Appendix A Proofs and additional details

In the following, we give proofs of the theorems and propositions of the paper. Furthermore, some more detailed explanations are included.

Theorems and Propositions have the same numbering as in the original document. Auxiliary Theorems etc. are numbered with roman numerals.

Expressing unweighted formulas over semirings If we have a semiring $\mathscr{R} = (R, \oplus, \otimes, e_\oplus, e_\otimes)$, where the addition $\oplus$ is invertible, then we can translate any formula ϕ without weights into one with weights $\tau(\phi)$ that does not use the boolean connectives and such that

$$[\![\tau(\phi)]\!]^\sigma_{\mathscr{R}}(\mathscr{I}_w) = e_\otimes \text{ if } \mathscr{I}_w \models \phi \text{ and } [\![\tau(\phi)]\!]^\sigma_{\mathscr{R}}(\mathscr{I}_w) = e_\oplus \text{ if } \mathscr{I}_w \not\models \phi. \tag{A1}$$

We define $\tau(\phi)$ inductively as follows:

- if $\phi = p(\vec{x})$, then $\tau(\phi) = p(\vec{x})$
- if $\phi = \phi_1 \wedge \phi_2$, then $\tau(\phi) = \tau(\phi_1) * \tau(\phi_2)$
- if $\phi = \phi_1 \vee \phi_2$, then $\tau(\phi) = e_\otimes + -(e_\otimes + -\tau(\phi_1)) * (e_\otimes + -\tau(\phi_2))$
- if $\phi = \phi_1 \to \phi_2$, then $\tau(\phi) = \tau(\phi_1) \to \tau(\phi_2)$
- if $\phi = \forall y \phi_1(y)$, then $\tau(\phi) = \Pi y \tau(\phi_1(y))$
- if $\phi = \exists y \phi_1(y)$, then $\tau(\phi) = e_\otimes + -\Pi y (e_\otimes + -\tau(\phi_1(y))$

We can show by induction that $[\![\tau(\phi)]\!]_{\mathscr{R}}(\mathscr{I}_w) \in \{e_\oplus, e_\otimes\}$ and thus that the product Πy is always defined. Therefore, this encoding even includes the quantifiers. Furthermore, it is easy to see that the formula $\tau(\phi)$ is constructible in linear time from ϕ, and that hence the size of $\tau(\phi)$ is linear in the size ϕ. It is thus possible to eliminate unweighted formulas with polynomial overhead if addition $\oplus$ is invertible.

We note that while addition over the natural numbers $\mathbb{N}$ is not invertible, we can use the encoding $\tau(\phi)$ by moving from $\mathbb{N}$ to the integers $\mathbb{Z}$ and exploiting the fact that every natural number can be written as the sum of the squares of four integers (known as Lagrange's Four-Square Theorem); we then can add for global variables x over $\mathbb{N}$ the algebraic constraint $x \geq_\mathbb{Z} 0$ and use for local variables y in constraint formulas over $\mathbb{N}$ the expression $y_1^2 + y_2^2 + y_3^2 + y_4^2$ over four local variables $y_1, \ldots, y_4$ over $\mathbb{Z}$.

A similar translation is possible if the addition $\oplus$ is idempotent, i.e., $k \oplus k = k$ (as in the Boolean semiring), where the value of $\Sigma y \alpha(y)$ over $\mathscr{I}_w$ is naturally defined as k whenever a non-empty support $\mathrm{supp}_\oplus(\alpha(x), \mathscr{I}_w)$ leads to the single value k, i.e., for every $\xi \in \mathrm{supp}_\oplus(\alpha(x), \mathscr{I}_w)$ we have $[\![\alpha(\xi)]\!]^\sigma_{\mathscr{R}}(\mathscr{I}_w) = k$, since $\mathrm{supp}_\oplus(\tau(\phi), \mathscr{I}_w)$ leads to the single value $e_\otimes$.

3 ASP($\mathscr{AC}$)

Proposition 6 (Generalisation)**.** *Let ϕ be a σ-sentence and $\mathscr{I}_w$ be a pointed σ-HT-interpretation. Then, for the weighted σ-sentence α over the Boolean semiring $\mathbb{B}$, obtained from ϕ by replacing $\bot, \vee, \wedge, \to, \exists, \forall$ with $0, +, *, \to_\mathbb{B}, \Sigma, \Pi$, respectively, we have $[\![\alpha]\!]^\sigma_\mathbb{B}(\mathscr{I}_w) = 1$ iff $\mathscr{I}_w \models_\sigma \phi$.*

Proof. The claim can be easily verified by comparing the weighted semantics for $\mathscr{R} = \mathbb{B}$ and the unweighted semantics. We consider the case $\phi = \psi \to \theta$ in more detail. Then $\alpha = \beta \to_\mathbb{B} \theta$,

where β and γ correspond to the rewritten versions of ψ and θ.

$$\begin{aligned}
\mathscr{I}_w \models_\sigma \phi &\iff \mathscr{I}_{w'} \not\models_\sigma \psi \text{ or } \mathscr{I}_{w'} \models_\sigma \theta \text{ for } w' \geq w \\
&\iff [\![\beta]\!]^\sigma_{\mathbb{B}}(\mathscr{I}_w) \neq 1 \text{ or } [\![\gamma]\!]^\sigma_{\mathbb{B}}(\mathscr{I}_w) = 1 \text{ for } w' \geq w \\
&\iff [\![\beta]\!]^\sigma_{\mathbb{B}}(\mathscr{I}_w) = 0 \text{ or } [\![\gamma]\!]^\sigma_{\mathbb{B}}(\mathscr{I}_w) \neq 0 \text{ for } w' \geq w \\
&\iff [\![\alpha]\!]^\sigma_{\mathbb{B}}(\mathscr{I}_w) = 1
\end{aligned}$$

□

The proof for $\rightarrow_{\mathscr{R}}$ works over any semiring and not just the Boolean semiring. Therefore, we can drop the subscript $\mathscr{R}$.

Proposition 8 (Persistence)**.** *For any σ-sentence ϕ and σ-HT-interpretation $(\mathscr{I}^H, \mathscr{I}^T)$, it holds that $\mathscr{I}_H \models_\sigma \phi$ implies $\mathscr{I}_T \models_\sigma \phi$.*

Proof. It is know that the proposition holds for formulas ϕ without algebraic constraints (Pearce and Valverde 2006). We can use the same proof by structural induction, given that we can prove that the claim holds for the additional base case $\phi = k \sim_{\mathscr{R}} \alpha$.

In this case however, the definition of satisfaction tells us that

$$\mathscr{I}_w \models_\sigma k \sim_{\mathscr{R}} \alpha \iff k \sim [\![\alpha]\!]^\sigma_{\mathscr{R}}(\mathscr{I}_{w'}), \text{ for all } w' \geq w.$$

So, since $T \geq H$ from $\mathscr{I}_H \models_\sigma k \sim_{\mathscr{R}} \alpha$ follows $\mathscr{I}_T \models_\sigma k \sim_{\mathscr{R}} \alpha$. □

4 Constructs in ASP($\mathscr{AC}$) and in other formalisms

Conditionals We consider in more detail, how the conditional semantics *vc* and *df* of (Cabalar et al. 2020) can be modelled in our formalism. Since we do not capture arbitrary constraints as Cabalar et al. do, we assume instead that conditionals in weighted formulas are allowed and show that it is unnecessary to allow them explicitly. We start with *vc*.

Let $r(s) = H(r(s)) \leftarrow B(r(s))$ be some rule containing a conditional $s = (s'|s'' : \phi)$ which is supposed to be evaluated under *vc* semantics. This means that

$$\mathscr{I}_w \models r(s) \iff \begin{cases} \mathscr{I}_w \models r(s') & \text{if } \mathscr{I}_w \models \phi \\ \mathscr{I}_w \models r(s'') & \text{if } \mathscr{I}_w \models \neg\phi \\ \mathscr{I}_T \models r(s) & \text{otherwise.} \end{cases}.$$

Now if we simply replace s by $\phi * s' + \neg\phi * s''$ we get

$$\mathscr{I}_w \models r(\phi * s' + \neg\phi * s'') \iff \begin{cases} \mathscr{I}_w \models r(s') & \text{if } \mathscr{I}_w \models \phi \\ \mathscr{I}_w \models r(s'') & \text{if } \mathscr{I}_w \models \neg\phi \\ \mathscr{I}_T \models r(s) \text{ and } \mathscr{I}_w \models r(e_\oplus) & \text{otherwise.} \end{cases}.$$

This is obviously different, however if we use instead the rule

$$r'(s) = H(r(s)) \leftarrow B(r(s)), 1 =_{\mathbb{B}} \phi + \neg\phi$$

we obtain

$$
\begin{aligned}
&\mathscr{I}_w \models r'(\phi * s' + \neg\phi * s'') \\
\iff &\begin{cases} \mathscr{I}_w \models r'(s') & \text{if } \mathscr{I}_w \models \phi \\ \mathscr{I}_w \models r'(s'') & \text{if } \mathscr{I}_w \models \neg\phi \\ \mathscr{I}_T \models r'(s) \text{ and } \mathscr{I}_w \models H(r(e_\oplus)) \leftarrow B(r(e_\oplus)), 1 =_{\mathbb{B}} \phi + \neg\phi & \text{otherwise.} \end{cases} \\
\iff &\begin{cases} \mathscr{I}_w \models r(s') & \text{if } \mathscr{I}_w \models \phi \\ \mathscr{I}_w \models r(s'') & \text{if } \mathscr{I}_w \models \neg\phi \\ \mathscr{I}_T \models r(s) \text{ and } \mathscr{I}_w \models H(r(e_\oplus)) \leftarrow B(r(e_\oplus)), 1 =_{\mathbb{B}} 0 & \text{otherwise.} \end{cases} \\
\iff &\begin{cases} \mathscr{I}_w \models r(s') & \text{if } \mathscr{I}_w \models \phi \\ \mathscr{I}_w \models r(s'') & \text{if } \mathscr{I}_w \models \neg\phi \\ \mathscr{I}_T \models r(s) & \text{otherwise.} \end{cases}
\end{aligned}
$$

as desired.

In order to model *df*, we further need that the addition $\oplus$ of the semiring $\mathscr{R} = (R, \oplus, \otimes, e_\oplus, e_\otimes)$ is invertible, i.e. that we can use the connective $-$. Assume this is the case and let $s = (s' \mid s'' : \phi)$ be a conditional over the semiring $\mathscr{R}$ that we want to evaluate under *df*. Then its semantics is

$$
df_{\mathscr{I}_w}(s) = \begin{cases} s', & \text{if } \mathscr{I}_w \models_\sigma \phi, \\ s'', & \text{otherwise.} \end{cases}
$$

We can simply use the weighted formula $\phi * s' + (e_\otimes + -\phi) * s''$:

$$
\begin{aligned}
[\![\phi * s' + (e_\otimes + -\phi) * s'']\!]_{\mathscr{R}}(\mathscr{I}_w) &= [\![\phi * s']\!]_{\mathscr{R}}(\mathscr{I}_w) \oplus [\![(e_\otimes + -\phi) * s'']\!]_{\mathscr{R}}(\mathscr{I}_w) \\
&= \begin{cases} e_\otimes \otimes [\![s']\!]_{\mathscr{R}}(\mathscr{I}_w) \oplus (e_\otimes \oplus - e_\otimes) \otimes [\![s'']\!]_{\mathscr{R}}(\mathscr{I}_w) & \mathscr{I}_w \models \phi \\ e_\oplus \otimes [\![s']\!]_{\mathscr{R}}(\mathscr{I}_w) \oplus (e_\otimes \oplus - e_\oplus) \otimes [\![s'']\!]_{\mathscr{R}}(\mathscr{I}_w) & \text{otherwise} \end{cases} \\
&= \begin{cases} s' \oplus e_\oplus \otimes s'' & \mathscr{I}_w \models \phi \\ e_\otimes \otimes s'' & \text{otherwise} \end{cases} \\
&= \begin{cases} s' & \mathscr{I}_w \models \phi \\ s'' & \text{otherwise} \end{cases}
\end{aligned}
$$

Constraints in the heads of rules

Proposition 13 (Algebraic Choice Semantics)**.** *Let $r = k \sim^c_{\mathscr{R}} \alpha \leftarrow B(r)$ be a rule with global variables $x_1, \ldots, x_n$ and $(\mathscr{I}^H, \mathscr{I}^T)$ be a σ-HT-interpretation. Then $\mathscr{I}_H \models_\sigma k \sim^c_{\mathscr{R}} \alpha \leftarrow B(r)$ iff*

(i) $\mathscr{I}_H \models_\sigma k \sim_{\mathscr{R}} \alpha \leftarrow B(r)$ *and*

(ii) *for all $\xi_1 \in r(s(x_1)), \ldots, \xi_n \in r(s(x_n))$ it holds that if $\mathscr{I}_H \models_\sigma (B_\wedge(r)(\xi_1, \ldots, \xi_n)^\sigma$, then $[\![(\alpha(\xi_1, \ldots, \xi_n))^\Sigma]\!]^\sigma_{\mathscr{R}}(\mathscr{I}_T) = [\![(\alpha(\xi_1, \ldots, \xi_n))^\Sigma]\!]^\sigma_{\mathscr{R}}(\mathscr{I}_H)$*

Proof. According to our definition

$$
\begin{aligned}
&\mathscr{I}_H \models_\sigma k \sim^c_{\mathscr{R}} \alpha \leftarrow B(r) \\
\iff &\mathscr{I}_H \models_\sigma k \sim_{\mathscr{R}} \alpha \leftarrow B(r) \text{ and } \mathscr{I}_H \models_\sigma X =_{\mathscr{R}} \alpha \leftarrow X =_{\mathscr{R}} \alpha^{\neg\neg}, B(r) \\
\iff &\mathscr{I}_H \models_\sigma k \sim_{\mathscr{R}} \alpha \leftarrow B(r) \text{ and } \mathscr{I}_H \models_\sigma \forall x_1, \ldots, x_n \, \forall X \, X =_{\mathscr{R}} \alpha^{\neg\neg}, B(r) \rightarrow X =_{\mathscr{R}} \alpha
\end{aligned}
$$

By the definition of $\alpha^{\neg\neg}$ we know that $[\![\alpha^{\neg\neg}]\!]_{\mathscr{R}}(\mathscr{I}_H)$ is equal to $[\![\alpha]\!]_{\mathscr{R}}(\mathscr{I}_T)$. Therefore, we only need to consider the grounding of the rule where X is replaced by $[\![\alpha]\!]_{\mathscr{R}}(\mathscr{I}_T)$. Then

$$\begin{aligned} & \mathscr{I}_H \models r \\ \iff & \mathscr{I}_H \models_\sigma k \sim_{\mathscr{R}} \alpha \leftarrow B(r) \text{ and } \mathscr{I}_H \models_\sigma \forall x_1,\ldots,x_n\, B(r) \rightarrow [\![\alpha]\!]_{\mathscr{R}}(\mathscr{I}_T) =_{\mathscr{R}} \alpha \end{aligned}$$

□

5 Provenance

For provenance we define a translation of a positive datalog program $\Pi = \{r_1,\ldots,r_m\}$ as follows. First, we discuss terminology. For each predicate q in Π we introduce the following predicates.

- $p_q(\vec{Y},V,L,i,\vec{Z})$, which stores the value V of the provenance of $q(\vec{Y})$ using any derivation that uses exactly L leaf nodes, uses the rule r_i last and the global variables in r_i that do not occur in the head of r_i had the value $\vec{Z}$.
- $p_q(\vec{Y},V,L,i)$, which stores the value V of the provenance of $q(\vec{Y})$ using any derivation that uses exactly L leaf nodes, uses the rule r_i last and the global variables in r_i that do not occur in the head of r_i took any value.
- $p_q(\vec{Y},V,L)$, which stores the value V of the provenance of $q(\vec{Y})$ using any derivation that uses exactly L leaf nodes and uses any rule r_i last.
- $p_q(\vec{Y},V)$, which stores the value V of the provenance of $q(\vec{Y})$.
- $d_q(\vec{Y},L,i,\vec{Z})$, which asserts that there is a derivation of $q(\vec{Y})$ using exactly L leaf nodes that uses the rule r_i last and the global variables in r_i that do not occur in the head of r_i had the value $\vec{Z}$.
- $d_q(\vec{Y},L,i)$, which asserts that there is a derivation of $q(\vec{Y})$ using exactly L leaf nodes that uses the rule r_i last and the global variables in r_i that do not occur in the head of r_i took any value.
- $d_q(\vec{Y},L)$, which asserts that there is a derivation of $q(\vec{Y})$ using exactly L leaf nodes that uses any rule r_i.

Let

$$r_i = r(\vec{Y}) \leftarrow q_1(\vec{X}_1),\ldots,q_n(\vec{X}_n)$$

be some rule with index i, where w.l.o.g. $n > 1$ (we can always add a new extensional atom $e()$ with provenance $e_\otimes$. We add the following rules to our translation $T(\Pi)$:

$$p_r(\vec{Y},V,L,i,\vec{Z}) \leftarrow p_{q_1}(\vec{X}_1,V_1,L_1),\ldots,p_{q_n}(\vec{X}_n,V_n,L_n), L =_{\mathbb{N}} L_1 + \ldots + L_n, V =_{\mathscr{R}} V_1 * \ldots * V_n \tag{E1}$$

$$d_r(\vec{Y},L,i,\vec{Z}) \leftarrow p_{q_1}(\vec{X}_1,V_1,L_1),\ldots,p_{q_n}(\vec{X}_n,V_n,L_n), L =_{\mathbb{N}} L_1 + \ldots + L_n \tag{E2}$$

$$p_r(\vec{Y},V,L,i) \leftarrow d_r(\vec{Y},L,i,\vec{Z}), V =_{\mathscr{R}} p_r(\vec{Y},V^*,L,i,\vec{Z}^*) * V^* \tag{E3}$$

$$d_r(\vec{Y},L,i) \leftarrow d_r(\vec{Y},L,i,\vec{Z}) \tag{E4}$$

Here $\vec{Z} = \bigcup_{i=1}^n \vec{X}_i \setminus \vec{Y}$, i.e. the global variables of r_i that do not occur in its head.

Further, for every predicate q in Π we add the rules

$$p_r(\vec{Y},V,L) \leftarrow d_r(\vec{Y},L,I), V =_{\mathscr{R}} p_r(\vec{Y},V^*,L,I^*) * V^* \tag{E5}$$

$$d_r(\vec{Y},L) \leftarrow d_r(\vec{Y},L,I) \tag{E6}$$

$$p_r(\vec{Y},V) \leftarrow d_r(\vec{Y},L), V =_{\mathscr{R}} p_r(\vec{Y},V^*,L^*) * V^* \tag{E7}$$

This translation works as follows. The last rule sums up the value of each derivation tree to obtain the final provenance, whereas the previous rules calculate the provenance of less and less restricted derivation trees. Therefore, there is at least one answer set $\mathscr{I}$ such that $p_r(\vec{Y},V) \in \mathscr{I}$ iff the provenance of $r(\vec{Y})$ is V. On the other hand, since these rules are also all positive, there is exactly one answer set. The following results shows the correctness of the translation.

Theorem 14 (Provenance Encoding)**.** *Given a positive datalog program Π the $\mathscr{AC}$-program $T(\Pi)$ computes the provenance semantics over the ω-continuous semiring $\mathscr{R}$ in the following sense. Let D be an edb and $r(\vec{x})$ a query result of $D \cup \Pi$ with semiring provenance v. Then the unique equilibrium model $\mathscr{I}$ of $T(\Pi) \cup \{p_e(\vec{x},v,1) \leftarrow\mid (e(\vec{x}),v) \in D\}$ contains $p_r(\vec{x},v')$ iff $v' = v$.*

Proof. Let $\mathscr{I}$ be the unique equilibrium model of $T(\Pi) \cup \{p_e(\vec{x},v,1) \leftarrow\mid (e(\vec{x}),v) \in D\}$. We proceed by induction on the number L of leaf nodes that are used in the derivation tree, to show that our construction is correct and the predicates indeed behave as they should according to their description, which among other things implies that v is the provenance of the query result $r(\vec{x})$ iff the unique equilibrium model $\mathscr{I}$ of $T(\Pi) \cup \{p_e(\vec{x},v,1) \leftarrow\mid (e(\vec{x}),v) \in D\}$ contains $p_r(\vec{x},v)$. In the proof we only consider the predicates for the values of the provenance in detail. The correctness of the derivability predicates $d_r(\vec{Y},L[,i,\vec{Z}])$ follows analogous reasoning since the rules for derivability are just simplified versions of the rules for the provenance values.

The case $L = 0$ is impossible, since we always use at least one leaf node in each derivation.

The case $L = 1$ occurs exactly when $r(\vec{Y})$ is a leaf node. Since edb predicates do not occur in heads of rules in Π, the only rules we have to consider are of the form $p_e(\vec{x},v,1) \leftarrow$. Here, the claim holds.

Assume the claim holds for all $L' < L$.

Consider the rule (E1). For the body to be satisfied,we need that $p_{q_1}(\vec{X}_1,V_1,L_1),\ldots,$ $p_{q_n}(\vec{X}_n,V_n,L_n)$ are contained in $\mathscr{I}$. Since $L_i > 0$, $L = L_1 + \cdots + L_n$ and $n > 1$, we know that $L_i < L$ and therefore the claim holds for $p_{q_1}(\vec{X}_1,V_1,L_1),\ldots,p_{q_n}(\vec{X}_n,V_n,L_n)$. Therefore, $p_{q_i}(\vec{x}_i,v_i,l_i) \in \mathscr{I}$ iff v_i is the provenance of $q_i(\vec{x}_i)$ using any derivation that uses exactly l_i leaf nodes. Let us denote by $dtree_\ell(q_i(\vec{x}_i))$ the set of all derivation trees τ for $q_i(\vec{x}_i)$ that use exactly ℓ leaf nodes, and by $leaves(\tau)$ the set of leaf nodes in the tree τ. Then

$$v_i = \bigoplus_{\tau \in dtree_{l_i}(q_i(\vec{x}_i))} \bigotimes_{t \in leaves(\tau)} R(t). \tag{E8}$$

Then for $v = v_1 \otimes \cdots \otimes v_n$, and $l = l_1 + \cdots + l_n$ we have that $p_r(\vec{y},v,l,i,\vec{z})$ is in $\mathscr{I}$ iff for $i = 1,\ldots,n$ the atoms $p_{q_i}(\vec{x}_i,v_i,l_i)$ are in $\mathscr{I}$. This however means that for $i = 1,\ldots,n$ the equation (E8) holds. Therefore v is the value

$$v = v_1 \otimes \ldots \otimes v_n = \bigoplus_{\tau \in dtree_{l_1}(q_1(\vec{x}_1))} \bigotimes_{t \in leaves(\tau)} R(t) \otimes \cdots \otimes \bigoplus_{\tau \in dtree_{l_n}(q_n(\vec{x}_n))} \bigotimes_{t \in leaves(\tau)} R(t).$$

We use that for every combination of derivations $\tau_1,\ldots,\tau_n$ respectively for $q_1(\vec{x}_1),\ldots,q_n(\vec{x}_n)$ there is a derivation of $r(\vec{y})$ using the rule r_i last, where the global variables that do not occur in

the head of r_i have the value $\vec{z}$. According to the distributive law, assuming that $last(\tau)$ denotes the last rule in derivation tree τ and that $gvar(r_i)$ denotes the value of the global variables in rule r_i that do not occur in the head of r_i, we obtain that

$$v = v_1 \otimes \ldots \otimes v_n = \bigoplus_{\tau \in dtree_l(r(\vec{y})), r_i = last(\tau), gvar(r_i) = \vec{z}} \bigotimes_{t \in leaves(\tau)} R(t).$$

It follows that rule (E1) ensures that the predicates of the form $p_r(\vec{Y}, V, L, i, \vec{Z})$ satisfy our claim (for L).

Next for rule (E3). Here, we simply aggregate over the global variables in r_i that do not occur in the head of r_i.

The head atom $p_r(\vec{Y}, V, L, i)$ should describe the value V of the provenance of $r(\vec{Y})$ using any derivation that uses exactly L leaf nodes and uses rule r_i last. This value is given by

$$\bigoplus_{\tau \in dtree_L(r(\vec{x})), r_i = last(\tau)} \bigotimes_{t \in leaves(\tau)} R(t) = \bigoplus_{\vec{z}} \bigoplus_{\tau \in dtree_l(r(\vec{x})), r_i = last(\tau), gvar(r_i) = \vec{z}} \bigotimes_{t \in leaves(\tau)} R(t).$$

We know that according to the previous rule (E1), the predicate $p_r(\vec{Y}, V, L, i, \vec{Z})$ encodes exactly the inner sum. Since rule (E3) performs the outer sum, it follows that rule (E3) ensures that the predicates of the form $p_r(\vec{Y}, V, L, i)$ satisfy our claim (for L).

Next, the rule (E5) aggregates over the different rules that were used last to derive $r(\vec{x})$ using l leaf nodes. The argumentation is analogous to the one for the last rule (E3), where we aggregated over $\vec{z}$ instead of the rule index i like here.

Overall, the inductive proof of the correctness of the rules specifying the predicates $p_r(\vec{Y}, V, L[, i, \vec{Z}])$ succeeds, since all the predicates are correctly defined for L given that predicates are well defined for $L' < L$.

Last but not least we consider the last rule (E7) which should produce the final result. According to the definition of provenance for datalog programs in (Green et al. 2007), the label v of the query result $r(\vec{x})$ is

$$v = \bigoplus_{\tau \in dtree(r(\vec{x}))} \bigotimes_{t \in leaves(\tau)} R(t),$$

where $dtree(r(\vec{x}))$ is the set of all derivation trees for $r(\vec{x})$ and $R(t)$ is the provenance of the leaf t. We reformulate this equation as follows:

$$v = \bigoplus_{\tau \in dtree(r(\vec{x}))} \bigotimes_{t \in leaves(\tau)} R(t) = \bigoplus_{l \geq 0} \bigoplus_{\tau \in dtree_l(r(\vec{x}))} \bigotimes_{t \in leaves(\tau)} R(t);$$

since $p_r(\vec{x}, v', l) \in \mathscr{I}$ iff $v' = \bigoplus_{\tau \in dtree_l(r(\vec{x}))} \bigotimes_{t \in leaves(\tau)} R(t)$, we obtain

$$v = \bigoplus_{l \geq 0} \bigoplus_{p_r(\vec{x}, v', l) \in \mathscr{I}} v'.$$

As due to rule (E7), we have $p_r(\vec{x}, v) \in \mathscr{I}$ iff

$$v = \bigoplus_{l \geq 0} \bigoplus_{p_r(\vec{x}, v', l) \in \mathscr{I}} v',$$

we obtain that $p_r(\vec{x}, v) \in \mathscr{I}$. This concludes the proof. □

6 Language Aspects

6.1 Safety

Theorem I (Support Independence)**.** *Let σ_1, σ_2 be semiring signatures, $\mathscr{I}_w$ be a pointed σ_i-HT-interpretation ($i = 1,2$) and α be a weighted σ_i-formula ($i = 1,2$) that is syntactically domain independent w.r.t. variable x. Then*

$$\{\xi \in r_1(s(x)) \mid [\![\alpha(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} = \{\xi \in r_2(s(x)) \mid [\![\alpha(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}.$$

Proof. We give a proof using structural induction on the syntactically domain independent formula α. In the following let $\mathscr{I}_w$ some pointed HT-interpretation and σ_1, σ_2 semiring signatures that contain all the constants of α and $\mathscr{I}_w$

- Case $\alpha = k$:
  This formula contains no local variables, therefore the equality is trivially fulfilled.
- Case $\alpha = \phi(\vec{x})$:
  The given formulas are all range restricted. For range restricted formulas it is known, that they are domain independent (see for example (Demolombe 1992)), which implies that when they are seen as weighted formulas, their support does not depend on the signature.
- Case $\alpha = \neg_\oplus \beta(x)$:
  The semantics of $\neg_\oplus \beta$ is the inverse of the semantics of β w.r.t. $\oplus$, which is $e_\oplus$ iff the semantics of β is $e_\oplus$. Therefore we have

$$\begin{aligned}
&\{\xi \in r_1(s(x)) \mid [\![\neg_\oplus \beta(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\
=&\{\xi \in r_1(s(x)) \mid [\![\beta(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\
=&\{\xi \in r_2(s(x)) \mid [\![\beta(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\
=&\{\xi \in r_2(s(x)) \mid [\![\neg_\oplus \beta(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}.
\end{aligned}$$

- Case $\alpha = \neg_\otimes \beta(x)$:
  The semantics of $\neg_\otimes \beta$ is the inverse of the semantics of β w.r.t. $\otimes$ or $e_\oplus$ if the semantics of β is $e_\oplus$. Therefore we have on the one hand that

$$[\![\beta(\xi)]\!]^{\sigma_i}_{\mathscr{R}}(\mathscr{I}) = e_\oplus \Rightarrow [\![\neg_\otimes \beta(\xi)]\!]^{\sigma_i}_{\mathscr{R}}(\mathscr{I}) = e_\oplus.$$

  Furthermore, we have for the other direction that

$$[\![\neg_\otimes \beta(\xi)]\!]^{\sigma_i}_{\mathscr{R}}(\mathscr{I}) = e_\oplus \Rightarrow [\![\beta(\xi)]\!]^{\sigma_i}_{\mathscr{R}}(\mathscr{I}) = e_\oplus \vee [\![\beta(\xi)]\!]^{\sigma_i}_{\mathscr{R}}(\mathscr{I}) \otimes e_\oplus = e_\otimes$$

  The second disjunct implies that $e_\oplus = e_\otimes$ since $e_\oplus$ annihilates R. Therefore since $\forall r \in R : e_\otimes \otimes r = r$ holds we have that $\forall r \in R : e_\oplus \otimes r = e_\oplus = r$, meaning our semiring has exactly one element, namely $e_\oplus$. Therefore we have

$$[\![\neg_\otimes \beta(\xi)]\!]^{\sigma_i}_{\mathscr{R}}(\mathscr{I}) = e_\oplus \Rightarrow [\![\beta(\xi)]\!]^{\sigma_i}_{\mathscr{R}}(\mathscr{I}) = e_\oplus$$

  So we know that the semantics of $\neg_\otimes \beta$ is $e_\oplus$ iff the semantics of β is $e_\oplus$ and as in the previous case we obtain

$$\begin{aligned}
&\{\xi \in r_1(s(x)) \mid [\![\neg_\otimes \beta(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\
=&\{\xi \in r_1(s(x)) \mid [\![\beta(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\
=&\{\xi \in r_2(s(x)) \mid [\![\beta(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\
=&\{\xi \in r_2(s(x)) \mid [\![\neg_\otimes \beta(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}.
\end{aligned}$$

- Case $\alpha = \neg\neg\alpha_1(x)$:
  We know that $\{\xi \in r_1(s(x)) \mid [\![\neg\neg\alpha_1(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}$ is equal to $\{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}$. Therefore this case follows immediately from the inductive hypothesis for $\alpha_1(x)$.
- Case $\alpha = \alpha_1(x) + \alpha_2(x)$:
  Assume that there is

  $$\xi \in r_2(s(x)) \setminus r_1(s(x)) \text{ s.t. } [\![\alpha_1(\xi) + \alpha_2(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\},$$

  then we know that

  $$[\![\alpha_1(\xi)]\!]_{\mathscr{R}}(\mathscr{I}) \neq e_\oplus \text{ or } [\![\alpha_2(\xi)]\!]_{\mathscr{R}}(\mathscr{I}) \neq e_\oplus$$

  and further that $\xi \notin \mathscr{D}_2$, since σ_1, σ_2 are semiring signatures. Then it however holds that

  $$\xi \in \{\xi \in r_2(s(x)) \mid [\![\alpha_1(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \setminus \{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}$$

  or

  $$\xi \in \{\xi \in r_2(s(x)) \mid [\![\alpha_2(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \setminus \{\xi \in r_1(s(x)) \mid [\![\alpha_2(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}.$$

  This is impossible due to the induction hypothesis for $\alpha_1(x)$ and $\alpha_2(x)$. Analogously we can show that

  $$\{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi) + \alpha_2(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \setminus \{\xi \in r_2(s(x)) \mid [\![\alpha_1(\xi) + \alpha_2(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}$$

  is empty, and therefore that

  $$\begin{aligned} &\{\xi \in r_2(s(x)) \mid [\![\alpha_1(\xi) + \alpha_2(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\ =&\{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi) + \alpha_2(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}. \end{aligned}$$

- Case $\alpha = \alpha_1(x) * \alpha_2$:
  First note that:

  $$\{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi) * \alpha_2]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \subseteq \{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\}$$

  We use the induction hypothesis on $\alpha_1(x)$ to obtain the equality:

  $$\begin{aligned} &\{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi) * \alpha_2]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\ =&\{\xi \in \{\xi \in r_1(s(x)) \mid [\![\alpha_1(\xi)]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \mid [\![\alpha_1(\xi) * \alpha_2]\!]^{\sigma_1}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\ =&\{\xi \in \{\xi \in r_2(s(x)) \mid [\![\alpha_1(\xi)]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \mid [\![\alpha_1(\xi) * \alpha_2]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \\ =&\{\xi \in r_2(s(x)) \mid [\![\alpha_1(\xi) * \alpha_2]\!]^{\sigma_2}_{\mathscr{R}}(\mathscr{I}_w) \neq e_\oplus\} \end{aligned}$$

- Case $\alpha = \alpha_1 * \alpha_2(x)$:
  works analogously to the one above.
- Case $\alpha = \alpha_1(x) * \phi(\vec{X}')$, where $\vec{X}' \subseteq \{x\}$:
  works analogously to the one above.

The proof for more than one local variable works analogously. □

Theorem 16 is a corollary of Theorem I:

Theorem 16 (Domain Independence)**.** *If a formula is syntactically domain independent, then it is also domain independent.*

Proof. When we evaluate $\alpha(\vec{X})$ we take the sum over $\mathrm{supp}_{\oplus}(\alpha(\vec{X}), \mathscr{I}_w)$. Due to the previous lemma we know that the support is invariant under changing the domain. Further, we know that for a given assignment of the local variables the semantics is independent of the domain. Therefore, the semantics is invariant under changing the domain for syntactically domain independent formulas. □

Theorem 18 (Program Domain Independence)**.** *If a program* Π *is safe, then it is domain independent.*

Proof (sketch). Let $\sigma_i = \langle \mathscr{D}_i, \mathscr{P}, \mathscr{X}, \mathscr{S}, r_i \rangle, i = 1,2$ be semiring signatures s.t. Π is a σ_i-formula for $i = 1,2$ and let $\mathscr{I}_w = (\mathscr{I}^H, \mathscr{I}^H, w)$ be a pointed σ_i-HT-interpretation for $i = 1,2$.

Let $r \in \Pi$. If r does not contain global variables, the claim is evident. Otherwise assume $r = \forall x_1, \ldots, x_n \; \alpha(x_1, \ldots, x_n)$. For $\xi_i \in r_1(s(x_i)) \cap r_2(s(x_i))$ the semantics of $\alpha(\xi_1, \ldots, \xi_n)$ does not depend on σ_i. Otherwise, assume for some j it holds that $\xi_j \in r_1(s(x_j)) \setminus r_2(s(x_j))$. Then $\mathscr{I}_w \models_{\sigma_i} \alpha(\xi_1, \ldots, \xi_n)$ for $i = 1,2$ since x_j satisfies condition (ii.1) and therefore there exists an atom in the body that is not satisfied by $\mathscr{I}_w$. □

6.2 Program Equivalence

Theorem 20. *For any* Π_1, Π_2 *programs, we have that* $\Pi_1 \equiv_s \Pi_2$ *iff* Π_1 *has the same HT-models as* Π_2.

Proof (sketch). The direction $\Leftarrow$ is clear. For $\Rightarrow$ we can generalise the proof in (Lifschitz et al. 2001), by constructing Π', which asserts a subset of the interpretation $\mathscr{I}^T$ that is ensured to be stable ($\mathscr{I}^H$), and a subset that if partly present is ensured to be fully present ($\mathscr{I}^T \setminus \mathscr{I}^H$).

Let Π_1 and Π_2 have different HT-models. W.l.o.g. there must be at least one HT-interpretation $(\mathscr{I}^H, \mathscr{I}^T)$ that is an HT-model of Π_1 but not of Π_2. As in (Lifschitz et al. 2001) we simply define

$$\Pi' = \{p(\vec{x}) \leftarrow \mid p(\vec{x}) \in \mathscr{I}^H\} \cup \{p(\vec{x}) \leftarrow q(\vec{y}) \mid p(\vec{x}), q(\vec{y}) \in \mathscr{I}^T \setminus \mathscr{I}^H\}$$

Then $\mathscr{I}^T$ is an equilibrium model of $\Pi_2 \cup \Pi'$, but not of $\Pi_1 \cup \Pi'$ and therefore Π_1 and Π_2 are not strongly equivalent. □

The above proof relies on the fact that our semantics is defined for program with infinite sets of rules. If we want to avoid this, there are multiple options. In (Lifschitz et al. 2007) the strong equivalence of arbitrary first-order formulas was considered and characterised as equivalence in HT Logic. The proof however uses the fact that the strong equivalence considered in their work is for any first-order sentence and not only for programs, which are a syntactic fragment. A straight forward way to reproduce their proof strategy in our setting seems not to be apparent.

Nevertheless, it is possible to prove the statement when programs are finite sets of rules in our setting, provided that auxiliary predicate symbols are available not occurring in the program (which trivially holds if we have infinitely many predicates of each arity in the underlying predicate signature $\mathscr{P}$).

Proof (sketch). We have two directions to prove. Based on the idea of (Lifschitz et al. 2001).
($\Rightarrow$) We prove this direction using contraposition, that is we assume that we have two programs Π_1, Π_2 s.t. for some HT-interpretation $(\mathscr{I}^H, \mathscr{I}^T)$ it holds that (w.l.o.g.) $(\mathscr{I}^H, \mathscr{I}^T, H) \models \Pi_1$ and $(\mathscr{I}^H, \mathscr{I}^T, H) \not\models \Pi_2$. Next we show that there exists a program Δ s.t. $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ have different answer sets.

The program Δ consists of the following rules, where $\mathscr{G}$ is the set of predicates that occur in $\Pi_1 \cup \Pi_2$:

$$\texttt{repair}_p(X_1,\ldots,X_n) \leftarrow \top =_{\mathbb{B}} \neg\neg\texttt{repair}_p(X_1,\ldots,X_n), \quad \text{for } p \in \mathscr{G} \text{ with arity } n. \tag{F1}$$

$$p(X_1,\ldots,X_n) \leftarrow \texttt{repair}_p(X_1,\ldots,X_n), \quad \text{for } p \in \mathscr{G} \text{ with arity } n. \tag{F2}$$

$$\texttt{fill}_{p,q}(X_1,\ldots,X_n,Y_1,\ldots,Y_m) \leftarrow \top =_{\mathbb{B}} \neg\neg\texttt{fill}_{p,q}(X_1,\ldots,X_n,Y_1,\ldots,Y_m), \quad \text{for } \texttt{p}, \texttt{q} \in \mathscr{G} \text{ with arities } n,m. \tag{F3}$$

$$p(X_1,\ldots,X_n) \leftarrow q(Y_1,\ldots,Y_m), \texttt{fill}_{p,q}(X_1,\ldots,X_n,Y_1,\ldots,Y_m), \quad \text{for } \texttt{p}, \texttt{q} \in \mathscr{G} \text{ with arities } n,m. \tag{F4}$$

Intuitively $\texttt{repair}_p(X_1,\ldots,X_n)$ guesses some tuple $(X_1,\ldots,X_n)$ for predicate p such that the atom $p(X_1,\ldots,X_n)$ should definitely be satisfied. Similarly, $\texttt{fill}_{p,q}(X_1,\ldots,X_n,Y_1,\ldots,Y_m)$ guesses some values $(X_1,\ldots,X_n)$ and $(Y_1,\ldots,Y_m)$ for the predicates $\texttt{p}$ and $\texttt{q}$, respectively, such that if $p(X_1,\ldots,X_n)$ is satisfied then also $q(Y_1,\ldots,Y_n)$ should be satisfied.

Consider now the interpretation

$$\mathscr{I}^* = \mathscr{I}^T \cup \{\texttt{repair}_p(x_1,\ldots,x_n) \mid p(x_1,\ldots,x_n) \in \mathscr{I}^H\} \tag{F5}$$

$$\cup \{\texttt{fill}_{p,q}(x_1,\ldots,x_n,y_1,\ldots,y_m) \mid p(x_1,\ldots,x_n), q(y_1,\ldots,y_n) \in \mathscr{I}^T \setminus \mathscr{I}^H\}. \tag{F6}$$

Then we have that $(\mathscr{I}^H,\mathscr{I}^*,H) \models \Pi_1 \cup \Delta$, therefore $\mathscr{I}^*$ is not an equilibrium model of $\Pi_1 \cup \Delta$. However for Π_2 we have that $(\mathscr{I}^*,\mathscr{I}^*,H) \models \Pi_2 \cup \Delta$. Furthermore, consider now some interpretation $\mathscr{I}' \subseteq \mathscr{I}^*$ s.t. $(\mathscr{I}',\mathscr{I}^*,H) \models \Pi_2 \cup \Delta$. Due to the included repairs, we know that at least $\mathscr{I}^H \subseteq \mathscr{I}'$. Moreover, we know that this inclusion is strict even when we consider only the predicates occurring in $\Pi_1 \cup \Pi_2$, since $(\mathscr{I}^H,\mathscr{I}^T,H) \not\models \Pi_2$. Therefore, due to the fills we have to include all the predicates from $\mathscr{I}^T$. It follows that $\mathscr{I}' = \mathscr{I}^*$ and therefore that $\mathscr{I}^*$ is an equilibrium model of $\Pi_2 \cup \Delta$.

($\Leftarrow$) Assume that Π_1 has the same HT-models as Π_2 and consider for an arbitrary program Δ the HT-models of $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$. Those are exactly the HT-models $(\mathscr{I}^H,\mathscr{I}^T)$ s.t.

$$(\mathscr{I}^H,\mathscr{I}^T,H) \models \Pi_1 \text{ and } (\mathscr{I}^H,\mathscr{I}^T,H) \models \Delta,$$

which is however equivalent to

$$(\mathscr{I}^H,\mathscr{I}^T,H) \models \Pi_2 \text{ and } (\mathscr{I}^H,\mathscr{I}^T,H) \models \Delta$$

since Π_1 and Π_2 have the same HT-models. Since the HT-models of $\Pi_1 \cup \Delta$ and $\Pi_2 \cup \Delta$ are the same, we also know that the equilibrium models $\mathscr{I}$ are the same, since it follows that

$$(\mathscr{I},\mathscr{I},H) \models \Pi_1 \cup \Delta \text{ and } \forall \mathscr{I}' \subsetneq \mathscr{I} : (\mathscr{I}',\mathscr{I},H) \not\models \Pi_1 \cup \Delta$$
$$\iff (\mathscr{I},\mathscr{I},H) \models \Pi_2 \cup \Delta \text{ and } \forall \mathscr{I}' \subsetneq \mathscr{I} : (\mathscr{I}',\mathscr{I},H) \not\models \Pi_2 \cup \Delta.$$

□

7 Complexity

Theorem II (Complexity of evaluation)**.** *Let $\mathscr{R} = (R,\oplus,\otimes,e_\oplus,e_\otimes)$ some semiring and $e : R \to \mathbb{N}$ some encoding function s.t. $\mathscr{R}$ is efficiently encoded by e.*

Then for a quantifier-free weighted formula over $\mathscr{R}$ and pointed HT-interpretation $\mathscr{I}_w$, we can calculate $e([\![\alpha]\!]_{\mathscr{R}}(\mathscr{I}_w))$ in polynomial time.

Proof. The proof is by structural induction on the formula α, with induction invariant that $t(\alpha)$ the time needed is in $\mathscr{O}(N^n)$, where N is the size of the input, $n \in \mathbb{N}$ is a constant not depending on the input. Further, $s(\alpha)$ the size of the representation of the obtained value, i.e. $\|[\![\alpha]\!]_{\mathscr{R}}(\mathscr{I}_w)\|$, is in $\mathscr{O}(N)$.‘

- Base Cases:
  - $\alpha = e(k)$: Then one can evaluate the expression by simply returning $e(k)$. This is feasible in polynomial time. The size of the output is linear in the size of the input.
  - $\alpha = \phi$: We simply check if $\mathscr{I}_w \models \phi$ and return $e_\oplus$ or $e_\otimes$ accordingly. This is possible in polynomial time since ϕ is quantifier-free and the size of the output is also bounded by a constant.

  We have shown the invariant for all formulae up to a certain structural complexity.
- Induction Step:
  - $\alpha = \beta_1 \to \beta_2$: We know that for β_i the invariant holds, therefore we can check in time bounded polynomially in the size of the formula, whether $[\![\beta_i]\!]_{\mathscr{R}}(\mathscr{I}_w) = e_\oplus$ and output $e_\oplus$ or $e_\otimes$ accordingly. The size of the output is again bounded by a constant.
  - $\alpha = \beta_1 + \beta_2$: We know that the invariant holds for β_1, β_2. Further $t(\alpha) = t(\beta_1) + t(\beta_2) + x$, where x is the time needed for addition of the results for β_1 and β_2. We know that x is polynomial in $s(\beta_1) + s(\beta_2)$, which we know to be in $\mathscr{O}(N)$. Therefore $x \in \mathscr{O}(N^l)$, where l is the degree of the polynomial bounding the time needed to add two numbers. It follows that $t(\alpha) \in \mathscr{O}(N^n)$. For $s(\alpha)$ we can see that

    $$\begin{aligned} s(\alpha) &= \|[\![\alpha]\!]_{\mathscr{R}}(\mathscr{I}_w)\| \\ &\leq \|[\![\beta_1]\!]_{\mathscr{R}}(\mathscr{I}_w)\| + \|[\![\beta_2]\!]_{\mathscr{R}}(\mathscr{I}_w)\| + C \end{aligned}$$

    And therefore $s(\alpha) \in \mathscr{O}(N)$.
  - $\alpha = \beta_1 * \beta_2$: The proof works analogously to the proof for the case $\alpha = \beta_1 + \beta_2$.
  - $\alpha = -\beta$: We know that the invariant holds for β. Further $t(\alpha) = t(\beta) + x$, where x is the time needed for inversion of the result for β. We know that x is polynomial in $s(\beta)$, which we know to be in $\mathscr{O}(N)$. Therefore $x \in \mathscr{O}(N^l)$, where l is the degree of the polynomial bounding the time needed to invert a number. It follows that $t(\alpha) \in \mathscr{O}(N^n)$. For $s(\alpha)$ we can see that

    $$\begin{aligned} s(\alpha) &= \|[\![\alpha]\!]_{\mathscr{R}}(\mathscr{I}_w)\| \\ &\leq \|[\![\beta]\!]_{\mathscr{R}}(\mathscr{I}_w)\| + C \end{aligned}$$

    And therefore $s(\alpha) \in \mathscr{O}(N)$.
  - $\alpha = \beta^{-1}$: The proof works analogously to the proof for the case $\alpha = -\beta$.

□

Theorem 22 (Ground Complexity)**.** *Let Π be a variable-free program s.t. each semiring in Π is efficiently encoded. Then*

- *MC is co-NP-complete.*
- *(propositional) SE is co-NP-complete.*

- *SAT is σ_2^p-complete.*

Proof (sketch). The hardness parts are inherited from the complexity of the respective problems for disjunctive logic programs (Dantsin et al. 2001; Lin 2002): The disjunctive logic programming rule

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \ldots, b_m, \neg c_1, \ldots, \neg c_k$$

is strongly equivalent to the $\mathscr{AC}$-rule

$$1 =_{\mathbb{B}} a_1 + \ldots + a_n \leftarrow b_1, \ldots, b_m, \neg c_1, \ldots, \neg c_k.$$

The memberships follow from the possibility of applying guess and check algorithms. We only need that given $(\mathscr{I}^H, \mathscr{I}^T)$ and algebraic constraint $k \sim_{\mathscr{R}} \alpha$, we can decide in polynomial time whether $\mathscr{I}_H \models k \sim_{\mathscr{R}} \alpha$. This is possible since we know that $\mathscr{R}$ is efficiently encoded: We only need to perform polynomially many additions, multiplications and inversions which each take polynomial time as Theorem II says. □

Theorem 23 (Non-gound Complexity)**.** *Let* Π *be a safe program such that each semiring in* Π *is efficiently encoded. Then*

(i) *MC is in EXPTIME, and co-NP$^{NP^{PP}}$-hard (thus also co-NP^{PP}-hard and NP^{PP}-hard).*
(ii) *SAT is undecidable.*
(iii) *SE is undecidable.*

Proof (sketch). (i) Given the interpretation $\mathscr{I}$ (as set of ground atoms), we can iterate over all $\mathscr{I}' \subsetneq \mathscr{I}$ and check $(\mathscr{I}, \mathscr{I}, H) \models r'$, as well as $(\mathscr{I}', \mathscr{I}, H) \models r'$ for each ground instance r' of a rule $r \in \Pi$ in exponential time. The iteration and considering one ground instance r' at a time is feasible in polynomial space; the evaluation of algebraic constraints $k \sim_{\mathscr{R}} \alpha$ is feasible in exponential time, since if α is of the form $\Sigma y_1, \ldots, y_n \alpha'(y_1, \ldots, y_n)$ where α' is quantifier-free, by safety of the program each y_i must occur in some atom $p(\vec{x})$. That is, to evaluate α, we only need to consider values $\xi(y_i)$ for y_i, $i = 1, \ldots, n$ that occur in the interpretation $\mathscr{I}$. There are exponentially many such ξ; for each of them, the value of $\alpha'(\xi(y_1), \ldots, \xi(y_n))$ can be computed in polynomial time given that $\mathscr{R}$ is efficiently encoded, yielding a value r_ξ such that $e(r_\xi)$ occupies polynomially many bits. The aggregation $\Sigma_\xi r_\xi$ over all ξ is then feasible in exponential time by the assertion that $\|r_1 \oplus r_2\| \leq \|r_1\| + \|r_2\| + c$ and that $e(r_1 \oplus r_2)$ is computable in polynomial time given $e(r_1), e(r_2)$.

We note that the value of $\Sigma_\xi r_\xi$ may under the assertions occupy exponentially many bits; under stronger assumptions on the encoding $e(r)$, a smaller upper bound may be derived. E.g., we obtain membership in PSPACE if it is ensured that for the addition, we have the stronger condition $\|r_1 \oplus r_2 \oplus \cdots \oplus r_n\| \leq (1 + \log n) \max_i \|r_i\| + c$, for every $n \geq 2$ where c is a constant. For example, the canonical semiring $\mathbb{N}$ of the natural numbers satisfies this property.

The co-NP$^{\text{NP}^{\text{PP}}}$-hardness is due to a reduction from AE-MAJSAT, which asks whether for a Boolean formula $\phi(x_1, \ldots, x_n)$ for all assignments to $x_1, \ldots, x_m$ a partial assignment to $x_{m+1}, \ldots, x_k$

exists s.t. more than 2^{n-k-1} of the assignments to $x_{k+1},\ldots,x_n$ satisfy $\phi(\vec{x})$. Then the program

$$\begin{aligned}
e(0) &\leftarrow \\
e(1) &\leftarrow \\
1 =_{\mathbb{B}} a_1(0) + a_1(1) &\leftarrow \\
&\ldots \\
1 =_{\mathbb{B}} a_m(0) + a_m(1) &\leftarrow \\
1 =_{\mathbb{B}} a_1(0) * a_1(1) &\leftarrow a_1(X_1),\ldots,a_m(X_m),e(X_{m+1}),\ldots,e(X_k), \\
&\qquad 2^{n-k-1} <_{\mathbb{N}} e(X_{k+1}) * \ldots * e(X_n) * \phi(\vec{X}) \\
&\ldots \\
1 =_{\mathbb{B}} a_m(0) * a_m(1) &\leftarrow a_1(X_1),\ldots,a_m(X_m),e(X_{m+1}),\ldots,e(X_k), \\
&\qquad 2^{n-k-1} <_{\mathbb{N}} e(X_{k+1}) * \ldots * e(X_n) * \phi(\vec{X})
\end{aligned}$$

has an equilibrium model $\mathscr{I} = \{a_1(0),a_1(1),\ldots,a_m(0),a_m(1),e(0),e(1)\}$ iff the answer for AE-MAJSAT is yes.

Assume the answer for AE-MAJSAT is yes. Then for every subset $\mathscr{I}' \subseteq \mathscr{I}$ we have $(\mathscr{I}',\mathscr{I},H) \not\models \Pi$. If we remove $e(i)$ or both $a_i(0)$ and $a_i(1)$ for some i, this is clear. Otherwise, we know that for each i some $a_i(j_i)$ holds. Then for these values there exist values $j_{m+1},\ldots,j_k$ s.t. $\phi(j_1,\ldots,j_k,X_{k+1},\ldots,X_n)$ is a yes instance for MAJSAT. Therefore the body

$$a_1(j_1),\ldots,a_m(j_m),e(X_{j+1}),\ldots,e(j_k),2^{n-k-1} <_{\mathbb{N}} e(X_{k+1}) * \ldots * e(X_n) * \phi(\vec{X})$$

is satisfied and if $(\mathscr{I}',\mathscr{I},H) \models \Pi$ we know that $\mathscr{I}' = \mathscr{I}$.

On the other hand if the answer for AE-MAJSAT is no, due to the partial assignment $j_1,\ldots,j_m$ to the variables $x_1,\ldots,x_m$, for $\mathscr{I}' = \{a_1(j_1),\ldots,a_m(j_m),e(0),e(1)\}$ we have $(\mathscr{I}',\mathscr{I},H) \models \Pi$, and therefore $\mathscr{I}$ is not an equilibrium model.

(ii) The undecidable Mortal Matrix Problem asks whether any product of matrices in $X = \{X_1,\ldots,X_n\} \subset \mathbb{Z}^{d\times d}$ evaluates to the zero matrix 0_d (Cassaigne et al. 2014). $(\mathbb{Z}^{d\times d},+,\cdot,0_d,1_d)$ is efficiently encodable. The program

$$\begin{aligned}
p(X_i) &\leftarrow \qquad (i = 1,\ldots,n) \\
\bot &\leftarrow \neg p(0_d) \\
p(Y) &\leftarrow p(Z_1),p(Z_2),Y =_{\mathbb{Z}^{d\times d}} Z_1 * Z_2
\end{aligned}$$

has a stable model iff the answer to the mortal matrix problem for X is yes, since $p(0_d)$ needs to be supported.

(iii) Let Π be the program from above. Then the answer to the mortal matrix problem for X is yes iff Π is strongly equivalent to $\Pi' = \Pi \setminus \{\bot \leftarrow \neg p(0)_d)\}$. This can be seen as follows.

As Π' has no negation, its HT-models are the interpretations $(\mathscr{I}',\mathscr{I})$ where both $\mathscr{I}'$ and $\mathscr{I}$ are closed under the rules of Π', sets S such that $p(X_1),\ldots,p(X_n) \in S$ and whenever $p(Y),p(Z) \in S$ then also $p(Y*Z) \in S$. Similarly, the HT-models of Π are the interpretations $(\mathscr{I}',\mathscr{I})$ where $\mathscr{I}'$ and $\mathscr{I}$ are closed under the rules of Π' and in addition $p(0_d) \in \mathscr{I}'$.

Therefore, $\Pi \equiv_s$ and Π' iff $p(0_d) \in L$, where L is the least set closed under the rules of Π', which holds iff the answer for the mortal matrix problem on X is yes. □

Corollary III. *When the program Π is over the semiring $\mathbb{N}$ of the natural numbers, we have co-$NP^{NP^{PP}}$-completeness for MC.*

Proof. The hardness is due to the proof of the previous theorem. The membership follows from the fact that we can check the satisfaction of constraints over $\mathbb{N}$ using a PP oracle.

This can be seen as follows. We can evaluate weighted formulas over $\mathbb{N}$ of the form $\Sigma y_1 \dots \Sigma y_n \alpha$ where α is quantifier-free using a #P oracle: we can non-deterministically choose an assignment ξ to $y_1, \dots, y_n$, calculate $r = [\![\alpha(\xi)]\!]_{\mathbb{N}}(\mathscr{I}_w)$ in polynomial time and generate r accepting branches.

Since P^{PP} is equal to $\mathrm{P}^{\#\mathrm{P}}$ also co-$\mathrm{NP}^{\mathrm{NP}^{\mathrm{PP}}}$ is equal to co-$\mathrm{NP}^{\mathrm{NP}^{\#\mathrm{P}}}$.

As for the co-$\mathrm{NP}^{\mathrm{NP}^{\#\mathrm{P}}}$ membership: Given a program Π and a potential equilibrium model $\mathscr{I}$ we can guess a subset $\mathscr{I}' \subsetneq \mathscr{I}$ and check whether $(\mathscr{I}', \mathscr{I}, H) \models \Pi$. The latter can be achieved in co-$\mathrm{NP}^{\#\mathrm{P}}$ by guessing a rule $r \in \Pi$ and an assignment ξ to its global variables. Then we can check whether $(\mathscr{I}', \mathscr{I}, H) \models r(\xi)$ in $\mathrm{P}^{\#\mathrm{P}}$ by checking satisfaction of each atom and constraint in $r(\xi)$. □

We note that MC is decidable for $\mathscr{AC}$-programs over the natural numbers while SAT and SE are undecidable. This may not be much surprising from Theorem **??**, given that the semiring $\mathbb{Z}^{d\times d}$ is (efficiently) encodable to $\mathbb{N}$. The undecidability can directly be shown by a reduction from solving Diophantine equations, i.e., polynomial equations $P(x_1, ..., x_n) = 0$ in variables $x_1, \dots, x_n$ over the integers, which by Matiyasevich's celebrated result is undecidable; this holds if the solutions are restricted to the natural numbers (Matiyjasevich 1996). We can equivalently consider polynomial equations $P(x_1, ..., x_n) = Q(x_1, ..., x_n)$ where all coefficients in the polynomial expressions $P(x_1, \dots, x_n)$ and $Q(x_1, \dots, x_n)$ are non-negative. We then can write a program Π consisting of the rules

$$
\begin{aligned}
&\mathtt{n}(0) \leftarrow .\\
&\mathtt{n}(X) \leftarrow \mathtt{n}(Y), X =_{\mathbb{N}} 1 + Y.\\
&\mathtt{sol} \leftarrow \mathtt{n}(X_1), ..., \mathtt{n}(X_n), Y =_{\mathbb{N}} P(X_1, ..., X_n), Y =_{\mathbb{N}} Q(X_1, ..., X_n).\\
&\bot \leftarrow \neg \mathtt{sol}.
\end{aligned}
$$

The program Π is safe and it has a (unique) equilibrium model (in which `sol` is true) iff a solution to $P(x_1, ..., x_n) = Q(x_1, ..., x_n)$ exists. Furthermore, the existence of an equilibrium model is equivalent to $\Pi \equiv_s \Pi \setminus \{\bot \leftarrow \neg \mathtt{sol}\}$.

Definition IV (Finite Groundability)**.** *Let σ some semiring signature and Π an $\mathscr{AC}$-program over σ. Then Π is* finitely groundable *if there is a signature $\sigma' = \langle \mathscr{D}, \mathscr{P}, \mathscr{X}, \mathscr{S}, r\rangle$ s.t. the equilibrium models of Π over σ are the same as the equilibrium models over σ' and $\mathscr{D}$ is finite.*

Theorem V. *For finitely ground programs over $\sigma' = \langle \mathscr{D}, \mathscr{P}, \mathscr{X}, \mathscr{S}, r\rangle$ ($|\mathscr{D}| < \infty$) that only use computable semirings, SAT and SE are decidable.*

Proof. We can replace universally quantified formulas with finite conjunctions over all the substitutions and existentially quantifies formulas with finite disjunctions over all the substitutions.

For variable free programs we have decidability when all the semirings are computable. □

Theorem VI. *Let Π be a safe program over σ without value invention, where all algebraic constraints in heads are domain restricted. Then Π is finitely ground over $\sigma' = \langle \mathscr{D}, \mathscr{P}, \mathscr{X}, \mathscr{S}, r\rangle$, where $\mathscr{D}$ is the subset of domain values that occur in Π.*

Proof. Let $\mathscr{I}$ be a σ-interpretation s.t. $(\mathscr{I}, \mathscr{I}, T) \models \Pi$. Then for $\mathscr{I}'$ obtained from $\mathscr{I}$ by removing all atoms that contain constants not from $\mathscr{D}$, we have $(\mathscr{I}', \mathscr{I}, H) \models \Pi$. This can be seen as follows: Assume $r \in \Pi$ with global variables $x_1, \dots, x_n$. Since r is safe and does not contain value

invention, the body of r can only be satisfied for substitutions of the variables with elements from $\mathscr{D}$. Therefore, if the head of r is an atom $p(\vec{x})$, we can only derive $p(\vec{\xi})$ for substitutions from $\mathscr{D}$ and therefore, if the rule was satisfied previously, it is still satisfied.

Otherwise, if the head of r is a constraint, we know that it is domain restricted, i.e. of the form

$$k \sim_{\mathscr{R}} \neg\neg\alpha(\vec{X}) * (\alpha(\vec{X}) \to \beta(\vec{X})) * \gamma(\vec{X}),$$

where $\alpha(\vec{X}), \beta(\vec{X})$ are syntactically domain independent and all atoms in $\gamma(\vec{X})$ are locally ground.

Let $\vec{\xi}$ be some assignments to $\vec{X}$ over the original domain.

We consider first $\neg\neg\alpha(\vec{\xi})$. It holds that

$$\begin{aligned}
& [\![\neg\neg\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_H) = e_\oplus \iff [\![\neg\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_H) = e_\otimes \vee [\![\neg\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T) = e_\otimes \\
& \iff [\![\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_H) = e_\oplus \wedge [\![\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T) = e_\oplus \vee [\![\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T) = e_\oplus \\
& \iff [\![\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T) = e_\oplus
\end{aligned}$$

Therefore this part of the formula is not influenced by $\mathscr{I}'$.

Secondly we consider $\alpha(\vec{\xi}) \to \beta(\vec{\xi})$. We only need to consider this value if $[\![\neg\neg\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_H)$ is unequal to zero, i.e. if $[\![\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T)$ is unequal to zero. Now if $[\![\alpha(\vec{\xi}) \to \beta(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T)$ is unequal to zero this implies that $[\![\beta(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T)$ is unequal to zero. If all the values in $\vec{\xi}$ are from $\mathscr{D}$, there is no change. Otherwise we know that $[\![\alpha(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_H) = [\![\beta(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_H) = e_\oplus$ since $\alpha(\vec{X})$ and $\beta(\vec{X})$ are syntactically domain independent and therefore have value $e_\oplus$ for values that are not mentioned in the interpretation $\mathscr{I}'$ (see proof of the invariance of the support for syntactically domain independent weighted formulas). It follows that $[\![\alpha(\vec{\xi}) \to \beta(\vec{\xi})]\!]_{\mathscr{R}}(\mathscr{I}_T)$ is also unequal to zero.

Since $\gamma(\vec{X})$ contains only locally ground atoms, the restriction of the interpretation to $\mathscr{I}'$ does not change the value of $\gamma(\vec{\xi})$.

Therefore

$$[\![\neg\neg\alpha(\vec{X}) * (\alpha(\vec{X}) \to \beta(\vec{X})) * \gamma(\vec{X})]\!]_{\mathscr{R}}(\mathscr{I}_H) = [\![\neg\neg\alpha(\vec{X}) * (\alpha(\vec{X}) \to \beta(\vec{X})) * \gamma(\vec{X})]\!]_{\mathscr{R}}(\mathscr{I}_T)$$

and

$$\begin{aligned}
& \mathscr{I}_H \models k \sim_{\mathscr{R}} \neg\neg\alpha(\vec{X}) * (\alpha(\vec{X}) \to \beta(\vec{X})) * \gamma(\vec{X}) \\
\iff & \mathscr{I}_T \models k \sim_{\mathscr{R}} \neg\neg\alpha(\vec{X}) * (\alpha(\vec{X}) \to \beta(\vec{X})) * \gamma(\vec{X}).
\end{aligned}$$

We see that since $(\mathscr{I}, \mathscr{I}, T) \models \Pi$ also $(\mathscr{I}', \mathscr{I}, T) \models \Pi$. Therefore $\mathscr{I}$ can only be an equilibrium model if it contains only constants from $\mathscr{D}$, which implies that Π is finitely ground over σ' □

Theorem 25. *For safe programs without value invention where all algebraic constraints in rule heads are domain restricted and all semirings are computable, both SAT and SE are decidable.*

Proof. This result follows easily from Theorems V and VI. □

References

CABALAR, P., FANDINNO, J., SCHAUB, T., AND WANKO, P. 2020. An ASP semantics for constraints involving conditional aggregates. *arXiv preprint arXiv:2002.06911*.

CASSAIGNE, J., HALAVA, V., HARJU, T., AND NICOLAS, F. 2014. Tighter undecidability bounds for matrix mortality, zero-in-the-corner problems, and more. *CoRR abs/1404.0644*.

DANTSIN, E., EITER, T., GOTTLOB, G., AND VORONKOV, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv. 33,* 3, 374–425.

DEMOLOMBE, R. 1992. Syntactical characterization of a subset of domain-independent formulas. *Journal of the ACM (JACM) 39,* 1, 71–94.

GREEN, T. J., KARVOUNARAKIS, G., AND TANNEN, V. 2007. Provenance semirings. In *Proc. ACM PODS'07*. ACM, 31–40.

LIFSCHITZ, V., PEARCE, D., AND VALVERDE, A. 2001. Strongly equivalent logic programs. *ACM TOCL 2,* 4, 526–541.

LIFSCHITZ, V., PEARCE, D., AND VALVERDE, A. 2007. A characterization of strong equivalence for logic programs with variables. In *International Conference on Logic Programming and Nonmonotonic Reasoning*. Springer, 188–200.

LIN, F. 2002. Reducing strong equivalence of logic programs to entailment in classical propositional logic. *KR 2*, 170–176.

MATIYJASEVICH, Y. 1996. Hilbert's tenth problem: what can we do with Diophantine equations? English version of a talk given by the author. Available at `http://logic.pdmi.ras.ru/~yumat/personaljournal/H10history/H10histe.pdf`.

PEARCE, D. AND VALVERDE, A. 2006. Quantified Equilibrium Logic and the First Order Logic of Here-and-There. *Technical Report MA-06-02*.