Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning

Forgetting in Logic Programs under Strong Equivalence

Yisong Wang
Department of Computer Science,
Guizhou University, China

Yan Zhang and **Yi Zhou**
Intelligent Systems Laboratory,
University of Western Sydney, Australia

Mingyi Zhang
Guizhou Academy of Sciences,
Guiyang, China

Abstract

In this paper, we propose a semantic forgetting for arbitrary logic programs (or propositional theories) under answer set semantics, called *HT-forgetting*. The HT-forgetting preserves strong equivalence in the sense that strongly equivalent logic programs will remain strongly equivalent after forgetting the same set of atoms. The result of an HT-forgetting is always expressible by a logic program, and in particular, the result of an HT-forgetting in a Horn program is expressible in a Horn program; and a representation theorem shows that HT-forgetting can be precisely characterized by Zhang-Zhou's four forgetting postulates under the logic of here-and-there. We also reveal underlying connections between HT-forgetting and classical forgetting, and provide complexity results for decision problems.

Introduction

Motivated from Lin and Reiter's seminal work (Lin and Reiter 1994), the notion of forgetting in classical propositional and first-order logics has attracted extensive interests in KR community (Lang and Marquis 2010). In recent years, researchers have developed forgetting notions and theories in other non-classical logic systems from various perspectives, such as forgetting in logic programs (Zhang and Foo 2006; Eiter and Wang 2008; Wong 2009), forgetting in description logic (Wang et al. 2010; Lutz and Wolter 2011), and knowledge forgetting in modal logic (Zhang and Zhou 2009; Su et al. 2009; Liu and Wen 2011).

It is easy to see that for classical propositional logic, forgetting preserves logical equivalence. That is, logically equivalent formulas (theories) will remain logically equivalent after forgetting the same set of atoms. For logic programs, the issue of logical equivalence is rather complicated due to its different notions of "equivalence": (weak) equivalence and strong equivalence.

There have been several attempts to define the notion of forgetting in logic programs, but none of these approaches is fully satisfactory. Zhang and Foo (2006) first defined syntax oriented weak and strong forgetting notions for normal logic programs. But these forgetting notions preserve neither (weak) equivalence nor strong equivalence. Eiter and Wang (2008) then proposed a semantic forgetting for consistent disjunctive logic programs, which preserves equivalence but not strong equivalence. They specifically indicated the importance of preserving strong equivalence in logic programming forgetting and raised this issue as a future work. Wong (2009) proposed two forgetting operators for disjunctive logic programs. Although Wong's forgetting indeed preserves strong equivalence, it may lose the intuition of weakening under various circumstances (see Related Work for details).

In addition to preserving strong equivalence, expressibility is another desired criterion for logic programming forgetting. Ideally we would expect that the result of forgetting some atoms from a logic program is still expressible by a logic program. Finally, we believe that as a way of weakening, forgetting in logic programs should obey some common intuitions shared by forgetting in classical logics. For instance, forgetting something from a logic program should lead to a weaker program in certain sense. On the other hand, such weakening should only be associated to the relevant information that has been forgotten. For this purpose, Zhang and Zhou (2009) proposed four forgetting postulates to formalize these common intuitions and showed that forgetting in classical propositional logic and modal logic S5 can be precisely captured by these postulates. Interestingly, none of previous forgetting notions in logic programs actually satisfies Zhang-Zhou's postulates.

In summary, we consider the following criteria that a forgetting notion in logic program should meet:

- Expressibility. The result of forgetting in an arbitrary logic program should also be expressible via an arbitrary logic program;
- Preserving strong equivalence. Two strongly equivalent programs should remain strongly equivalence after forgetting the same set of atoms;
- Satisfying common intuitions of forgetting. Preferably, forgetting in logic programs should be semantically characterized by Zhang-Zhou's four forgetting postulates.

In this paper we present a comprehensive study on forgetting in the context of arbitrary logic programs (propositional theories) under answer set semantics. In our approach, a program Π is viewed as a theory of the logic of here-and-there (or simply called HT logic), then forgetting a set V of

atoms from Π, is the theory consisting of the consequences of Π in HT logic that mention no atoms from V. Our semantic forgetting meets all above criteria, and hence is of primary advantages comparing to previous logic program forgetting notions. We also investigate the relationship between the HT-forgetting and classical forgetting, and the relating computational complexity of HT-forgetting.

Preliminaries

Consider a propositional language $\mathcal{L}$ over a finite set $\mathcal{A}$ of propositional atoms ($\mathcal{A}$ is also called the *signature* of $\mathcal{L}$). *Formulas* of $\mathcal{L}$ are built from $\mathcal{L}$'s signature $\mathcal{A}$ and the 0-place connective $\bot$ ("*false*") using the binary connectives $\wedge, \vee$ and $\supset$[1]. $\top$ ("*true*") is the shorthand of $\bot \supset \bot$, $\neg\varphi$ for $\varphi \supset \bot$, and $\psi \leftrightarrow \phi$ for $(\psi \supset \phi) \wedge (\phi \supset \psi)$. A *theory* is a set of propositional formulas. An *interpretation* is a set I of atoms from $\mathcal{A}$, where each atom of $\mathcal{A}$ is viewed to be true if it is in I, and false otherwise. Then notions of *model* and *satisfaction relation* $\models$ are defined in a standard way. For two propositional formulas ϕ and ψ, $\phi \equiv \psi$ is used to denote $\phi \models \psi$ and $\psi \models \phi$.

Forgetting in propositional logic

Let V, M_1 and M_2 be sets of atoms. We say that M_1 and M_2 are *V-identical*, denoted by $M_1 \sim_V M_2$, if M_1 and M_2 agree on everything except possibly on V, i.e. $M_1 \setminus V = M_2 \setminus V$. Given a theory Σ, a theory, denoted as $\mathsf{Forget}(\Sigma, V)$, is a *result of forgetting V in Σ*, iff for any set M' of atoms, M' is a model of Σ iff there exists a model M of Σ such that $M \sim_V M'$. Alternatively, $\Sigma' \equiv \mathsf{Forget}(\Sigma, V)$ iff

$$\mathsf{Mod}(\Sigma') = \{M' \mid \exists M \models \Sigma \text{ s.t } M \sim_V M'\},$$

where $\mathsf{Mod}(\Sigma)$ denotes the set of models of Σ. This is a semantic forgetting defined in (Lin and Reiter 1994). A syntactic counterpart of the semantic forgetting is given in (Lang, Liberatore, and Marquis 2003) as: $\mathsf{Forget}(\Sigma, p) = \Sigma[p/\top] \vee \Sigma[p/\bot]$, and $\mathsf{Forget}(\Sigma, V \cup \{p\}) = \mathsf{Forget}(\mathsf{Forget}(\Sigma, p), V)$. These two definitions of forgetting are equivalent (cf. Corollary 5 of (Lang, Liberatore, and Marquis 2003)).

Answer sets for propositional theories

Given a formula ψ and a set X of atoms, the *reduct* ψ^X of ψ relative to X is obtained from ψ recursively via the following steps:

- for each atom p, if $X \models p$ then p^X is p, otherwise it is $\bot$; $\bot^X = \bot$; and
- for each formula ψ and ϕ, if $X \models \psi \otimes \phi$ then $(\psi \otimes \phi)^X$ is $\psi^X \otimes \phi^X$; otherwise it is $\bot$, where $\otimes \in \{\vee, \wedge, \supset\}$.

The *reduct* Π^X of a propositional theory Π relative to X is $\{\psi^X | \psi \in \Pi\}$. A set X of atoms is an *answer set* (or *stable model*) of a set of propositional formulas Π if X is a minimal model (in terms of set inclusion) satisfying Π^X (Ferraris 2011). Under this semantics, one should note that $\neg\neg p$ is not "equivalent to" p, as $\neg\neg p$ has no answer set while $\{p\}$ is the unique answer set of p. Two theories Π_1 and Π_2 are *strongly equivalent* if $\Pi_1 \cup \Sigma$ and $\Pi_2 \cup \Sigma$ have the same answer sets for any theory Σ. Cabalar and Ferraris (2007) showed that a propositional theory (under answer set semantics) exactly captures a logic program with nested expressions (Lifschitz, Tang, and Turner 1999).

The logic of here-and-there

The syntax of HT logic is the same as classical propositional logic. An *HT-interpretation* is a pair $\langle H, T\rangle$ such that $H \subseteq T \subseteq \mathcal{A}$. The satisfiability relation between an HT-interpretation $\langle H, T\rangle$ and a formula ψ, denoted by $\langle H, T\rangle \models \psi$, is recursively defined:

- $\langle H, T\rangle \models p$ if $p \in H$;
- $\langle H, T\rangle \not\models \bot$;
- $\langle H, T\rangle \models \psi_1 \vee \psi_2$ if $\langle H, T\rangle \models \psi_1$ or $\langle H, T\rangle \models \psi_2$;
- $\langle H, T\rangle \models \psi_1 \wedge \psi_2$ if $\langle H, T\rangle \models \psi_1$ and $\langle H, T\rangle \models \psi_2$;
- $\langle H, T\rangle \models \psi_1 \supset \psi_2$ if both (i) $T \models \psi_1 \supset \psi_2$, and (ii) $\langle H, T\rangle \models \psi_1$ implies $\langle H, T\rangle \models \psi_2$.

An HT-interpretation $\langle H, T\rangle$ is an *HT-model* of ψ if $\langle H, T\rangle \models \psi$. An HT-model $\langle T, T\rangle$ of ψ is an *equilibrium model* of ψ if there is no T' such that $T' \subset T$ and $\langle T', T\rangle \models \psi$. The logic based on this semantics is called *equilibrium logic* (Pearce 1996). The satisfiability relation, HT-model and equilibrium model are extended to theories in a standard way. We denote $\mathsf{Mod}_{\mathsf{ht}}(\Sigma)$ the set of HT-models of the theory Σ. In particular, if Σ is a singleton $\{\psi\}$, we simply write it as $\mathsf{Mod}_{\mathsf{ht}}(\psi)$. Let Π and Σ be two theories. By $\Pi \models_{\mathsf{ht}} \Sigma$, we mean that every HT-model of Π is an HT-model of Σ, and by $\Pi \equiv_{\mathsf{ht}} \Sigma$ we mean $\Pi \models_{\mathsf{ht}} \Sigma$ and $\Sigma \models_{\mathsf{ht}} \Pi$. In the latter case, we call Π and Σ are *HT-equivalent*. It has been shown that, a set X of atoms is an answer set of a logic program Π iff $\langle X, X\rangle$ is an equilibrium model of Π. In addition, two logic programs Π and Σ are strongly equivalent iff $\Pi \equiv_{\mathsf{ht}} \Sigma$ (Pearce, Tompits, and Woltran 2001; Lifschitz, Pearce, and Valverde 2001; Ferraris 2011). The following proposition shows some basic properties of HT logic that we will need in our next study.

Proposition 1 *Let ϕ and ψ be two formulas and $\langle X, Y\rangle$ an HT-interpretation.*

(i) If $\langle X, Y\rangle \models \phi$ then $\langle Y, Y\rangle \models \phi$ (i.e., $Y \models \phi$),
(ii) $\langle X, Y\rangle \models \neg\phi$ iff $Y \models \neg\phi$,
(iii) $\langle X, Y\rangle \models \phi$ iff $X \models \phi^Y$,
(iv) If $\phi \equiv_{\mathsf{ht}} \psi$, then $\phi \equiv \psi$,
(v) $\neg\bot \equiv_{\mathsf{ht}} \neg\neg\top \equiv_{\mathsf{ht}} \top$ and $\neg\top \equiv_{\mathsf{ht}} \neg\neg\bot \equiv_{\mathsf{ht}} \bot$.

HT-Forgetting in Logic Programs

Differently from previous approaches, our forgetting notion in logic programs will be based on the logic of here-and-there, which will lead to meet all criteria of forgetting we addressed earlier. Let $\mathcal{K}_1 = \langle H_1, T_1\rangle$ and $\mathcal{K}_2 = \langle H_2, T_2\rangle$ be two HT-interpretations and V a set of atoms. $\mathcal{K}_1$ and $\mathcal{K}_2$ are said to be *V-identical*, denoted as $\mathcal{K}_1 \sim_V \mathcal{K}_2$, whenever $H_1 \sim_V H_2$ and $T_1 \sim_V T_2$.

[1] In the rest of this paper, whenever there is no confusion, we may not explicitly mention the signature when we talk about formulas of $\mathcal{L}$.

Definition 1 *Let ψ be a formula and V a set of atoms. A formula ϕ is called a* result of HT-forgetting V in ψ, *iff the following condition holds:*

$$\mathsf{Mod_{ht}}(\phi) = \{\langle H, T\rangle \mid \langle H, T\rangle \textit{ is an HT-interpretation s.t. } \exists \langle X, Y\rangle \in \mathsf{Mod_{ht}}(\psi) \textit{ and } \langle X, Y\rangle \sim_V \langle H, T\rangle\}.$$

As we will see later, the forgetting result always exists and it is unique (up to the equivalence in here-and-there), we will denote the forgetting result by $\mathsf{Forget_{ht}}(\Pi, V)$ in what follows. From Definition 1, it is not difficult to see that HT-forgetting is independent of the order of forgetting atoms.

Proposition 2 *Let ψ be a formula, V_1 and V_2 two sets of atoms. Then we have* $\mathsf{Forget_{ht}}(\mathsf{Forget_{ht}}(\psi, V_1), V_2) \equiv_{\mathsf{ht}} \mathsf{Forget_{ht}}(\mathsf{Forget_{ht}}(\psi, V_2), V_1) \equiv_{\mathsf{ht}} \mathsf{Forget_{ht}}(\psi, V_1 \cup V_2)$.

As HT-interpretations are related to a given signature, in what follows, we shall assume that the signature of a formula/theory is implicitly given by the atoms occurring in the formula/theory, unless explicitly stated otherwise.

Semantic Characterizations

Unlike propositional logic, HT logic does not hold a model characteristic property in general. That is, given a class of HT-interpretations, there may not exist a formula whose HT-models exactly correspond to those HT-interpretations. For example, let $\mathcal{M} = \{\langle \emptyset, \{a\}\rangle\}$, we can show that there is no formula that has a unique HT-model $\langle \emptyset, \{a\}\rangle$. To see this, suppose that there is a formula ψ such that $\mathsf{Mod_{ht}}(\psi) = \mathcal{M}$, then we have $\langle \{a\}, \{a\}\rangle \models_{\mathsf{ht}} \psi$ by (i) of Proposition 1. But this is not the case. Hence, as a semantic forgetting notion, the study on the expressibility of HT-forgetting is necessary.

Expressibility

To begin with, we first introduce the notion of irrelevance which has a close connection to forgetting. A formula ψ is *HT-irrelevant* to a set V of atoms, denoted as $\mathsf{IR_{ht}}(\psi, V)$, if there exists a formula ϕ mentioning no atoms from V and $\psi \equiv_{\mathsf{ht}} \phi$. For convenience, in the following, we will simply write a singleton set $\{\alpha\}$ as α, and thus we may denote $\mathsf{Forget_{ht}}(\psi, \{p\})$ as $\mathsf{Forget_{ht}}(\psi, p)$, and $\mathsf{IR_{ht}}(\psi, \{p\})$ as $\mathsf{IR_{ht}}(\psi, p)$, etc..

Note that Definition 1 is semantic, which does not guarantee the existence of a formula ϕ such that $\mathsf{Mod_{ht}}(\phi) = \mathsf{Mod_{ht}}(\mathsf{Forget_{ht}}(\psi, V))$. However, the following theorem states that such formula always exists.

Theorem 1 (Expressibility theorem) *Let ψ be a formula and V a set of atoms. There always exists a formula ϕ such that ϕ is a result of HT-forgetting V in ψ.*

As one knows that disjunctive programs, positive programs, normal logic programs and Horn programs are four types of special cases of (arbitrary) logic programs under our setting. Then it is interesting to consider whether the expressibility result also holds for each of these special programs. For instance, we would like to know whether the result of forgetting in a disjunctive (positive, normal, and Horn) program is still expressible by a disjunctive (resp. positive, normal, and Horn) program. It turns out that the answer is negative for HT-forgetting in disjunctive, positive and normal programs.

Example 1 Consider the following normal logic program Π over signature $\{p, q\}$:

$$\neg p \supset q, \qquad \neg q \supset p, \qquad p \wedge q \supset \bot.$$

Π has two HT-models: $\langle \{p\}, \{p\}\rangle$ and $\langle \{q\}, \{q\}\rangle$. Then $\mathsf{Mod_{ht}}(\mathsf{Forget_{ht}}(\Pi, p))$ contains the HT-interpretations that are $\{p\}$-identical to one of the following HT-interpretations:

$$\langle \emptyset, \emptyset\rangle, \qquad \langle \{q\}, \{q\}\rangle,$$

from which we conclude that $\mathsf{Forget_{ht}}(\Pi, p) \equiv_{\mathsf{ht}} q \vee \neg q$. It is shown that $q \vee \neg q$ cannot be expressed as a normal or disjunctive program in the sense that there is no normal or disjunctive program Π which is strongly equivalent to $q \vee \neg q$ (Eiter, Tompits, and Woltran 2005). □

Example 2 Let Π be a positive logic program over signature $\{p, q, r\}$ as follows:

$$p \vee q \vee r, \qquad p \wedge q \supset r, \qquad p \wedge r \supset q, \qquad q \wedge r \supset p.$$

By HT-forgetting q from Π, we have $\mathsf{Forget_{ht}}(\Pi, q) \equiv_{\mathsf{ht}} (\neg r \supset p \vee \neg p) \wedge (\neg p \supset r \vee \neg r)$, which cannot be expressed by any positive logic program. □

HT-forgetting in Horn programs is of special interest, because unlike disjunctive, positive and normal programs, the result of HT-forgetting in a Horn program is always expressible by a Horn program.

Theorem 2 (Horn program expressibility) *Let Π be a Horn logic program and V a set of atoms. Then there exists a Horn logic program Π' such that* $\mathsf{Forget_{ht}}(\Pi, V) \equiv_{\mathsf{ht}} \Pi'$.

Strong equivalence and other properties

One can easily see that our HT-forgetting preserves strong equivalence. From Definition 1 and Theorem 1, the following result is obvious.

Proposition 3 *Let ψ and φ be two formulas and V a set of atoms. If $\psi \equiv_{\mathsf{ht}} \varphi$, then* $\mathsf{Forget_{ht}}(\psi, V) \equiv_{\mathsf{ht}} \mathsf{Forget_{ht}}(\varphi, V)$.

The property of preserving strong equivalence of HT-forgetting then immediately follows from the equivalent relationship between strong equivalence and HT-equivalence (Lifschitz, Pearce, and Valverde 2001). The following proposition illustrates some essential properties of HT-forgetting.

Proposition 4 *Let ψ and ϕ be two formula and V a set of atoms. Then the following results hold.*

(i) ψ *has an HT-model iff* $\mathsf{Forget_{ht}}(\psi, V)$ *has.*

(ii) $\psi \models_{\mathsf{ht}} \mathsf{Forget_{ht}}(\psi, V)$.

(iii) If $\psi \models_{\mathsf{ht}} \phi$ *then* $\mathsf{Forget_{ht}}(\psi, V) \models_{\mathsf{ht}} \mathsf{Forget_{ht}}(\phi, V)$.

(iv) $\mathsf{Forget_{ht}}(\psi \vee \phi, V) \equiv_{\mathsf{ht}} \mathsf{Forget_{ht}}(\psi, V) \vee \mathsf{Forget_{ht}}(\phi, V)$.

(v) $\mathsf{Forget_{ht}}(\psi \wedge \phi, V) \models_{\mathsf{ht}} \mathsf{Forget_{ht}}(\psi, V) \wedge \mathsf{Forget_{ht}}(\phi, V)$.

(vi) $\mathsf{Forget_{ht}}(\psi \wedge \phi, V) \equiv_{\mathsf{ht}} \mathsf{Forget_{ht}}(\psi, V) \wedge \phi$ *if* $\mathsf{IR_{ht}}(\phi, V)$.

Forgetting postulates

Zhang and Zhou (2009) proposed four forgetting postulates in their work of knowledge forgetting, and showed that their knowledge forgetting can be precisely characterized by these postulates. In the following, we show that HT-forgetting is exactly captured by these postulates, which we think is one major advantage over other logic program forgetting approaches.

The below proposition actually implies the uniform interpolation property (Visser 1996) of the logic of here-and-there.

Proposition 5 *Let ψ and φ be two formulas, V a set of atoms and $\mathsf{IR}(\varphi, V)$. Then we have*

$$\psi \models_{\mathsf{ht}} \varphi \quad \textit{iff} \quad \mathsf{Forget}_{\mathsf{ht}}(\psi, V) \models_{\mathsf{ht}} \varphi.$$

Let ψ and ϕ be two formulas and V a set of atoms. The following are Zhang-Zhou's four postulates under the logic of here-and-there.

(W) Weakening: $\psi \models_{\mathsf{ht}} \phi$.

(PP) Positive persistence: if $\mathsf{IR}_{\mathsf{ht}}(\xi, V)$ and $\psi \models_{\mathsf{ht}} \xi$ then $\phi \models_{\mathsf{ht}} \xi$.

(NP) Negative persistence: if $\mathsf{IR}_{\mathsf{ht}}(\xi, V)$ and $\psi \not\models_{\mathsf{ht}} \xi$ then $\phi \not\models_{\mathsf{ht}} \xi$.

(IR) Irrelevance: $\mathsf{IR}_{\mathsf{ht}}(\phi, V)$.

By specifying $\phi \equiv_{\mathsf{ht}} \mathsf{Forget}_{\mathsf{ht}}(\psi, V)$, (W), (PP), (NP) and (IR) are called *postulates for HT-forgetting*. Weakening (W) requires that forgetting results in weaker knowledge. The postulates of positive persistence (PP) and negative persistence (NP) simply state that forgetting a set of atoms should not affect those positive or negative information respectively that is irrelevant to this set of atoms. Finally, irrelevance (IR) means that after forgetting, the resulting knowledge should be irrelevant to those forgotten atoms.

Theorem 3 (Representation theorem) *Let ψ and ϕ be two formulas and V a set of atoms. Then the following statements are equivalent:*

(i) $\phi \equiv_{\mathsf{ht}} \mathsf{Forget}_{\mathsf{ht}}(\psi, V)$.

(ii) $\phi \equiv_{\mathsf{ht}} \{\varphi \mid \psi \models_{\mathsf{ht}} \varphi \textit{ and } \mathsf{IR}_{\mathsf{ht}}(\varphi, V)\}$.

(iii) Postulates (W), (PP), (NP) and (IR) hold.

HT-forgetting and Classical Forgetting

It has been shown that strong equivalence of logic programs may be related to the equivalence of propositional logic (Pearce, Tompits, and Woltran 2001). As the HT-forgetting does preserve strong equivalence, it is worth exploring further connections between HT-forgetting and the classical one.

Proposition 6 *Let ψ be a formula and V a set of atoms. The following results hold.*

(i) $\mathsf{Forget}_{\mathsf{ht}}(\psi, V) \equiv \mathsf{Forget}(\psi, V)$.

(ii) $\mathsf{Forget}(\neg\psi, V) \models_{\mathsf{ht}} \mathsf{Forget}_{\mathsf{ht}}(\neg\psi, V)$.

The result (i) in Proposition 6 simply says that HT-forgetting and classical forgetting are equivalent under the classical propositional logic. From this result and Theorem 2, we immediately have the following corollary.

Corollary 4 *Let Π be a Horn program and V a set of atoms. Then $\mathsf{Forget}(\Pi, V)$ is expressible by a Horn program.*

We also note that the converse of (ii) of Proposition 6 does not hold. That is, we usually do not have $\mathsf{Forget}_{\mathsf{ht}}(\neg\psi, V) \models_{\mathsf{ht}} \mathsf{Forget}(\neg\psi, V)$. For example, let $\psi = \neg(p \wedge \neg\neg q)$, then we have that $\mathsf{Forget}(\neg\psi, q) \equiv p$ and $\mathsf{Forget}_{\mathsf{ht}}(\neg\psi, q) \equiv_{\mathsf{ht}} \neg\neg p$. Clearly, $\neg\neg p \not\models_{\mathsf{ht}} p$.

Nevertheless, together with (i) of Proposition 6, the following result states that for Horn programs, HT-forgetting and classical forgetting are also equivalent under HT logic.

Proposition 7 *Let Π be a Horn program, V a set of atoms and Π' a Horn program such that $\Pi' \equiv \mathsf{Forget}(\Pi, V)$. Then $\mathsf{Forget}_{\mathsf{ht}}(\Pi, V) \equiv_{\mathsf{ht}} \Pi'$.*

Proposition 7 provides a method of computing HT-forgetting in a Horn program through its corresponding classical forgetting, since we know that $\mathsf{Forget}(\Pi, V \cup \{p\}) \equiv \mathsf{Forget}(\mathsf{Forget}(\Pi, p), V)$, and $\mathsf{Forget}(\psi, p) \equiv \psi[p/\bot] \vee \psi[p/\top]$ for any formula (program) ψ.

Proposition 8 *Let ψ and ϕ be two formulas and V a set of atoms. The following results hold.*

(i) $\phi \equiv \mathsf{Forget}(\psi, V)$ *iff* $\neg\neg\phi \equiv_{\mathsf{ht}} \mathsf{Forget}_{\mathsf{ht}}(\neg\neg\psi, V)$,

(ii) $\mathsf{Forget}(\phi, V) \equiv \mathsf{Forget}(\psi, V)$ *iff* $\mathsf{Forget}_{\mathsf{ht}}(\neg\neg\phi, V) \equiv_{\mathsf{ht}} \mathsf{Forget}_{\mathsf{ht}}(\neg\neg\psi, V)$.

Computational Complexity

Given a formula ψ and a set V of atoms, from Theorem 3 (i.e. (ii) in the representation theorem), we can see that the result of the HT-forgetting V in ψ, $\mathsf{Forget}_{\mathsf{ht}}(\psi, V)$, can be generated by computing all logical consequences of ψ which are HT-irrelevant to V. Indeed, since there is a sound and complete axiomatic system for the logic of here-and-there (Jongh and Hendriks 2003), this is feasible. Nevertheless, it is also observed that from a computational viewpoint, like the classical forgetting, the process of generating $\mathsf{Forget}_{\mathsf{ht}}(\psi, V)$ would be expensive as shown by the below theorem.

Theorem 5 *Let ψ and ϕ be two formulas and V a set of atoms. We have*

(i) deciding if $\psi \equiv_{\mathsf{ht}} \mathsf{Forget}_{\mathsf{ht}}(\phi, V)$ *is* Π_2^P*-complete,*

(ii) deciding if $\psi \models_{\mathsf{ht}} \mathsf{Forget}_{\mathsf{ht}}(\phi, V)$ *is* Π_2^P*-complete,*

(iii) deciding if $\mathsf{Forget}_{\mathsf{ht}}(\psi, V) \models_{\mathsf{ht}} \phi$ *is coNP-complete.*

Related Work

As mentioned in Introduction, several approaches of forgetting in logic programs have been developed earlier in the literature. While Zhang and Foo's weak and strong forgetting (Zhang and Foo 2006), and Eiter and Wang's semantic forgetting (Eiter and Wang 2008) do not preserve strong equivalence, Wong's forgetting operators (Wong 2009) does. In the section we compare our HT-forgetting with the ones proposed by Wong.

Wong developed his forgetting for disjunctive logic programs. Differently from Zhang and Foo's and Eiter and Wang's approaches, Wong's forgetting is defined based on the logic of here-and-there. In this sense, Wong's approach

probably shares a common logic ground with HT-forgetting. Wong also defined two forgetting operators $\mathsf{F_S}$ and $\mathsf{F_W}$, which correspond to two series of program transformations. The interesting feature of Wong's forgetting is that it preserves strong equivalence.

However, a major issue with Wong's forgetting is that: on one hand, forgetting may cause unnecessary information loss; on the other hand, forgetting may also introduce extra information that we do not want, as illustrated by the following example.

Example 3 Let us consider the normal logic program Π consisting of:

$$a \leftarrow x, \qquad y \leftarrow a, not\ z,$$
$$q \leftarrow not\ p, \qquad p \leftarrow not\ q, \qquad \leftarrow p, q.$$

Then we have:

$\mathsf{F_S}(\Pi, \{a, p\}) \equiv_{\mathsf{ht}} \{y \leftarrow x, not\ z\}$,

$\mathsf{F_W}(\Pi, \{a, p\}) \equiv_{\mathsf{ht}} \{y \leftarrow x, not\ z, \quad \leftarrow x, \quad q \leftarrow\}$,

$\mathsf{Forget_{ht}}(\Pi, \{a, p\}) \equiv_{\mathsf{ht}} \{y \leftarrow x, not\ z, \quad q \leftarrow not\ not\ q\}$.

Since $\Pi \models_{\mathsf{ht}} \{q \leftarrow not\ not\ q\}$, which is irrelevant to atoms a and p, it seems to us that forgetting $\{a, p\}$ from Π should not affect this fact. But clearly $\mathsf{F_S}(\Pi, \{a, p\}) \not\models_{\mathsf{ht}} \{q \leftarrow not\ not\ q\}$. In this sense, we see $\mathsf{F_S}$ has lost some information that we wish to keep.

On the other hand, from the fact that $\Pi \not\models_{\mathsf{ht}} q$ but $\mathsf{F_W}(\Pi, \{a, p\}) \models_{\mathsf{ht}} q$, it appears that $\mathsf{F_W}$ may introduce unnecessary information, which indeed conflicts our intuition of program weakening via forgetting. □

Concluding Remarks

In the paper we proposed a semantic forgetting in arbitrary logic program which preserves strong equivalence and also satisfies other important semantic properties that previous forgetting approaches do not have.

Some related issues remain for the future work. Forgetting in logic programs has demonstrated its applications in conflict resolution and knowledge base update (Zhang and Foo 2006; Eiter and Wang 2008). We believe that our HT-forgetting can be used to develop a general framework for knowledge bases merging and update where each knowledge base is represented as an arbitrary logic program.

It is also an interesting topic to develop a notion of forgetting in first-order logic programs. It is not clear yet whether the HT-forgetting proposed in this paper may be extended to arbitrary first-order logic programs. Nevertheless, some general principles for forgetting such as preserving strong equivalence and forgetting postulates that we addressed in this paper may provide an essential guide for our study on this topic.

Acknowledgement

We thanks the anonymous reviewers for their detailed comments. Yisong Wang and Mingyi Zhang are partially supported by the Natural Science Foundation of China under grant 60963009, while Yisong is also partially supported by Open Funds of the State Key Laboratory of Computer Science of Chinese Academy of Science under grant SYSKF1106.

References

Cabalar, P., and Ferraris, P. 2007. Propositional theories are strongly equivalent to logic programs. *TPLP* 7(6):745–759.

Eiter, T., and Wang, K. 2008. Semantic forgetting in answer set programming. *AIJ* 172(14):1644–1672.

Eiter, T.; Tompits, H.; and Woltran, S. 2005. On solution correspondences in answer-set programming. In *IJCAI-2005*, 97–102. Edinburgh, Scotland, UK: Professional Book Center.

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

Jongh, D. D., and Hendriks, L. 2003. Characterization of strongly equivalent logic programs in intermediate logics. *TPLP* 3(3):259–270.

Lang, J., and Marquis, P. 2010. Reasoning under inconsistency: A forgetting-based approach. *AIJ* 174(12-13):799–823.

Lang, J.; Liberatore, P.; and Marquis, P. 2003. Propositional independence: Formula-variable independence and forgetting. *JAIR* 18:391–443.

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

Lifschitz, V.; Tang, L. R.; and Turner, H. 1999. Nested expressions in logic programs. *AMAI* 25(3-4):369–389.

Lin, F., and Reiter, R. 1994. Forget it! In *In Proceedings of the AAAI Fall Symposium on Relevance*, 154–159.

Liu, Y., and Wen, X. 2011. On the progression of knowledge in the situation calculus. In *IJCAI-2011*, 976–982. Barcelona, Catalonia, Spain: IJCAI/AAAI.

Lutz, C., and Wolter, F. 2011. Foundations for uniform interpolation and forgetting in expressive description logics. In *IJCAI-2011*, 989–995. Barcelona, Catalonia, Spain: IJCAI/AAAI.

Pearce, D.; Tompits, H.; and Woltran, S. 2001. Encodings for equilibrium logic and logic programs with nested expressions. In *EPIA-2001*, 306–320. London, UK: Springer-Verlag.

Pearce, D. 1996. A new logical characterisation of stable models and answer sets. In *NMELP'96*, volume 1216 of *LNCS*, 57–70. Bad Honnef, Germany: Springer.

Su, K.; Sattar, A.; Lv, G.; and Zhang, Y. 2009. Variable forgetting in reasoning about knowledge. *JAIR* 35:677–716.

Visser, A. 1996. Uniform interpolation and layered bisimulation. In *Gödel'96*, 139–164.

Wang, Z.; Wang, K.; Topor, R. W.; and Pan, J. Z. 2010. Forgetting for knowledge bases in dl-lite. *AMAI* 58(1-2):117–151.

Wong, K.-S. 2009. *Forgetting in Logic Programs*. Ph.D. Dissertation, The University of New South Wales.

Zhang, Y., and Foo, N. Y. 2006. Solving logic program conflict through strong and weak forgettings. *AIJ* 170(8-9):739–778.

Zhang, Y., and Zhou, Y. 2009. Knowledge forgetting: Properties and applications. *AIJ* 173(16-17):1525–1537.