.

References

Asuncion, V.; Chen, Y.; Zhang, Y.; and Zhou, Y. 2015. Ordered completion for logic programs with aggregates. *Artificial Intelligence*, 224: 72–102.

Balduccini, M. 2013. ASP with non-herbrand partial functions: A language and system for practical use. *Theory and Practice of Logic Programming*, 13(4-5): 547–561.

Bartholomew, M.; and Lee, J. 2019. First-order stable model semantics with intensional functions. *Artificial Intelligence*, 273: 56–93.

Bartholomew, M.; Lee, J.; and Meng, Y. 2011. First-Order Extension of the FLP Stable Model Semantics via Modified Circumscription. In Walsh, T., ed., *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI'11)*, 724–730. IJCAI/AAAI Press.

Cabalar, P. 2011. Functional answer set programming. *Theory and Practice of Logic Programming*, 11(2-3): 203–233.

Cabalar, P.; Fandinno, J.; Fariñas del Cerro, L.; and Pearce, D. 2018. Functional ASP with Intensional Sets: Application to Gelfond-Zhang Aggregates. *Theory and Practice of Logic Programming*, 18(3-4): 390–405.

Cabalar, P.; Fandinno, J.; Schaub, T.; and Schellhorn, S. 2019. Gelfond-Zhang aggregates as propositional formulas. *Artificial Intelligence*, 274: 26–43.

Dovier, A.; Pontelli, E.; and Rossi, G. 2003. Intensional Sets in CLP. In Palamidessi, C., ed., *Proceedings of the Nineteenth International Conference on Logic Programming (ICLP'03)*, volume 2916 of *Lecture Notes in Computer Science*, 284–299. Springer-Verlag.

Faber, W.; Pfeifer, G.; and Leone, N. 2011. Semantics and Complexity of Recursive Aggregates in Answer Set Programming. *Artificial Intelligence*, 175(1): 278–298.

Fandinno, J.; Hansen, Z.; and Lierler, Y. 2022. Axiomatization of Aggregates in Answer Set Programming. In *Proceedings of the Thirty-sixth National Conference on Artificial Intelligence (AAAI'22)*, 5634–5641. AAAI Press.

Fandinno, J.; Hansen, Z.; Lierler, Y.; Lifschitz, V.; and Temple, N. 2023. External Behavior of a Logic Program and Verification of Refactoring. *Theory and Practice of Logic Programming*, 23(4): 933–947.

Fandinno, J.; and Lifschitz, V. 2023. On Heuer's Procedure for Verifying Strong Equivalence. In Gaggl, S.; Martinez, M.; and Ortiz, M., eds., *Proceedings of the Eighteenth European Conference on Logics in Artificial Intelligence (JELIA'23)*, volume 14281 of *Lecture Notes in Computer Science*. Springer-Verlag.

Fandinno, J.; Lifschitz, V.; Lühne, P.; and Schaub, T. 2020. Verifying Tight Logic Programs with anthem and vampire. *Theory and Practice of Logic Programming*, 20(5): 735–750.

Ferraris, P. 2011. Logic programs with propositional connectives and aggregates. *ACM Transactions on Computational Logic*, 12(4): 25:1–25:40.

Ferraris, P.; Lee, J.; and Lifschitz, V. 2011. Stable models and circumscription. *Artificial Intelligence*, 175(1): 236–263.

Gebser, M.; Harrison, A.; Kaminski, R.; Lifschitz, V.; and Schaub, T. 2015. Abstract Gringo. *Theory and Practice of Logic Programming*, 15(4-5): 449–463.

Gelfond, M.; and Zhang, Y. 2014. Vicious Circle Principle and Logic Programs with Aggregates. *Theory and Practice of Logic Programming*, 14(4-5): 587–601.

Gelfond, M.; and Zhang, Y. 2019. Vicious Circle Principle, Aggregates, and Formation of Sets in ASP Based Languages. *Artificial Intelligence*, 275: 28–77.

Harrison, A.; and Lifschitz, V. 2018. Relating Two Dialects of Answer Set Programming. In Fermé, E.; and Villata, S., eds., *Proceedings of the Seventeenth International Workshop on Nonmonotonic Reasoning (NMR'18)*, 99–108.

Harrison, A.; and Lifschitz, V. 2019. Relating Two Dialects of Answer Set Programming. *Theory and Practice of Logic Programming*, 19(5-6): 1006–1020.

Kemp, D.; and Stuckey, P. 1991. Semantics of Logic Programs with Aggregates. In Saraswat, V.; and Ueda, K., eds., *Proceedings of the 1991 International Symposium on Logic Programming (ISLP'91)*, 387–401. MIT Press.

Lee, J.; Lifschitz, V.; and Palla, R. 2008. A Reductive Semantics for Counting and Choice in Answer Set Programming. In Fox, D.; and Gomes, C., eds., *Proceedings of the Twenty-third National Conference on Artificial Intelligence (AAAI'08)*, 472–479. AAAI Press.

Lee, J.; and Meng, Y. 2012. Stable Models of Formulas with Generalized Quantifiers (Preliminary Report). In Dovier, A.; and Santos Costa, V., eds., *Technical Communications of the Twenty-eighth International Conference on Logic Programming (ICLP'12)*, volume 17, 61–71. Leibniz International Proceedings in Informatics (LIPIcs).

Lifschitz, V. 2012. Logic Programs with Intensional Functions. In Brewka, G.; Eiter, T.; and McIlraith, S., eds., *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning (KR'12)*. AAAI Press.

Lifschitz, V. 2019. *Answer Set Programming*. Springer-Verlag.

Lifschitz, V. 2022. Strong Equivalence of Logic Programs with Counting. *Theory and Practice of Logic Programming*, 22(4): 573–588.

Lifschitz, V.; Pearce, D.; and Valverde, A. 2001. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2(4): 526–541.

Lifschitz, V.; Pearce, D.; and Valverde, A. 2007. A Characterization of Strong Equivalence for Logic Programs with

Variables. In Baral, C.; Brewka, G.; and Schlipf, J., eds., *Proceedings of the Ninth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'07)*, volume 4483 of *Lecture Notes in Artificial Intelligence*, 188–200. Springer-Verlag.

Lin, F.; and Wang, Y. 2008. Answer Set Programming with Functions. In Brewka, G.; and Lang, J., eds., *Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, 454–465. AAAI Press.

Mumick, I.; Pirahesh, H.; and Ramakrishnan, R. 1990. The Magic of Duplicates and Aggregates. In McLeod, D.; Sacks-Davis, R.; and Schek, H., eds., *Proceedings of the Sixteenth International Conference on Very Large Data Bases (VLDB'90)*, 264–277. Morgan Kaufmann Publishers.

Pearce, D.; and Valverde, A. 2008. Quantified Equilibrium Logic and Foundations for Answer Set Programs. In Garcia de la Banda, M.; and Pontelli, E., eds., *Proceedings of the Twenty-fourth International Conference on Logic Programming (ICLP'08)*, volume 5366 of *Lecture Notes in Computer Science*, 546–560. Springer-Verlag.

Pelov, N.; Denecker, M.; and Bruynooghe, M. 2007. Well-founded and stable semantics of logic programs with aggregates. *Theory and Practice of Logic Programming*, 7(3): 301–353.

Ross, K.; and Sagiv, Y. 1997. Monotonic Aggregation in Deductive Databases. *Journal of Computer and System Sciences*, 54(1): 79–97.

Simons, P.; Niemelä, I.; and Soininen, T. 2002. Extending and implementing the stable model semantics. *Artificial Intelligence*, 138(1-2): 181–234.

Son, T.; and Pontelli, E. 2007. A Constructive Semantic Characterization of Aggregates in Answer Set Programming. *Theory and Practice of Logic Programming*, 7(3): 355–375.

Truszczyński, M. 2012. Connecting First-Order ASP and the Logic FO(ID) through Reducts. In Erdem, E.; Lee, J.; Lierler, Y.; and Pearce, D., eds., *Correct Reasoning: Essays on Logic-Based AI in Honour of Vladimir Lifschitz*, volume 7265 of *Lecture Notes in Computer Science*, 543–559. Springer-Verlag.

Proof of Results

Some Results on Here-and-There Logic

The following results show that some of the usual properties of the logic of Here-and-There are preserved in the extension introduced here.

Proposition 13. *The following properties hold:*

- $\langle I, I\rangle \models_{ht} F$ *iff* $I \models F$.
- *If* $\langle H, I\rangle \models_{ht} F$*, then* $I \models F$.
- $\langle H, I\rangle \models_{ht} \neg F$ *iff* $I \not\models F$.
- $\langle H, I\rangle \models_{ht} \neg\neg F$ *iff* $I \models F$.

Proof. Item 1 is immmediate when $H = I$. *Item 2.* If F is an atomic sentence or a sentence of the forms $\llcorner F_1$ or $F_1 \to F_2$, the result follows from the definition of $\models_{ht}$. The remaining cases are proved by induction on the size of F. *Item 3.* $\langle H, I\rangle \models_{ht} \neg F$ iff $\langle H, I\rangle \models_{ht} F \to \bot$ iff $I \models F \to \bot$ and $\langle H, I\rangle \not\models_{ht} F$ iff $I \not\models F$ (the last equivalece is a consequence of Item 2). *Item 4* is an immmediate consequence of Item 3. □

The second item of Proposition 13 shows that the persistence property of the logic of Here-and-There is preserved in the extension introduced here.

The following result sheds some light on the behavior of the new negation connective.

Proposition 14. *The following properties hold:*

- $\langle H, I\rangle \models_{ht} \llcorner F$ *iff* $I \models \llcorner F$ *and* $H \models \llcorner F$.
- $\langle H, I\rangle \models_{ht} \llcorner\llcorner F$ *iff* $I \models F$ *and* $H \models F$.
- $\langle H, I\rangle \models_{ht} \llcorner\llcorner F$ *implies* $\langle H, I\rangle \not\models_{ht} \llcorner F$.
- $\langle H, I\rangle \models_{ht} \llcorner\llcorner p(\mathbf{t})$ *iff* $\langle H, I\rangle \models p(\mathbf{t})$.

Proof. Item 1: $\langle H, I\rangle \models_{ht} \llcorner F$ iff $I \not\models F$ and $H \not\models F$ (by definition) iff $I \models \llcorner F$ and $H \models \llcorner F$ (by definition).

Item 2: $\langle H, I\rangle \models_{ht} \llcorner\llcorner F$ iff $I \not\models \llcorner F$ and $H \not\models \llcorner F$ (by definition) iff $I \models F$ and $H \models F$.

Item 3: $\langle H, I\rangle \models_{ht} \llcorner\llcorner F$ implies $I \models F$ and $H \models F$ (Item 2) and, thus, $\langle H, I\rangle \not\models_{ht} \llcorner F$.

Item 4: $\langle H, I\rangle \models_{ht} \llcorner\llcorner p(\mathbf{t})$ iff $I \models p(\mathbf{t})$ and $H \models p(\mathbf{t})$ (by Item 2) iff $\langle H, I\rangle \models p(\mathbf{t})$ (by definition). □

Proposition 15. *The following properties hold if* $\mathbf{t}$ *does not contain intensional symbols.*

- $\langle H, I\rangle \models_{ht} p(\mathbf{t})$ *iff* $H \models p(\mathbf{t})$,
- $\langle H, I\rangle \models_{ht} \llcorner p(\mathbf{t})$ *iff* $\langle H, I\rangle \models_{ht} \neg p(\mathbf{t})$ *iff* $I \not\models p(\mathbf{t})$.

Proof. Item 1: $\langle H, I\rangle \models_{ht} p(\mathbf{t})$ iff $H \models p(\mathbf{t})$ and $I \models p(\mathbf{t})$ iff $\mathbf{t}^H \in p^H$ and $\mathbf{t}^I \in p^I$ iff $\mathbf{t}^H \in p^H$ (because $p^H \subseteq p^I$ and $\mathbf{t}^H = \mathbf{t}^I$) iff $H \models p(\mathbf{t})$.

Item 2: $\langle H, I\rangle \models_{ht} \llcorner p(\mathbf{t})$ iff $H \not\models p(\mathbf{t})$ and $I \not\models p(\mathbf{t})$ iff $\mathbf{t}^H \notin p^H$ and $\mathbf{t}^I \notin p^I$ iff $\mathbf{t}^I \notin p^I$ ($p^H \subseteq p^I$ and $\mathbf{t}^H = \mathbf{t}^I$) iff $I \not\models p(\mathbf{t})$. □

The first item of the Proposition 15 means that, when a theory is standard and has no intensional functions, our satisfaction relation is equivalent to the standard satisfaction relation in QEL (Pearce and Valverde 2008). The third item of the Proposition 15 implies that $\langle H, I\rangle \models_{ht} \llcorner\llcorner p(\mathbf{t})$ iff $\langle H, I\rangle \models_{ht} \neg\neg p(\mathbf{t})$ does not hold even when $\mathbf{t}$ does not contain intensional symbols. This behavior is consistent with the way a straightforward generalization of the FLP-reduct (Faber, Pfeifer, and Leone 2011) treats double negation (Harrison and Lifschitz 2018).

Proof of Section Correspondence with Clingo and DLV

Grounding

Lemma 16. *An interpretation* I *satisfies a sentence* F *over* σ^I *iff* I *satisfies* $gr_I(F)$.

Proof. By induction on the size of F.

Case 1: F is an atomic sentence that contains intensional symbols. Then, $gr_I(F) = F$ and the result is trivial.

Case 2: F is an atomic sentence that does not contain intensional symbols. Then, $gr_I(F) = \top$ if $I \models F$ and $gr_I(F) = \bot$ otherwise. The result follows immediately.

Case 3: F is of the form $\llcorner G$. Then, $gr_I(F) = \llcorner gr_I(G)$ and the result follows by induction hypothesis.

Case 4: F is $\forall X G(X)$ with X a variable of sort s. Then, $gr_I(F) = \{gr_I(G(d^*)) \mid d^* \in |I|^s\}^\wedge$ and
$\langle H, I\rangle \models_{ht} F$
iff $\langle H, I\rangle \models_{ht} G(d^*)$ for each $d \in |I|^s$
iff $\langle H, I\rangle \models_{ht} gr_I(G(d^*))$ for each $d \in |I|^s$ (induction)
iff $\langle H, I\rangle \models_{ht} gr_I(F)$.

The case where F is $\exists X G(X)$ is analogous to Case 2. The remaining cases where F is $G_1 \wedge G_2$, $G_1 \vee G_2$ or $F_1 \to F_2$ follow immediately by induction. □

Proof of Proposition 2. By induction on F similar to Lemma 16.

Case 1: F is an implication of the form $G_1 \to G_2$. Then, $gr_I(F)$ is the implication $gr_I^{\mathcal{PF}}(G_1) \to gr_I(G_2)$ and
$\langle H, I\rangle \models_{ht} F$
iff $I \models F$ and either $\langle H, I\rangle \not\models_{ht} G_1$ or $\langle H, I\rangle \models_{ht} G_2$
iff $I \models gr_I(F)$ (Lemma 16) and
either $\langle H, I\rangle \not\models_{ht} gr_I^{\mathcal{PF}}(G_1)$ or $\langle H, I\rangle \models_{ht} gr_I(G_2)$
iff $\langle H, I\rangle \models_{ht} gr_I(F)$.

Case 2: F is of the form $\llcorner G$. Then, $gr_I(F) = \llcorner gr_I(G)$ and
$\langle H, I\rangle \models \llcorner G$
iff $I \not\models G$ and $H \not\models G$
iff $I \not\models gr_I(G)$ and $H \not\models gr_I(G)$ (Lemma 16)
iff $\langle H, I\rangle \models \llcorner gr_I(G)$

The other cases follow by induction as in Lemma 16. □

FT-reduct

Proof of Proposition 3. We proceed by induction on the rank r of F. For a formula F of rank $r + 1$, assume that, for all formulas G of lesser rank than F occurring in F, $\langle H, I\rangle \models_{ht} G$ iff $\mathcal{H} \models G^{\mathcal{I}}$.

Base Case: $r = 0$, F is a ground atomic formula. Then,
$\langle H, I\rangle \models_{ht} F$
iff $H \models F$ and $I \models F$
iff $H \models F$ and $F^{\mathcal{I}} = F$

iff $\mathcal{H} \models F^{\mathcal{I}}$

Induction Step:
Case 1: Formula F of rank $r+1$ has form $\Gamma^{\wedge}$. Then, $\langle H, I\rangle \models_{ht} F$
iff $\langle H, I\rangle \models_{ht} G$ for every formula G in Γ (by definition)
iff $\mathcal{H} \models G^{\mathcal{I}}$ for every formula G in Γ (by induction)
iff $\mathcal{H} \models \{G^{\mathcal{I}} \mid G \in \Gamma\}^{\wedge}$
iff $\mathcal{H} \models F^{\mathcal{I}}$

Case 2: Formula F of rank $r+1$ has form $\Gamma^{\vee}$. Then, $\langle H, I\rangle \models_{ht} F$
iff $\langle H, I\rangle \models_{ht} G$ for some formula G in Γ (by definition)
iff $\mathcal{H} \models G^{\mathcal{I}}$ for this certain formula G in Γ (by induction)
iff $\mathcal{H} \models \{G^{\mathcal{I}} \mid G \in \Gamma\}^{\vee}$
iff $\mathcal{H} \models F^{\mathcal{I}}$

Case 3: Formula F of rank $r+1$ has form $G_1 \to G_2$. Then, $\langle H, I\rangle \models_{ht} F$
iff $I \models G_1 \to G_2$ and $\langle H, I\rangle \not\models_{ht} G_1$ or $\langle H, I\rangle \models_{ht} G_2$ (by definition)
iff $\langle I, I\rangle \models_{ht} G_1 \to G_2$ and
$\langle H, I\rangle \not\models_{ht} G_1$ or $\langle H, I\rangle \models_{ht} G_2$ (by definition)
iff $\mathcal{I} \models G_1^{\mathcal{I}} \to G_2^{\mathcal{I}}$ and $\mathcal{H} \not\models G_1^{\mathcal{I}}$ or $\mathcal{H} \models G_2^{\mathcal{I}}$ (by induction)
iff $\mathcal{I} \models G_1 \to G_2$ and $\mathcal{H} \not\models G_1^{\mathcal{I}}$ or $\mathcal{H} \models G_2^{\mathcal{I}}$
iff $\mathcal{I} \models G_1 \to G_2$ and $\mathcal{H} \models G_1^{\mathcal{I}} \to G_2^{\mathcal{I}}$
iff $F^{\mathcal{I}} = G_1^{\mathcal{I}} \to G_2^{\mathcal{I}}$ and $\mathcal{H} \models G_1^{\mathcal{I}} \to G_2^{\mathcal{I}}$
iff $\mathcal{H} \models F^{\mathcal{I}}$. □

The τ translation. The τ translation transforms a logic program into an infinitary propositional formula (Gebser et al. 2015). For any ground atom A, it is defined as follows:

- $\tau(A)$ is A,
- $\tau(\text{not } A)$ is $\neg A$, and
- $\tau(\text{not not } A)$ is $\neg\neg A$.

For a comparison symbol $\prec$ and ground terms t_1 and t_2, it is defined as follows:

- $\tau(t_1 \prec t_2)$ is $\top$ if the relation $\prec$ holds between t_1 and t_2 and $\bot$ otherwise.

For an aggregate element E of the form of (5) with $\mathbf{Y}$ the list of local variables occurring on it, Ψ_E denotes the set of tuples $\mathbf{y}$ of ground program terms of the same length as $\mathbf{Y}$. We say that a subset Δ of Ψ_E *justifies* aggregate atom $\mathtt{op}\{E\} \prec u$ if the relation $\prec$ holds between $\hat{\mathtt{op}}[\Delta]$ and u where $[\Delta] = \{\mathbf{t}^{\mathbf{Y}}_{\mathbf{y}} \mid \mathbf{y} \in \Delta\}$ and $\mathbf{t}$ is the list of terms $t_1, \ldots, t_k$ in E. For an aggregate atom A of the form of $\mathtt{op}\{E\} \prec u$ with global vriables $\mathbf{X}$, $\tau(A)$ it is defined as the infinitary formula

$$\bigwedge_{\Delta \in \chi} \left(\bigwedge_{\mathbf{y} \in \Delta} \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}} \to \bigvee_{\mathbf{y} \in \Psi_{E^{\mathbf{X}}_{\mathbf{x}}} \setminus \Delta} \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}} \right) \tag{19}$$

where χ is the set of subsets Δ of Ψ_E that do not justify aggregate atom $\mathtt{op}\{E^{\mathbf{X}}_{\mathbf{x}}\} \prec u$, and $\mathbf{l}$ is the list $l_1, \ldots, l_m$ of literals in E. We omit the parentheses and write τF instead of $\tau(F)$ when clear. For a rule R of the form of (6) with global variables $\mathbf{Z}$, τR is the infinitary conjunction of all formulas of the form

$$\tau(B_1)^{\mathbf{Z}}_{\mathbf{z}} \wedge \ldots \wedge \tau(B_n)^{\mathbf{Z}}_{\mathbf{z}} \to \tau Head^{\mathbf{Z}}_{\mathbf{z}} \tag{20}$$

with $\mathbf{z}$ being a list of ground program terms of the same length as $\mathbf{Z}$. For a program Π, $\tau\Pi = \{\tau R \mid R \in \Pi\}^{\wedge}$.

Correspondence with `clingo`. For any rule R without aggregates, it is not difficult to see that $\tau R = gr_I(\tau^{cli} R)$ for any standard interpretation I. For rules with aggregates τR and $gr_I(\tau^{cli} R)$ only differ in the translation of aggregates. The following two results show the relation between τA and $gr_I(\tau^{cli} A)$ for any aggregate atom A.

Lemma 17. *Let I be an agg-interpretation,* $\mathtt{op}$ *be an operation name and $x \in \{cli, dlv\}$. Then, I satisfies* $\mathtt{op}(s^{x}_{|E/\mathbf{X}|}(\mathbf{x})) \prec u$ *iff I satisfies* (19).

Proof. Let $\mathbf{Y}$ be the list of variables occurring in E that do not occur in $\mathbf{X}$. Let $\Delta_I = \{\, \mathbf{y} \in \Psi_E \mid I \models \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}} \,\}$ and F_I be the formula

$$\bigwedge_{\mathbf{y} \in \Delta_I} \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}} \to \bigvee_{\mathbf{y} \in \Psi_E \setminus \Delta_I} \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}}$$

Then, $I \not\models F_I$ and

$$s_{|E/\mathbf{X}|}(\mathbf{x})^I = \{\mathbf{t}^{\mathbf{XY}}_{\mathbf{xy}} \mid \mathbf{y} \in \Delta_I\} = [\Delta_I].$$

Consequently, we have

$I \models (19)$ iff F_I is not a conjunctive term of (19)
iff Δ_I justifies $\mathtt{op}(E^{\mathbf{X}}_{\mathbf{x}}) \prec u$
iff $\hat{\mathtt{op}}[\Delta_I] \prec u$
iff $\hat{\mathtt{op}}(s_{|E/\mathbf{X}|}(\mathbf{x})^I) \prec u$
iff $I \models \mathtt{op}(s_{|E/\mathbf{X}|}(\mathbf{x})) \prec u$ □

Lemma 18. *Let* $\mathtt{op}$ *be an operation name. Then, an agg-ht-interpretation $\langle H, I\rangle$ satisfies* $\mathtt{op}(s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})) \prec u$ *iff $\langle H, I\rangle$ satisfies* (19).

Proof. Let us denote $\mathtt{op}(s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})) \prec u$ as A in the following.

Case 1: $I \not\models A$. By Lemma 17, it follows that $I \not\models (19)$. Therefore, $\langle H, I\rangle \not\models_{ht} A$ and $\langle H, I\rangle \not\models_{ht} (19)$.

Case 2: $I \models A$. By Lemma 17, it follows that $I \models (19)$. Let

$$\Delta_{\langle H,I\rangle} = \{\, \mathbf{y} \in \Psi_E \mid \langle H, I\rangle \models_{ht} \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}} \,\}$$

and $F_{\langle H,I\rangle}$ be the formula

$$\bigwedge_{\mathbf{y} \in \Delta_{\langle H,I\rangle}} \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}} \to \bigvee_{\mathbf{y} \in \Psi_E \setminus \Delta_{\langle H,I\rangle}} \mathbf{l}^{\mathbf{XY}}_{\mathbf{xy}}$$

Then, $\langle H, I\rangle \not\models F_{\langle H,I\rangle}$ and

$$s_{|E/\mathbf{X}|}(\mathbf{x})^H = \{\mathbf{t}^{\mathbf{XY}}_{\mathbf{xy}} \mid \mathbf{y} \in \Delta_{\langle H,I\rangle}\} = [\Delta_{\langle H,I\rangle}].$$

Consequently, we have

$$\begin{aligned}
\langle H, I\rangle \models (19) &\text{ iff } I \models (19) \text{ and}\\
&\quad F_{\langle H,I\rangle} \text{ is not a conjunctive term of (19)}\\
&\text{iff } \Delta_{\langle H,I\rangle} \text{ justifies } \mathtt{op}(s_{|E/\mathbf{X}|}(\mathbf{x})) \prec u\\
&\text{iff } \hat{\mathtt{op}}([\Delta_{\langle H,I\rangle}]) \prec u\\
&\text{iff } \hat{\mathtt{op}}(s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})^H) \prec u\\
&\text{iff } H \models \mathtt{op}(s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})) \prec u\\
&\text{iff } H \models A\\
&\text{iff } \langle H, I\rangle \models A
\end{aligned}$$
□

Lemma 19. *Let Π be a program, $\mathcal{P}$ be the set of all predicate symbols in σ other than comparisons, $\mathcal{F}$ be the set of all function symbols corresponding set symbols. Then,*

$$\langle H, I\rangle \models_{ht} \tau\Pi \quad \textit{iff} \quad \langle H, I\rangle \models_{ht} gr^{\mathcal{PF}}_I(\tau^{cli}\Pi)$$

for every agg-ht-interpretation $\langle H, I\rangle$.

Proof. Recall that comparisons are not intensional in the definition of the stable models of a program, that is, they do not belong to $\mathcal{P}$. Then, it is easy to see that $\tau\Pi$ can be obtained from $gr^{\mathcal{PF}}_I(\tau^{cli}\Pi)$ by replacing each occurrence of $op(s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})) \prec u$, where op is an operation name, by its corresponding formula of the form of (19): Hence, it is enough to show that

$$\langle H, I\rangle \models_{ht} op(s^{cli}_{|E/\mathbf{X}|}(\mathbf{x}) \prec u) \quad \text{iff} \quad \langle H, I\rangle \models_{ht} (19)$$

This follows from Lemma 18. □

Lemma 20. *Let Π be a program, $\mathcal{P}$ be the set of all predicate symbols in σ other than comparisons, $\mathcal{F}$ be the set of all function symbols corresponding set symbols. Then,*

$$\langle H, I\rangle \models_{ht} \tau\Pi \quad \textit{iff} \quad \langle H, I\rangle \models_{ht} \tau^{cli}\Pi$$

for every agg-ht-interpretation $\langle H, I\rangle$.

Proof. Directly by Proposition 2 and Lemma 19. □

Lemma 21. *Let Π be a program, $\mathcal{P}$ be the set of all predicate symbols in σ other than comparisons, $\mathcal{F}$ be the set of all function symbols corresponding set symbols. Then,*

$$\mathcal{H} \models (\tau\Pi)^{\mathcal{I}} \quad \textit{iff} \quad \langle H, I\rangle \models_{ht} \tau^{cli}\Pi$$

for every agg-ht-interpretation $\langle H, I\rangle$.

Proof. Since $\tau\Pi$ is an infinitary formula of σ^p, the result follows by Lemma 20 and Proposition 3. □

Lemma 22. *Let $\langle H, I\rangle$ be an agg-ht-interpretation. Then, $H <^{\mathcal{PF}} I$ iff $H <^{\mathcal{P}\emptyset} I$.*

Proof. Right-to-left. $H <^{\mathcal{P}\emptyset} I$ means that there is a predicate symbol p such that $p^H \subset p^I$ and, thus, $H <^{\mathcal{PF}} I$ also holds. *Lert-to-right.* $H <^{\mathcal{PF}} I$ means that on of the following holds

- $p^H \subset p^I$ for some intensional predicate symbol p; or
- $f^H \neq f^I$ for some intensional function symbol f.

The first immediately implies that $H <^{\mathcal{P}\emptyset} I$ also holds. For the latter, f must be of the form $s^x_{|E/\mathbf{X}|}$ for some aggregate element E. Therefore, the set of the set of all tuples of form $\langle (t_1)^{\mathbf{XY}}_{\mathbf{xy}}, \dots, (t_k)^{\mathbf{XY}}_{\mathbf{xy}}\rangle$ such that I satisfies $(l_1)^{\mathbf{XY}}_{\mathbf{xy}} \wedge \dots \wedge (l_m)^{\mathbf{XY}}_{\mathbf{xy}}$ and the set of all tuples of form $\langle (t_1)^{\mathbf{XY}}_{\mathbf{xy}}, \dots, (t_k)^{\mathbf{XY}}_{\mathbf{xy}}\rangle$ such that $\langle H, I\rangle$ or H satisfies $(l_1)^{\mathbf{XY}}_{\mathbf{xy}} \wedge \dots \wedge (l_m)^{\mathbf{XY}}_{\mathbf{xy}}$ must be different. This means that $p^H \neq p^I$ for some predicate symbols p and, thus, $p^H \subset p^I$ and $H <^{\mathcal{P}\emptyset} I$ follow. □

Proposition 4. By Lemma 22, it follows that $H <^{\mathcal{PF}} I$ iff $H <^{\mathcal{P}\emptyset} I$. Then, the result follows because the latter holds iff $\mathcal{H} \subset \mathcal{I}$. □

Proof of Theorem 5. Assume that $\mathcal{I}$ is a fo-clingo answer set of Π. By definition, there is a I is an agg-stable model of $\tau^{cli}\Pi$. In its turn, this implies that I is an agg-model of $\tau^{cli}\Pi$ and there is no agg-ht-model $\langle H, I\rangle$ of $\tau^{cli}\Pi$ with $H <^{\mathcal{PF}} I$. By Lemma 21, it follows that $\mathcal{I}$ is a model of $(\tau\Pi)^{\mathcal{I}}$. Suppose, for the sake of contradiction, that there is $\mathcal{H} \subset \mathcal{I}$ such that $\mathcal{H} \models (\tau\Pi)^{\mathcal{I}}$. Let $\langle H, I\rangle$ be the agg-interpretation with $\mathcal{H}$ and $\mathcal{I}$ the set of ground atoms of σ^p satisfied by H and I, respectively. Then, by Proposition 4 and Lemma 21, it follows $H <^{\mathcal{PF}} I$ and $\langle H, I\rangle \models_{ht} \tau^{cli}\Pi$. This is a contradiction because there is no agg-ht-model $\langle H, I\rangle$ of $\tau^{cli}\Pi$ with $H <^{\mathcal{PF}} I$.

Conversely, assume that $\mathcal{I}$ is a clingo answer set of Π. By definition, $\mathcal{I}$ is a model of $\tau\Pi$ and there is no model $\mathcal{H}$ of $(\tau\Pi)^{\mathcal{I}}$ with $\mathcal{H} \subset \mathcal{I}$. By Lemma 21, the former implies that there is an agg-model I of $(\tau^{cli}\Pi)^{\mathcal{I}}$. Suppose, for the sake of contradiction, that there is some agg-ht-model $\langle H, I\rangle$ of $\tau^{cli}\Pi$ with $H <^{\mathcal{PF}} I$. By Lemma 21 and Proposition 4, this implies that $\mathcal{H}$ satisfies $(\tau^{cli}\Pi)^{\mathcal{I}}$ with $\mathcal{H} \subset \mathcal{I}$, which is a contradiction. □

Correspondence with `dlv`. A *dlv-literal* is either an atomic formula, a truth constant ($\top$ or $\bot$) or an expression of the forms $\llcorner A$, $\llcorner\llcorner A$ with A an atomic formula. A *dlv-implication* is an implication of the form $F_1 \to F_2$ where F_1 is a conjunction of dlv-literals and F_2 is either an atomic formula or the truth constant $\bot$.

For any formula F, by $n(F)$ we denote the result of replacing all occurrences of $\llcorner$ by $\neg$ in F. If F is an infinitary propositional formula, then $n(F)$ is standard.

Lemma 23. *Let F be an infinitary propositional formula. Then, $I \models F$ iff $\mathcal{I} \models n(F)$.*

Proof. By definition, $\mathcal{I} \models \neg G$ iff $\mathcal{I} \models \llcorner G$. Then, by induction, it follows that $\mathcal{I} \models F$ iff $\mathcal{I} \models n(F)$. Finally, since $n(F)$ is standard, by Proposition 3, we get that the letter holds iff $I \models F$. □

Given an implication F of the form $F_1 \to F_2$, by $p(F)$ we denote the implication $\llcorner\llcorner F_1 \to F_2$ and by $pn(F)$ we denote the implication $\llcorner\llcorner n(F_1) \to F_2$. Note that $I \models F$ iff $I \models p(F)$ iff $I \models pn(F)$. For ht-interpretation these equivalences do not hold, but we have the following interesting relation with the FLP-reduct.

Lemma 24. *Let $\langle H, I\rangle$ be an ht-interpretation and F be an infinitary propositional formula of the form $F_1 \to F_2$ with F_2 an atomic formula or a truth constant. Then, $\langle H, I\rangle \models_{ht} p(F)$ iff $\mathcal{I} \models F$ and $\mathcal{H} \models FLP(F, \mathcal{I})$.*

Proof. Left-to-rigth. Assume $\langle H, I\rangle \models_{ht} p(F)$. By Proposition 13, we get $I \models F$ and, thus $I \models p(F)$. By Proposition 3, this implies $\mathcal{I} \models F$. *Case 1.* $\mathcal{I} \not\models F_1$. Then, $\mathcal{I} \models F$ and $FLP(F, \mathcal{I}) = \top$, and the results immediately holds. *Case 2.* $\mathcal{I} \models F_1$. Then, $FLP(F, \mathcal{I}) = F$. If $\mathcal{H} \not\models F_1$, then the result follows immediately. Otherwise, $\mathcal{H} \models F_1$, and this implies $\langle H, I\rangle \models_{ht} \neg\neg F_1$ (by Propositions 3 and 14, and fact $\mathcal{I} \models F_1$). Since $\langle H, I\rangle \models_{ht} p(F)$, this implies that $\langle H, I\rangle \models_{ht} F_2$. Hence, F_2 is not $\bot$ and $\mathcal{H} \models F_2$. This means that $\mathcal{H} \models FLP(F, \mathcal{I})$.

Right-to-left. Assume $\mathcal{I} \models F$ and $\mathcal{H} \models FLP(F, \mathcal{I})$. By Proposition 3, we get $I \models F$ and, thus, $I \models p(F)$. We proceed by cases. *Case 1.* $\mathcal{I} \not\models F_1$. By Proposition 3, this implies $I \not\models \neg\neg F_1$ and, by Proposition 13, it follows $\langle H, I\rangle \not\models_{ht} \neg\neg F_1$. Hence, $\langle H, I\rangle \models_{ht} p(F)$. *Case 2.* $\mathcal{I} \models F_1$. Then, $I \models \neg\neg F_1$ (by Proposition 3) and $FLP(F, \mathcal{I}) = F$ (by definition). The latter implies $\mathcal{H} \models F$. If $\langle H, I\rangle \not\models \neg\neg F_1$, then the result follows because $I \models p(F)$. Otherwise, $\langle H, I\rangle \models_{ht} \neg\neg F_1$ and this implies $\mathcal{H} \models F_1$ (by Proposition 14). Since $\mathcal{H} \models F$, this implies that $\mathcal{H} \models F_2$. Then F_2 is not $\bot$ and $\langle H, I\rangle \models_{ht} F_2$. Since $I \models F$, this implies $\langle H, I\rangle \models_{ht} F$. $\square$

Lemma 25. *Let $\langle H, I\rangle$ be an ht-interpretation and L be a dlv-literal. Then, $\langle H, I\rangle \models_{ht} L$ iff $I \models L$ and $H \models L$.*

Proof. If L is an atomic formula or a truth constant, the result holds by definition. Otherwise, it follows by Proposition 14. $\square$

Lemma 26. *Let $\langle H, I\rangle$ be an ht-interpretation and F be a conjunction of dlv-literals. Then, $\langle H, I\rangle \models_{ht} F$ iff $I \models F$ and $H \models F$.*

Proof. Let $F = L_1 \wedge \ldots \wedge L_n$. Then, $\langle H, I\rangle \models_{ht} F$ iff (by definition) $\langle H, I\rangle \models_{ht} L_i$ for all $1 \leq i \leq n$ iff (Lemma 25) $H \models L_i$ and $I \models L_i$ for all $1 \leq i \leq n$ iff $H \models L_i$ for all $1 \leq i \leq n$ and $I \models L_i$ for all $1 \leq i \leq n$ iff (by definition) $H \models F$ and $I \models F$. $\square$

Lemma 27. *Let $\langle H, I\rangle$ be an ht-interpretation and F be a dlv-implication. Then, $\langle H, I\rangle \models_{ht} F$ iff $\langle H, I\rangle \models_{ht} pn(F)$.*

Proof. Let F be of the form $F_1 \to F_2$. Note that $I \models F_1$ iff $I \models \neg\neg n(F_1)$. Hence, $I \models F$ iff $I \models pn(F)$. Therefore, $\langle H, I\rangle \models_{ht} F_1 \to F_2$ iff $\langle H, I\rangle \not\models_{ht} F_1$ or $\langle H, I\rangle \models F_2$ iff (Lemma 26) $I \not\models F_1$ or $H \not\models F_1$ or $\langle H, I\rangle \models F_2$ iff (Lemma 23) $I \not\models n(F_1)$ or $H \not\models n(F_1)$ or $\langle H, I\rangle \models F_2$ iff (Proposition 14) $\langle H, I\rangle \not\models_{ht} \neg\neg n(F_1)$ or $\langle H, I\rangle \models F_2$ iff $\langle H, I\rangle \models_{ht} \neg\neg n(F_1) \to F_2$. $\square$

Lemma 28. *Let R be a rule and $\langle H, I\rangle$ be an agg-ht-interpretation. Then,*

$$\langle H, I\rangle \models_{ht} p(\tau R) \quad \textit{iff} \quad \langle H, I\rangle \models_{ht} pn(R')$$

where R' is $gr_I^{\mathcal{PF}}(\tau^{dlv} R)$.

Proof. We can see that $p(\tau R)$ and $pn(R')$ are of the form of $\neg\neg F_1 \to F_2$ and $\neg\neg F_1' \to F_2$, respectively, with F_1 and F_1' differing only in the translation of aggregates, with the former containing formula (19) where the latter contains an atom of the form $\mathsf{op}(s^{dlv}_{|E/\mathbf{x}|}(\mathbf{x})) \prec u$. By Proposition 14, it follows that $\langle H, I\rangle \models_{ht} \neg\neg F_1$ iff $I \models F_1$ and $H \models F_1$, and $\langle H, I\rangle \models_{ht} \neg\neg F_1'$ iff $I \models F_1'$ and $H \models F_1'$. Finally, by Lemma 17, we get $I \models F_1$ iff $I \models F_1'$ and $H \models F_1$ iff $H \models F_1'$. $\square$

Lemma 29. *Let R be a rule and I be an agg-interpretation. Then,*

$$I \models \tau R \quad \textit{iff} \quad I \models \tau^{dlv} R$$

Proof. Let R' be the result of $gr_I^{\mathcal{PF}}(\tau^{dlv} R)$. Then, it follows that $I \models \tau R$ iff $I \models_{ht} p(\tau R)$ iff $\langle I, I\rangle \models_{ht} p(\tau R)$ iff (Lemma 28) $\langle I, I\rangle \models_{ht} pn(R')$ iff $I \models_{ht} pn(R')$ iff $I \models_{ht} R'$ iff (Proposition 16) $I \models \tau^{dlv} R$. $\square$

Lemma 30. *Let R be a rule and $\langle H, I\rangle$ be an agg-ht-interpretation. Then, the following two conditions are equivalent*

- *$\mathcal{I} \models \tau R$ and $\mathcal{H} \models FLP(\tau R, \mathcal{I})$, and*
- *$\langle H, I\rangle \models_{ht} \tau^{dlv} R$.*

Proof. Let R' be $gr_I^{\mathcal{PF}}(\tau^{dlv} R)$. By Proposition 2, we get $\langle H, I\rangle \models_{ht} \tau^{dlv} R$ iff $\langle H, I\rangle \models R'$. Furthermore, R' is a dlv-implication and, by Lemmas 24 and 27 we respectively get:

- $\langle H, I\rangle \models_{ht} p(\tau R)$ iff $\mathcal{I} \models F$ and $\mathcal{H} \models FLP(\tau R, \mathcal{I})$,
- $\langle H, I\rangle \models_{ht} R'$ iff $\langle H, I\rangle \models_{ht} pn(R')$.

Hence, it remains to be shown

$$\langle H, I\rangle \models_{ht} p(\tau R) \quad \text{iff} \quad \langle H, I\rangle \models_{ht} pn(R')$$

which follows by Lemma 28. $\square$

Proof of Theorem 6. Assume that $\mathcal{I}$ is a fo-dlv answer set of Π. By definition, there is a I is an agg-stable model of $\tau^{dlv}\Pi$. In its turn, this implies that I is an agg-model of $\tau^{dlv}\Pi$ and there is no agg-ht-model $\langle H, I\rangle$ of $\tau^{dlv}\Pi$ with $H <^{\mathcal{PF}} I$. By Lemma 29, it follows that $\mathcal{I}$ is a model of $\tau\Pi$. Suppose, for the sake of contradiction, that there is $\mathcal{H} \subset \mathcal{I}$ such that $\mathcal{H} \models FLP(\tau\Pi, \mathcal{I})$. Let $\langle H, I\rangle$ be the agg-ht-interpretation with $\mathcal{H}$ and $\mathcal{I}$ the set of ground atoms of σ^p satisfied by H and I, respectively. Then, by Proposition 4 and Lemma 30, it follows $H <^{\mathcal{PF}} I$ and $\langle H, I\rangle \models_{ht} \tau^{dlv}\Pi$. This is a contradiction because there is no agg-ht-model $\langle H, I\rangle$ of $\tau^{dlv}\Pi$ with $H <^{\mathcal{PF}} I$.

Conversely, assume that $\mathcal{I}$ is a dlv answer set of Π. By definition, $\mathcal{I}$ is a model of $\tau\Pi$ and there is no model $\mathcal{H}$ of $FLP(\tau\Pi, \mathcal{I})$ with $\mathcal{H} \subset \mathcal{I}$. By Lemma 29, the former implies that there is an agg-model I of $\tau^{dlv}\Pi$. Suppose, for the sake of contradiction, that there is some agg-ht-model $\langle H, I\rangle$ of $\tau^{dlv}\Pi$ with $H <^{\mathcal{PF}} I$. By Lemma 30 and Proposition 4, this implies that $\mathcal{H}$ satisfies $FLP(\tau^{dlv}\Pi, \mathcal{I})$ with $\mathcal{H} \subset \mathcal{I}$, which is a contradiction. $\square$

Proofs of Section Strong Equivalence

Lemma 31. *If Γ_1 and Γ_2 have the same agg-ht-models, then they have the same agg-stable models.*

Proof. Since Γ_1 and Γ_2 have the same agg-ht-models, they also have the same agg-models. Suppose, for the sake of contradiction, that they have different agg-stable models. Assume, without loss of generality, that I is an agg-stable model of Γ_1 but not of Γ_2. Since Γ_1 and Γ_2 have the same models and I is a model of Γ_1, it follows that I is a model of Γ_2. Since I is not an agg-stable model of Γ_2, there is a agg-ht-model $\langle H, I\rangle$ of Γ_2 such that $H <^{\mathcal{PF}} I$. Since Γ_1 and Γ_2 have the same agg-ht-models, this implies that $\langle H, I\rangle$ is also an agg-ht-model of Γ_1, which is a contradiction with the assumption that I is a agg-stable model of Γ_1. $\square$

For any interpretation I, by Δ_I we denote the program containing all facts of the form "$p(\mathbf{t})$" such that $I \models \tau^{cli}(p(\mathbf{t}))$ with $p \in \mathcal{P}$. Similarly, for an ht-interpretation $\langle H, I\rangle$, by $\Delta_{\langle H,I\rangle}$ we denote the program containing all facts in Δ_I plus all rules of the form "$p(\mathbf{t})$:- $q(\mathbf{u})$" such that $I \models \tau^{cli}(p(\mathbf{t}) \wedge q(\mathbf{u}))$ and $H \not\models \tau^{cli}(p(\mathbf{t}) \vee q(\mathbf{u}))$ with $p, q \in \mathcal{P}$.

Lemma 32. *Take any two sets of sentences, Γ_1 and Γ_2 and let I be a agg-model of Γ_1 that does not satisfy Γ_2. Then, I is an agg-stable model of $\Gamma_1 \cup \tau^x\Delta_I$, but not of $\Gamma_2 \cup \tau^x\Delta_I$ with $x \in \{cli, \mathtt{dlv}\}$.*

Proof. By the definition, it follows that $(A) \in \Pi$ iff $I \models \tau^{cli} A$ iff $I \models \tau^{dlv} A$. Note that $\tau^{cli}(A) = \tau^{dlv}(A)$ for all $A \in \Delta_I$. Thus, I is a model of $\tau^x\Pi$. Furthermore, there is no $\langle H, I\rangle$ with $H <^{\mathcal{P}\emptyset} I$ satisfies $\tau^x\Pi$. By Lemma 22, this implies that there is no agg-interpretation H with $H <^{\mathcal{PF}} I$ such that $\langle H, I\rangle$ satisfies $\tau^x\Pi$. Since I is also a model of Γ_1, it follows that I is a model of $\Gamma_1 \cup \tau^x\Pi$ and, thus, it a agg-stable model of $\Gamma_1 \cup \tau^x\Pi$. Since I does not satisfy Γ_2, it follows that I is not an agg-stable model of $\Gamma_2 \cup \tau^x\Pi$. $\square$

Lemma 33. *Take any two sets of sentences, Γ_1 and Γ_2 with the same classical models and let $\langle H, I\rangle$ be an agg-ht-model of Γ_1 that does not satisfy Γ_2. Then, I is an agg-stable model of $\Gamma_3 \cup \tau^x\Delta_{\langle H,I\rangle}$, but not of $\Gamma_1 \cup \tau^x\Delta_{\langle H,I\rangle}$ with $x \in \{cli, \mathtt{dlv}\}$.*

Proof. First note that $\tau^{cli}(\Delta_{\langle H,I\rangle}) = \tau^{dlv}(\Delta_{\langle H,I\rangle})$. Hence, in the following, we do not distinguish between the two. Furthermore, I satisfies $\tau^x(\Delta_{\langle H,I\rangle})$ because, by definition, it satisfies the consequent of every rule in $\Delta_{\langle H,I\rangle}$. Hence, I is a model of $\Gamma_1 \cup \tau^x(\Delta_{\langle H,I\rangle})$ and $\Gamma_2 \cup \tau^x(\Delta_{\langle H,I\rangle})$. To see that I is an agg-stable model of $\Gamma_2 \cup \tau^x(\Delta_{\langle H,I\rangle})$, suppose for the sake of contradiction that there is an agg-interpretation J with $J <^{\mathcal{PF}} I$ such that $\langle J, I\rangle$ satisfies $\Gamma_2 \cup \tau^x(\Delta_{\langle H,I\rangle})$. This implies that $\langle J, I\rangle$ satisfies $\tau^x(\Delta_I)$ and, thus that $H \leq^{\mathcal{P}\emptyset} J$. Furthermore, H must be different from J because $\langle H, I\rangle$ does not satisfy Γ_2 and $\langle J, I\rangle$ does. By Lemma 22, it follows that $J <^{\mathcal{PF}} I$ implies $J <^{\mathcal{P}} I$. Hence, $H <^{\mathcal{P}} J <^{\mathcal{P}} I$. Let $p(\mathbf{t})$ be an atom with $p \in \mathcal{P}$ such that $J \models \tau^x(p(\mathbf{t}))$ and $H \not\models \tau^x(p(\mathbf{t}))$. Let $q(\mathbf{u})$ be an atom with $p \in \mathcal{P}$ such that $I \models \tau^x(q(\mathbf{u}))$ and $J \not\models \tau^x(q(\mathbf{u}))$. Therefore, rule "$q(\mathbf{u})$:- $p(\mathbf{t})$" belongs to $\Delta_{\langle H,I\rangle}$. Let this rule be named R. Then, $\langle J, I\rangle$ does not satisfies $\tau^x R$. This implies that $\langle J, I\rangle$ does not satisfy $\Gamma_2 \cup \tau^x(\Delta_{\langle H,I\rangle})$, which is a contradiction with the assumption. Hence, I is an agg-stable model of $\Gamma_2 \cup \tau^x(\Delta_{\langle H,I\rangle})$.

It remains to be shown that I is not an agg-stable model of $\Gamma_1 \cup \tau^x(\Delta_{\langle H,I\rangle})$. We show that $\langle H, I\rangle$ satisfies $\Gamma_1 \cup \tau^x(\Delta_{\langle H,I\rangle})$. It is a model of Δ_I. Furthermore, it also satisfies every rule R in $\Delta_{\langle H,I\rangle}$ of the form "$q(\mathbf{u})$:- $p(\mathbf{t})$" because $\langle H, I\rangle \not\models_{ht} \tau^x(p(\mathbf{t}))$ and $I \models \tau^x(q(\mathbf{u}))$. Hence, $\langle H, I\rangle$ satisfies $\Gamma_1 \cup \tau^x(\Delta_{\langle H,I\rangle})$ and, thus, I is not an agg-stable model of $\Gamma_1 \cup \tau^x(\Delta_{\langle H,I\rangle})$. $\square$

Lemma 34. *If Γ_1 and Γ_2 do not have the same agg-ht-models, then there is some program Δ without aggregates nor double negation such that $\Gamma_1 \cup \tau^x(\Delta)$ and $\Gamma_2 \cup \tau^x(\Delta)$ do not have the same agg-stable models, with $x \in \{cli, \mathtt{dlv}\}$.*

Proof. We proceed by cases. *Case 1.* Γ_1 and Γ_2 do not have the same agg-models. Assume without loss of generality that I is an agg-model of Γ_1 but not of Γ_2. By Lemma 32, it follows that I is an agg-stable model of $\Gamma_1 \cup \tau^x(\Delta_I)$ but not of $\Gamma_2 \cup \tau^x(\Delta_I)$. *Case 2.* Γ_1 and Γ_2 have the same agg-models. By Lemma 33, it follows that I is an agg-stable model of $\Gamma_2 \cup \tau^x(\Delta_{\langle H,I\rangle})$ but not of $\Gamma_1 \cup \tau^x(\Delta_{\langle H,I\rangle})$. In both cases, $\Gamma_2 \cup \tau^x(\Delta_{\langle H,I\rangle})$ and $\Gamma_1 \cup \tau^x(\Delta_{\langle H,I\rangle})$ have different agg-stable models. $\square$

Lemma 35. *If $\tau^x(\Pi_1)$ and $\tau^x(\Pi_2)$ have the same agg-ht-models, then $\tau^x(\Pi_1 \cup \Delta)$ and $\tau^x(\Pi_2 \cup \Delta)$ they have the same agg-stable models, with $x \in \{cli, \mathtt{dlv}\}$, for any program Δ.*

Proof. Assume that $\tau^x(\Pi_1)$ and $\tau^x(\Pi_2)$ have the same agg-ht-models. Then, $\tau^x(\Pi_1 \cup \Delta) = \tau^x(\Pi_1) \cup \tau^x(\Delta)$ has the same agg-ht-model as $\tau^x(\Pi_2 \cup \Delta) = \tau^x(\Pi_2) \cup \tau^x(\Delta)$. By Lemma 31, this implies that both have same agg-stable models. $\square$

Lemma 36. *If $\tau^x(\Pi_1)$ and $\tau^x(\Pi_2)$ do not have the same agg-ht-models, then there is some program Δ without aggregates nor double negation such that $\tau^x(\Pi_1 \cup \Delta)$ and $\tau^x(\Pi_2 \cup \Delta)$ do not have the same agg-stable models, with $x \in \{cli, \mathtt{dlv}\}$.*

Proof. In this case, by Lemma 34 with $\Gamma_1 = \tau^x(\Pi_1)$ and $\Gamma_2 = \tau^x(\Pi_2)$ by noting that $\tau^x(\Pi_1 \cup \Delta) = \tau^x(\Pi_1) \cup \tau^x(\Delta)$ and $\tau^x(\Pi_2 \cup \Delta) = \tau^x(\Pi_2) \cup \tau^x(\Delta)$. $\square$

Proof of Theorem 7. Assume that $\tau^{cli}(\Pi_1)$ and $\tau^{cli}(\Pi_2)$ have the same agg-ht-models and let Δ be a program. By Lemma 35, it follows that $\tau^{cli}(\Pi_1 \cup \Delta)$ and $\tau^{cli}(\Pi_2 \cup \Delta)$ have the same agg-stable models. This implies that $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have the same fo-clingo answer sets. By Theorem 7, this implies that $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have the same clingo answer sets and, thus, are strongly equivalent under the `clingo` semantics.

Conversely, suppose $\tau^{cli}(\Pi_1)$ and $\tau^{cli}(\Pi_2)$ do not have the same agg-ht-models. By Lemma 36, there is a program Δ such that $\tau^{cli}(\Pi_1 \cup \Delta)$ and $\tau^{cli}(\Pi_2 \cup \Delta)$ do not have the same agg-stable models, and, thus they do not have the same fo-clingo answer sets. By Theorem 5, this implies $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have different clingo answer sets. Hence, Π_1 and Π_2 are not strongly equivalent under the `clingo` semantics. $\square$

Proof of Theorem 8. The proof is analogous to the one of Theorem 7. Assume that $\tau^{dlv}(\Pi_1)$ and $\tau^{dlv}(\Pi_2)$ have the same agg-ht-models and let Δ be a program. By Lemma 35, it follows that $\tau^{dlv}(\Pi_1 \cup \Delta)$ and $\tau^{dlv}(\Pi_2 \cup \Delta)$ have the same agg-stable models. This implies that $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have the same fo-dlv answer sets. By Theorem 8, this implies that $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have the same dlv answer sets and, thus, are strongly equivalent under the `dlv` semantics.

Conversely, suppose $\tau^{dlv}(\Pi_1)$ and $\tau^{dlv}(\Pi_2)$ do not have the same agg-ht-models. By Lemma 36, there is a program Δ such that $\tau^{dlv}(\Pi_1 \cup \Delta)$ and $\tau^{dlv}(\Pi_2 \cup \Delta)$ do not have the same agg-stable models, and, thus they do not have the same fo-dlv answer sets. By Theorem 6, this implies $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have different dlv answer sets. Hence, Π_1 and Π_2 are not strongly equivalent under the `dlv` semantics. $\square$

Proof of Theorem 9. The proof is similar to the those of Theorems 7 and 8. Assume that $\tau^{cli}(\Pi_1)$ and $\tau^{dlv}(\Pi_2)$ have the same agg-ht-models and let Δ be a program such that $\tau^{cli}(\Delta)$ and $\tau^{dlv}(\Delta)$ have the same agg-ht-models. Then, $\tau^{cli}(\Pi_1 \cup \Delta) = \tau^{cli}(\Pi_1) \cup \tau^{cli}(\Delta)$ and $\tau^{dlv}(\Pi_2) \cup \Delta = \tau^{dlv}(\Pi_2) \cup \tau^{dlv}(\Delta)$ have the same agg-ht-models. By Lemma 31, it follows that $\tau^{dlv}(\Pi_1 \cup \Delta)$ and $\tau^{dlv}(\Pi_2 \cup \Delta)$ have the same agg-stable models. This implies that the fo-clingo answer sets of $\Pi_1 \cup \Delta$ and the same fo-dlv answer sets of $\Pi_2 \cup \Delta$ coincide. By Theorems 7 and 8, this implies that clingo answer sets $\Pi_1 \cup \Delta$ coincide with the dlv answer sets of $\Pi_2 \cup \Delta$.

Conversely, suppose $\tau^{cli}(\Pi_1)$ and $\tau^{dlv}(\Pi_2)$ do not have the same agg-ht-models. By Lemma 34, there is a program Δ without aggregates nor double negation such that $\tau^{cli}(\Pi_1 \cup \Delta) = \tau^{cli}(\Pi_1) \cup \tau^{x}(\Delta)$ and $\tau^{dlv}(\Pi_2 \cup \Delta) = \tau^{dlv}(\Pi_2) \cup \tau^{x}(\Delta)$ do not have the same agg-stable models. Note that, since Δ does not contain aggregates nor double negation, it follows that $\tau^{cli}(\Delta) = \tau^{dlv}(\Delta)$. Hence, the fo-clingo answer sets of $\Pi_1 \cup \Delta$ do not coincide with the fo-dlv answer sets of $\Pi_2 \cup \Delta$. By Theorems 7 and 8, this implies that the clingo answer sets of $\Pi_1 \cup \Delta$ are different from the dlv answer sets of $\Pi_2 \cup \Delta$. $\square$

Proof of Section Strong Equivalence using Classical Logic

Lemma 37. $\hat{t}^{I^H} = t^I$

Proof. If t is of the form $f()$ with f an extensional function, then $\hat{t} = f()$ and the result follows by definition. If t is of the form $f()$ with f an intensional function, then $\hat{t} = \hat{f}()$ and $\hat{f}^{I^H} = f^I$ follows by definition. The rest of the proof follows by induction on the structure of t in a similar way. $\square$

Lemma 38. $I^H \models \hat{p}(\hat{\mathbf{t}})$ *iff* $I \models p(\mathbf{t})$

Proof. By Lemma 37, we get $\hat{\mathbf{t}}^{I^H} = \mathbf{t}^I$. Then, $I^H \models \hat{p}(\hat{\mathbf{t}})$ iff $I^H \models \hat{p}(\hat{\mathbf{t}})$ iff $I^H \models \hat{p}((\hat{\mathbf{t}}^{I^H})^*)$ iff $I^H \models \hat{p}((\hat{\mathbf{t}}^I)^*)$ iff $I \models p((\hat{\mathbf{t}}^I)^*)$ iff $I \models p(\mathbf{t})$. $\square$

Lemma 39. $I^H \models \hat{F}$ *iff* $I \models F$.

Proof. We will consider the case of a ground atom A of the form $p(\mathbf{t})$; extension to arbitrary sentences by induction is straightforward. If p is extensional then $\hat{A}$ is $p(\hat{\mathbf{t}})$; $I^H \models p(\hat{\mathbf{t}})$ iff $I \models p(\mathbf{t})$ follows by Lemma 37 and the fact that I^H interprets extensional symbols in the same way as I. If p is intensional then $\hat{A}$ is $\hat{p}(\hat{\mathbf{t}})$, and the result follows by Lemma 38. $\square$

Lemma 40. *Let t is a term of σ. Then,* $t^{I^H} = t^H$

Proof. If t is of the form $f()$ with f an extensional function, then $t = f()$ and we have $f^{I^H} = f^I = f^H$. If t is of the form $f()$ with f an intensional function, then $t = f()$ and $f^{I^H} = f^H$ follows by definition. The rest of the proof follows by induction on the structure of t in a similar way. $\square$

Lemma 41. *Let F be a formula of σ. Then,* $I^H \models n(F)$ *iff* $H \models F$.

Proof. The proof is by induction on the number of connectives and quantifiers in F. We consider below the more difficult cases when F is an atomic formula or the negation $\llcorner$.

Case 1: F is an atomic formula $p(\mathbf{t})$. Let $J = I^H$. Then, $n(F)$ is also $p(\mathbf{t})$. By Lemma 40, we have $\mathbf{t}^J = \mathbf{t}^H$. Let $\mathbf{d}$ be the common value of $\mathbf{t}^{I^H}$ and $\mathbf{t}^H$. *Case 1.1:* p is intensional. The left-hand side of (**??**) is equivalent to $I^H \models p(\mathbf{d}^*)$ and consequently to $H \models p(\mathbf{d}^*)$. The right-hand side to is equivalent $H \models p(\mathbf{d}^*)$ as well. *Case 1.2:* p is extensional. Each of two sides of (**??**) is equivalent to $I \models p(\mathbf{d}^*)$.

Case 2: F is $\llcorner G$. Then, $n(F)$ is $\neg n(G)$; we need to check that $I^H \models \neg n(G)$ iff $H \models \neg G$. This is equivalent to check $I^H \not\models n(G)$ iff $H \not\models G$, which follows by induction hypothesis. $\square$

Proof of Proposition 10. We prove it for a formula F and its extension to theories is straightforward. The proof is by induction on the number of propositional connectives and quantifiers in F. We consider below the more difficult cases when F is an atomic formula, one of the negations, or an implication.

Case 1: F is an atomic formula $p(\mathbf{t})$. Then γF is $F \wedge \hat{F}$; we need to check that

$$I^H \models F \wedge \hat{F} \text{ iff } \langle H, I\rangle \models_{ht} F.$$

On the one hand, $\langle H, I\rangle \models_{ht} F$ iff (by definition) $I \models F$ and $H \models F$ iff (Lemma 39) $I^H \models F$ and $I \models F$. On the other hand, $I^H \models F \wedge \hat{F}$ iff $I^H \models F$ and $I^H \models \hat{F}$ iff $I \models F$ and $I^H \models F$. Hence, it is enough to show that

$$I^H \models F \text{ iff } H \models F.$$

which follows by Lemma 41.

Case 2: F is $\neg G$. Then γF is $\neg \hat{G}$; we need to check that

$$I^H \not\models \hat{G} \text{ iff } \langle H, I\rangle \models_{ht} \neg G.$$

By Lemma 39, the left-hand side is equivalent to $I \not\models G$. By Proposition 13, the right-hand side is equivalent to $I \not\models G$ as well.

Case 3: F is $\llcorner G$. Then γF is $\neg n(G) \wedge \neg \hat{G}$; we need to check that

$$I^H \models \neg n(G) \wedge \neg \hat{G} \text{ iff } \langle H, I\rangle \models_{ht} \llcorner G.$$

The left-hand side is equivalent to the conjunction of $I^H \not\models n(G)$ and $I^H \not\models \hat{G}$. By Lemmas 41 and 39, this is equivalent to the conjunction of $H \not\models G$ and $I \not\models G$, which by definition, is equivalent to the right-hand side.

Case 4: F is of the form $F_1 \to F_2$. Then γF is the conjunction $(\gamma F_1 \to \gamma F_2) \wedge (\hat{F_1} \to \hat{F_2})$, so that the condition $I^H \models \gamma F$ holds iff

$$I^H \not\models \gamma F_1 \text{ or } I^H \models \gamma F_2 \tag{21}$$

and

$$I^H \models \hat{F_1} \to \hat{F_2}. \tag{22}$$

By the induction hypothesis, (21) is equivalent to

$$\langle H, I\rangle \not\models_{ht} F_1 \text{ or } \langle H, I\rangle \models_{ht} F_2. \tag{23}$$

By Lemma 39, we get that (22) is equivalent to

$$I \models G \to H. \tag{24}$$

By definition, the conjunction of (23) and (24) is equivalent to $\langle H, I\rangle \models_{ht} F_1 \to F_2$. □

Lemma 42. *An interpretation of the signature $\hat{\sigma}$ satisfies HT iff it can be represented in the form I^H for some ht-interpretation $\langle H, I\rangle$.*

Proof. For the if-part, take any sentence of the form of $\forall \mathbf{X}(p(\mathbf{X}) \to \hat{p}(\mathbf{X}))$ from HT. We need to show that I^H satisfies all sentences of the form $p(\mathbf{d}^*) \to \hat{p}(\mathbf{d}^*)$. Assume that $I^H \models p(\mathbf{d}^*)$. Then, $\mathbf{d}^* \in p^H \subseteq p^I$, and consequently $I \models p(\mathbf{d}^*)$, which is equivalent to $I^H \models \hat{p}(\mathbf{d}^*)$.

For the only-if part, take any interpretation J of $\hat{\sigma}$ that satisfies HT. Let I be the interpretation of σ that has the same domains as J, interprets extensional symbols in the same way as J, interprets every intensional function f in accordance with the condition $f^I = \hat{f}^J$. and interprets every intensional p in accordance with the condition

$$I \models p(\mathbf{d}^*) \text{ iff } J \models \hat{p}(\mathbf{d}^*). \tag{25}$$

Similarly, let H be the interpretation of σ that has the same domains as J, interprets extensional symbols in the same way as J, interprets every intensional function f in accordance with the condition $f^H = f^J$, and interprets every intensional p in accordance with the condition

$$H \models p(\mathbf{d}^*) \text{ iff } J \models p(\mathbf{d}^*). \tag{26}$$

Then, I and H agree on all extensional symbols. Furthermore, since J satisfies AGG, J satisfies $\hat{p}(\mathbf{d}^*)$ for every atom $p(\mathbf{d}^*)$ satisfied by J. By (25) and (26), it follows that all atoms satisfied by H are satisfied by I. It follows that $\langle H, I\rangle$ is an ht-interpretation. Let us show that $I^H = J$. Each of the interpretations I^H and J has the same domains as I and interprets all extensional symbols in the same way as I. Furthermore, for intensional function symbols, we have

$$\hat{f}^{I^H} = f^I = \hat{f}^J$$
$$f^{I^H} = f^H = f^J.$$

For everyintensional p and any tuple $\mathbf{d}$ of elements of appropriate domains, each of the conditions $I^H \models p(\mathbf{d}^*)$, $J \models p(\mathbf{d}^*)$ is equivalent to $H \models p(\mathbf{d}^*)$, and each of the conditions $I^H \models \hat{p}(\mathbf{d}^*)$, $J \models \hat{p}(\mathbf{d}^*)$ is equivalent to $I \models p(\mathbf{d}^*)$. □

Lemma 43. *Let E be an aggregate element of the form* (5) *with global variables $\mathbf{X}$ and local variables $\mathbf{Y}$ and let $\langle H, I\rangle$ be a standard ht-interpretation. Let $\mathbf{x}$ be a list of ground terms of sort s_{gen} of the same length as $\mathbf{X}$, d_{tuple} a domain element of sort tuple, $l'_i = (l_i)^{\mathbf{X}}_{\mathbf{x}}$ and $t'_i = (t_i)^{\mathbf{X}}_{\mathbf{x}}$. Then, I^H satisfies*

$$d^*_{tuple} \in \hat{s}^x_{|E|}(\mathbf{x}) \leftrightarrow \exists \mathbf{Y}\, \hat{F} \tag{27}$$

where F is $(d^*_{tuple} = tuple(t'_1, \dots, t'_m) \wedge l'_1 \wedge \dots \wedge l'_n)$ iff the following two conditions are equivalent:

1. *d_{tuple} belongs to $\hat{s}^x_{|E|}(\mathbf{x})^I$,*
2. *there is a list $\mathbf{c}$ of domain elements of sort s_{gen} of the same length as $\mathbf{Y}$ such that $d_{tuple} = \langle (t''_1)^I, \dots, (t''_m)^I\rangle$ and I satisfies $l''_1 \wedge \dots \wedge l''_n$ with $t''_i = (t'_i)^{\mathbf{Y}}_{\mathbf{y}}$ and $l''_i = (l'_i)^{\mathbf{Y}}_{\mathbf{y}}$ and $\mathbf{y} = \mathbf{c}^*$.*

Proof. I satisfies (27) iff condition 1 is equivalent to:

$$I \text{ satisfies } \exists \mathbf{Y}\, (d^*_{tuple} = tuple(t'_1, \dots, t'_m) \wedge \hat{l_1}' \wedge \dots \wedge \hat{l_n}')$$

Furthermore, the latter holds iff there is a list $\mathbf{c}$ of domain elements of sort s_{gen} such that

$$d_{tuple} = tuple(t''_1, \dots, t''_m)^I = \langle (t''_1)^I, \dots, (t''_m)^I\rangle$$

and I satisfies $l''_1 \wedge \dots \wedge l''_n$ with $\mathbf{y} = \mathbf{c}^*$ iff condition 2 holds. □

Lemma 44. *Let $\langle H, I\rangle$ be a standard ht-interpretation. Then, I is an agg-interpretation iff I^H satisfies* (16) *for every $E/\mathbf{X}$ in $\mathcal{S}$.*

Proof. By Lemma 39, it follows that I^H satisfies (16) iff I satisfies $\forall \mathbf{X} T\big(T \in s^x_{|E/\mathbf{X}|}(\mathbf{X}) \leftrightarrow \exists \mathbf{Y} F^x\big)$. Then, I satisfies

$$\forall T\big(T \in s^x_{|E/\mathbf{X}|}(\mathbf{x}) \leftrightarrow \exists \mathbf{Y} (F^x)^{\mathbf{X}}_{\mathbf{x}}\big) \tag{28}$$

iff $s^x_{|E/\mathbf{X}|}(\mathbf{x})^I$ is the set of all tuples d_{tuple} such that, there is $\mathbf{y}$ such that I satisfies

$$d^*_{tuple} = \mathit{tuple}(t'_1, \dots, t'_k) \wedge \tau^x(l'_1) \wedge \cdots \wedge \tau^x(l'_m) \quad (29)$$

with $t'_i = (t_i)^{\mathbf{XY}}_{\mathbf{xy}}$ and $l'_i = (l_i)^{\mathbf{XY}}_{\mathbf{xy}}$ iff $s^x_{|E/\mathbf{X}|}(\mathbf{x})^I$ is the set of all tuples of the form

$$\langle t'_1, \dots, t'_k \rangle$$

such that I satisfies

$$\tau^x(l'_1) \wedge \cdots \wedge \tau^x(l'_m)$$

and the result holds. □

Lemma 45. *Let $\langle H, I\rangle$ be a standard ht-interpretation such that I is an agg-interpretation. Then, $\langle H, I\rangle$ satisfies the condition of being an agg-interpretation for every term of the form $s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})$ iff I^H satisfies* (17) *for every $E/\mathbf{X}$ in $\mathcal{S}$.*

Proof. I^H satisfies (17) iff the following two conditions are equivalent for every tuple $\mathbf{d}$ of domain element of sort general and every domain element d_{tuple} of sort tuple:

- I^H satisfies $d^*_{tuple} \in s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})$, and
- I^H satisfies $\exists \mathbf{Y} \gamma(F^{cli})$.

By Lemma 41 and Proposition 10 respectively, these two conditiosn are equivalent to

- H satisfies $d^*_{tuple} \in s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})$, and
- $\langle H, I\rangle$ satisfies $\exists \mathbf{Y} F^{cli}$.

The rest of the proof is analogous to that of Lemma 44. □

Lemma 46. *Let $\langle H, I\rangle$ be a standard ht-interpretation such that I is an agg-interpretation. Then, $\langle H, I\rangle$ satisfies the condition of being an agg-interpretation for every term of the form $s^{dlv}_{|E/\mathbf{X}|}(\mathbf{x})$ iff I^H satisfies* (18) *for every $E/\mathbf{X}$ in $\mathcal{S}$.*

Proof. I^H satisfies (18) iff the following two conditions are equivalent for every tuple $\mathbf{d}$ of domain element of sort general and every domain element d_{tuple} of sort tuple:

- I^H satisfies $d^*_{tuple} \in s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})$, and
- I^H satisfies $\exists \mathbf{Y} n(F^{cli})$.

By Lemma 41, these two conditiosn are equivalent to

- H satisfies $d^*_{tuple} \in s^{cli}_{|E/\mathbf{X}|}(\mathbf{x})$, and
- H satisfies $\exists \mathbf{Y} F^{dlv}$.

The rest of the proof is analogous to that of Lemma 44. □

Lemma 47. *Every standard ht-interpretation $\langle H, I\rangle$ is an agg-interpretation iff I^H satisfies AGG.*

Proof. Directly by Lemmas 44, 45 and 46. □

Proof of Proposition 11. For the if-part, take any interpretation J of $\hat{\sigma}$ that satisfies HT and AGG. Since J satisfies HT, by Lemma 42, there is an ht-interpretation $\langle H, I\rangle$ such that $I^H = J$. Furthermore, since J satisfies AGG, by Proposition 47, $\langle H, I\rangle$ is an agg-interpretation. For the only-if part, take any agg-ht-interpretation $\langle H, I\rangle$. By Proposition 47, I^H satisfies AGG. By Lemma 42, I^H satisfies HT. □

Proof of Theorem 12. First note that, since Π_1 and Π_2 are finite, there is a finite number of sentences in HT, AGG, $\gamma\tau^x\Pi_1$, and $\gamma\tau^x\Pi_2$. Hence, the conjunction of all the sentences in them is well-defined. Let G_i be the conjunction of all sentences in $\tau^x\Pi_i$, so that F_i is γG_i. We want to show that Π_1 is strongly equivalent to Π_2 iff

for every standard model J of $HT \cup AGG$
$J \models \gamma G_1$ iff $J \models \gamma G_2$.

By Proposition 11, this is further equivalent to the condition

for every agg-ht-interpretation $\langle H, I\rangle$
$I^H \models \gamma G_1$ iff $I^H \models \gamma G_2$.

By Proposition 10, this is further equivalent to stating that

for every agg-ht-interpretation $\langle H, I\rangle$
$\langle H, I\rangle \models_{ht} G_1$ iff $\langle H, I\rangle \models_{ht} G_2$.

that is, to the condition

$\tau^x\Pi_1$ and $\tau^x\Pi_2$ have the same agg-ht-models.

By Theorems 7 and 8, this is equivalent to the condition that Π_1 and Π_2 are strongly equivalent under the `clingo`and `dlv`semantics, respectively. □