Adding Operators. ACHTUNG NICHT FERTIG!

Thure Nebendahl

May 23, 2020

In this chapter we will extend the algorithm presented in [1] by the operators $\Box\phi$ (always), $\Box^{-}\phi$ (always in the past), $\Diamond\phi$ (eventually), $\Diamond^{-}\phi$ (some time in the past) that were already presented in the paper but not considered when defining the algorithm. We are extending the algorithm because these operators are necessary for a lot of relevant queries. First we will show how to modify the definitions of $\mathsf{eval}^n(\alpha)$, $\Phi_0(\psi)$ and $\Phi_i(\psi)$, before we will continue to prove why the modifications are correct. As a reminder the semantics of these four temporal queries (TQs) are defined as follows:

Definition 0.1 (semantics of TQs cf. Definition 3.3 in [1])**.** Let ϕ be a TQ, $\mathfrak{I} = (I_i)_{0\leq i\leq n}$ a sequence of interpretations over a common domain, $\mathfrak{a} : \mathsf{FVar}(\phi) \to \mathsf{N_C}$ a variable assignment, and i be an integer with $0 \leq i \leq n$. The *satisfaction relation* $\mathfrak{I}, i \models \mathfrak{a}(\phi)$ is defined by induction on the structure of ϕ as follows:

ϕ	$\mathfrak{I}, i \models \mathfrak{a}(\phi)$
$\Box\phi_1$	$\mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for all $k, i \leq k \leq n$
$\Box^{-}\phi_1$	$\mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for all $k, 0 \leq k \leq i$
$\Diamond\phi_1$	$\mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for some $k, i \leq k \leq n$
$\Diamond^{-}\phi_1$	$\mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for some $k, 0 \leq k \leq i$

Table 1: semantics of TQs

$\mathsf{FVar}(\phi)$ denotes the set of *free variables* of a TQ and is defined as the union of the sets $\mathsf{FVar}(\psi)$ of all queries ψ occurring in ϕ. $\mathsf{N_C}$ denotes a set of *constants*. If $\mathfrak{I}, i \models \mathfrak{a}(\phi)$, then $\mathfrak{a}$ is called an *answer* to ϕ w.r.t. $\mathfrak{I}$ at time point i. The set of all answers to ϕ w.r.t $\mathfrak{I}$ at time point i is denoted by $\mathsf{Ans}(\phi, \mathfrak{I}, i)$.

As in [1], we can show that

- $\Box\phi_1$ is equivalent to $\phi_1 \wedge \CIRCLE\Box\phi_1$;
- $\Diamond\phi_1$ is equivalent to $\phi_1 \vee \Circle\Diamond\phi_1$.

Because of the way $\Box$ is defined in [1] we have to use $\CIRCLE$ instead of $\Circle$ with the only difference that $\CIRCLE$ is tautological at the last time point. We can similarly show that

- $\Box^-\phi_1$ is equivalent to $\phi_1 \wedge \CIRCLE^-\Box^-\phi_1$;
- $\Diamond^-\phi_1$ is equivalent to $\phi_1 \vee \Circle^-\Diamond^-\phi_1$.

Analogously to the reason as mentioned above we have to use $\CIRCLE^-$ instead of $\Circle^-$ with the only difference that $\CIRCLE^-$ is tautological at the first time point. Thus at the last time point

- $\Box\phi_1$ is equivalent to ϕ_1 because $\CIRCLE\Box\phi_1$ is tautological
- $\Diamond\phi_1$ is equivalent to ϕ_1 because $\Circle\Diamond\phi_1$ does not have any answers

and at the first time point

- $\Box^-\phi_1$ is equivalent to ϕ_1 because $\CIRCLE^-\Box^-\phi_1$ is tautological
- $\Diamond^-\phi_1$ is equivalent to ϕ_1 because $\Circle^-\Diamond^-\phi_1$ does not have any answers

Proposition 0.2 (similar to Proposition 3.4 in [1]). *For* $\mathfrak{a} : \mathsf{FVar}(\phi) \to \mathsf{N_C}$ *and* $0 \le i \le n$*, we have*

1. $\mathfrak{I}, i \models \mathfrak{a}(\Box\phi_1)$ *iff*
   - $\mathfrak{I}, i \models \mathfrak{a}(\phi_1)$ *and*
   - $i < n$ *implies* $\mathfrak{I}, i+1 \models \mathfrak{a}(\Box\phi_1)$
2. $\mathfrak{I}, i \models \mathfrak{a}(\Box^-\phi_1)$ *iff*
   - $\mathfrak{I}, i \models \mathfrak{a}(\phi_1)$ *and*
   - $i > 0$ *implies* $\mathfrak{I}, i-1 \models \mathfrak{a}(\Box^-\phi_1)$
3. $\mathfrak{I}, i \models \mathfrak{a}(\Diamond\phi_1)$ *iff*
   - $\mathfrak{I}, i \models \mathfrak{a}(\phi_1)$ *or*
   - $i < n$ *and* $\mathfrak{I}, i+1 \models \mathfrak{a}(\Diamond\phi_1)$
4. $\mathfrak{I}, i \models \mathfrak{a}(\Diamond^-\phi_1)$ *iff*

- $\mathfrak{I}, i \models \mathfrak{a}(\phi_1)$ *or*
- $i > 0$ *and* $\mathfrak{I}, i-1 \models \mathfrak{a}(\Diamond^- \phi_1)$

ANNOTATION: CASES 1 AND 4 MAY BE SUFFICIENT

Proof. To prove the above proposition we will prove each equivalence. We will mainly show this on the basis of the semantics.

1. $\Box\phi_1 \equiv \phi_1 \wedge \bullet\Box\phi_1$

$$\mathfrak{I}, i \models \mathfrak{a}(\Box\phi_1) \tag{1}$$
$$\Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for all } k, i \leq k \leq n \tag{2}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ and } i < n \text{ implies } \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for all } k, i+1 \leq k \leq n \tag{3}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ and } i < n \text{ implies } \mathfrak{I}, i+1 \models \mathfrak{a}(\Box\phi_1) \tag{4}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1 \wedge \bullet\Box\phi_1) \tag{5}$$

(3) is equivalent to (2) because for $i < n$ in order to contain an answer the query needs to be satisfied now (at time point i) as well as at all future time points ($i + 1 \leq k \leq n$). Since $i < n$ is true the satisfaction of future time points solely depends on the second part of the "implies"-statement. In the case of $i = n$ this is equivalent as well because $\mathfrak{I}, i \models \mathfrak{a}(\phi_1) \Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for all $k, n \leq k \leq n$ and $i < n$ is not true.

2. $\Box^-\phi_1 \equiv \phi_1 \wedge \bullet^-\Box^-\phi_1$

$$\mathfrak{I}, i \models \mathfrak{a}(\Box^-\phi_1) \tag{6}$$
$$\Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for all } k, 0 \leq k \leq i \tag{7}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ and } i > 0 \text{ implies } \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for all } k, 0 \leq k \leq i-1 \tag{8}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ and } i > 0 \text{ implies } \mathfrak{I}, i-1 \models \mathfrak{a}(\Box^-\phi_1) \tag{9}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1 \wedge \bullet^-\Box^-\phi_1) \tag{10}$$

(8) is equivalent to (7) because for $i > 0$ in order to contain an answer the query needs to be satisfied now (at time point i) as well as at all past time points ($0 \leq k \leq i-1$). Since $i > 0$ is true the satisfaction of past time points solely depends on the second part of the "implies"-statement. In the case of $i = 0$ this is equivalent as well because $\mathfrak{I}, i \models \mathfrak{a}(\phi_1) \Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for all $k, 0 \leq k \leq 0$ and $i > 0$ is not true.

3. $\Diamond\phi_1 \equiv \phi_1 \vee \bigcirc\Diamond\phi_1$

$$\mathfrak{I}, i \models \mathfrak{a}(\Diamond\phi_1) \tag{11}$$
$$\Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for some } k, i \leq k \leq n \tag{12}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ or } i < n \text{ and } \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for some } k, i+1 \leq k \leq n \tag{13}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ or } i < n \text{ and } \mathfrak{I}, i+1 \models \mathfrak{a}(\Diamond\phi_1) \tag{14}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1 \vee \bigcirc\Diamond\phi_1) \tag{15}$$

(13) is equivalent to (12) because for $i < n$ in order to contain an answer the query needs to be satisfied now (at time point i) or at one future time point ($i+1 \leq k \leq n$). Since $i < n$ is true the satisfaction of future time points solely depends on the second part of the "implies"-statement. In the case of $i = n$ this is equivalent as well because $\mathfrak{I}, i \models \mathfrak{a}(\phi_1) \Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for all $k, n \leq k \leq n$ and $i < n$ is not true but $\mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for some $k, n+1 \leq k \leq n$ does not hold either, so we do not lose any answers.

4. $\Diamond^-\phi_1 \equiv \phi_1 \vee \bigcirc^-\Diamond^-\phi_1$

$$\mathfrak{I}, i \models \mathfrak{a}(\Diamond^-\phi_1) \tag{16}$$
$$\Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for some } k, 0 \leq k \leq i \tag{17}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ or } i > 0 \text{ and } \mathfrak{I}, k \models \mathfrak{a}(\phi_1) \text{ for some } k, 0 \leq k \leq i-1 \tag{18}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1) \text{ or } i > 0 \text{ and } \mathfrak{I}, i-1 \models \mathfrak{a}(\Diamond^-\phi_1) \tag{19}$$
$$\Leftrightarrow \mathfrak{I}, i \models \mathfrak{a}(\phi_1 \vee \bigcirc^-\Diamond^-\phi_1) \tag{20}$$

(18) is equivalent to (17) because for $i > 0$ in order to contain an answer the query needs to be satisfied now (at time point i) or at any past time point ($0 \leq k \leq i-1$). Since $i > 0$ is true the satisfaction of past time points solely depends on the second part of the "implies"-statement. In the case of $i = 0$ this is equivalent as well because $\mathfrak{I}, i \models \mathfrak{a}(\phi_1) \Leftrightarrow \mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for some $k, 0 \leq k \leq 0$ and $i > 0$ is not true but $\mathfrak{I}, k \models \mathfrak{a}(\phi_1)$ for some $k, 0 \leq k \leq 0-1$ does not hold either, so we do not lose any answers.

$\square$

The semantics of the four queries can be used now to extend the algorithm given in [1]. For that we need the notation of *answer terms*. We use the same simplification and assume in the following that $\mathsf{N_V}$, the set of *variables*, is finite and that answers are of the form $\mathfrak{a} : \mathsf{N_V} \to \Delta$ instead of $\mathfrak{a} : \mathsf{FVar}(\phi) \to \Delta$. $\mathsf{Ans}(\phi, \mathfrak{I}^{(n)})$ refers to a set of mappings $\mathfrak{a} : \mathsf{N_V} \to \Delta$, i.e, a subset of $\Delta^{\mathsf{N_V}}$.

Definition 0.3 (answer term cf. Definition 6.1 in [1])**.** Let $\mathsf{FSub}(\phi)$ denote the set of all subqueries of ϕ of the form $\bigcirc\psi_1, \newmoon\psi_1, \Box\psi_1, \Diamond\psi_1$ or $\psi_1 \mathsf{U} \psi_2$. For $j \geq 0$, we denote by Var^{ϕ}_j the set of all variables of the form x^{ψ}_j for $\psi \in \mathsf{FSub}(\phi)$. The set AT^i_{ϕ} of all *answer terms* for ϕ at $i \geq 0$ is the smallest set satisfying the following conditions:

- Every set $A \subseteq \Delta^{\mathsf{N_V}}$ is an answer term for ϕ at i.
- Every variable $x^{\psi}_j \in \mathsf{Var}^{\phi}_j$ with $j \leq i$ is an answer term for ϕ at i.
- If α_1 and α_2 are answer terms for ϕ at i, then so are $\alpha_1 \cap \alpha_2$ and $\alpha_1 \cup \alpha_2$.

The functions $\mathsf{eval}^n : \mathsf{AT}^n_{\phi} \to 2^{\Delta^{\mathsf{N_V}}}, n \geq 0$ in [1] have then to be extended as follows:

α	$\mathsf{eval}^n(\alpha)$
$x^{\Box\psi_1}_j$ with $j < n$	$\mathsf{Ans}(\Box\psi_1, \mathfrak{I}^{(n)}, j+1)$
$x^{\Diamond\psi_1}_j$ with $j < n$	$\mathsf{Ans}(\Diamond\psi_1, \mathfrak{I}^{(n)}, j+1)$
$x^{\Box\psi_1}_n$	$\Delta^{\mathsf{N_V}}$
$x^{\Diamond\psi_1}_n$	$\emptyset$

Table 2: $\mathsf{eval}^n(\alpha)$

The function $\Phi_0(\psi) : \mathsf{Sub}(\phi) \to \mathsf{AT}^0_{\phi}$ in [1] has to be expanded as follows:

ψ	$\Phi_0(\psi)$
$\Box\psi_1$	$\Phi_0(\psi_1) \cap x^{\Box\psi_1}_0$
$\Box^-\psi_1$	$\Phi_0(\psi_1)$
$\Diamond\psi_1$	$\Phi_0(\psi_1) \cup x^{\Diamond\psi_1}_0$
$\Diamond^-\psi_1$	$\Phi_0(\psi_1)$

Table 3: $\Phi_0(\psi)$

The function $\Phi^0_i(\psi) : \mathsf{Sub}(\phi) \to \mathsf{AT}^i_{\phi},\ i > 0$ in [1] has to be extended as follows:

ψ	$\Phi_i^0(\psi)$
$\Box\psi_1$	$\Phi_i^0(\psi_1) \cap x_i^{\Box\psi_1}$
$\Box^-\psi_1$	$\Phi_i^0(\psi_1) \cap \Phi_{i-1}(\Box^-\psi_1)$
$\Diamond\psi_1$	$\Phi_i^0(\psi_1) \cup x_i^{\Diamond\psi_1}$
$\Diamond^-\psi_1$	$\Phi_i^0(\psi_1) \cup \Phi_{i-1}(\Diamond^-\psi_1)$

Table 4: $\Phi_i^0(\psi)$

$\mathsf{Sub}(\phi)$ denotes the set of all TQs occurring as temporal subqueries in ϕ (including ϕ itself).

To prove that correctness and boundedness of the algorithm is preserved we will add the necessary cases to the corresponding proofs from [1].

Lemma 0.4 (cf. Lemma 6.3 in [1])**.** *The function Φ_0 is correct for 0.*

Proof. We show by induction on the structure of the subqueries $\psi \in \mathsf{Sub}(\phi)$ that $\mathsf{eval}^n(\Phi_0(\psi))$ is equal to $\mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, 0)$ for all $n \geq 0$.
If $\psi = \Box^-\psi_1$ or $\psi = \Diamond^-\psi_1$ then

$$\mathsf{eval}^n(\Phi_0(\psi)) = \mathsf{eval}^n(\Phi_0(\psi_1)).$$

This is by induction equal to $Ans(\psi_1, \mathfrak{I}^{(n)}, 0)$ which then is, as shown in Proposition 0.2, equal to $\mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, 0)$.
If $\psi = \Box\psi_1$, then

$$\begin{aligned}\mathsf{eval}^n(\Phi_0(\psi)) &= \mathsf{eval}^n(\Phi_0(\psi_1)) \cap \mathsf{eval}^n(x_0^\psi)\\ &= \mathsf{Ans}(\psi_1, \mathfrak{I}^{(n)}, 0) \cap \left\{ \begin{array}{ll} \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, 1) & \text{if } n > 0 \\ \Delta^{\mathsf{N_V}} & \text{if } n = 0 \end{array} \right\}\\ &= \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, 0)\end{aligned}$$

If $\psi = \Diamond\psi_1$, then

$$\begin{aligned}\mathsf{eval}^n(\Phi_0(\psi)) &= \mathsf{eval}^n(\Phi_0(\psi_1)) \cup \mathsf{eval}^n(x_0^\psi)\\ &= \mathsf{Ans}(\psi_1, \mathfrak{I}^{(n)}, 0) \cup \left\{ \begin{array}{ll} \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, 1) & \text{if } n > 0 \\ \emptyset & \text{if } n = 0 \end{array} \right\}\\ &= \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, 0)\end{aligned}$$

□

Lemma 0.5 (cf. Lemma 6.4 in [1])**.** *If Φ_{i-1} is correct for i-1, then Φ_i^0 is correct for i.*

Proof. We show by induction on the structure of the subqueries $\psi \in \mathsf{Sub}(\phi)$ that $\mathsf{eval}^n(\Phi_i^0(\psi))$ is equal to $\mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i)$ for all $n \geq i$.
If $\psi = \Box^- \psi_1$, then

$$\begin{aligned}\mathsf{eval}^n(\Phi_i^0(\psi)) &= \mathsf{eval}^n(\Phi_i^0(\psi_1)) \cap \mathsf{eval}^n(\Phi_{i-1}(\psi)) \\ &= \mathsf{Ans}(\psi_1, \mathfrak{I}^{(n)}, i) \cap \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i-1) \\ &= \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i)\end{aligned}$$

If $\psi = \Diamond^- \psi_1$, then

$$\begin{aligned}\mathsf{eval}^n(\Phi_i^0(\psi)) &= \mathsf{eval}^n(\Phi_i^0(\psi_1)) \cup \mathsf{eval}^n(\Phi_{i-1}(\psi)) \\ &= \mathsf{Ans}(\psi_1, \mathfrak{I}^{(n)}, i) \cup \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i-1) \\ &= \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i)\end{aligned}$$

If $\psi = \Box \psi_1$, then

$$\begin{aligned}\mathsf{eval}^n(\Phi_i^0(\psi)) &= \mathsf{eval}^n(\Phi_i^0(\psi_1)) \cap \mathsf{eval}^n(x_i^\psi) \\ &= \mathsf{Ans}(\psi_1, \mathfrak{I}^{(n)}, i) \cap \left\{ \begin{array}{ll} \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i+1) & \text{if } n > i \\ \Delta^{\mathsf{N_V}} & \text{if } n = i \end{array} \right\} \\ &= \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i)\end{aligned}$$

If $\psi = \Diamond \psi_1$, then

$$\begin{aligned}\mathsf{eval}^n(\Phi_i^0(\psi)) &= \mathsf{eval}^n(\Phi_i^0(\psi_1)) \cup \mathsf{eval}^n(x_i^\psi) \\ &= \mathsf{Ans}(\psi_1, \mathfrak{I}^{(n)}, i) \cup \left\{ \begin{array}{ll} \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i+1) & \text{if } n > i \\ \emptyset & \text{if } n = i \end{array} \right\} \\ &= \mathsf{Ans}(\psi, \mathfrak{I}^{(n)}, i)\end{aligned}$$

$\square$

Lemma 0.6 (cf. Lemma 6.5 in [1])**.** *If Φ_{i-1} is correct for i-1 and (i-1)-bounded, then we can construct a function $\Phi_i : \mathsf{Sub}(\phi) \rightarrow \mathsf{AT}_\phi^i$ that is correct for i and i-bounded.*

Proof. We need to expand the introduced function $\mathsf{update}(x_{i-1}^{\psi^j})$ before then showing for all $n \geq i$ that $\mathsf{eval}^n(x_{i-1}^{\psi^j})$ is still equal to $\mathsf{eval}^n(\mathsf{update}(x_{i-1}^{\psi^j}))$. After considering the new operators $\mathsf{update}(x_{i-1}^{\psi^j})$ looks like this:

$$\mathsf{update}(x_{i-1}^{\psi^j}) := \left\{ \begin{array}{ll} \Phi_i^{j-1}(\psi_1) & \text{if } \psi^j = \bigcirc \psi_1 \text{ or } \psi^j = \bullet \psi_1 \\ \Phi_i^{j-1}(\psi^j) & \text{if } \psi^j = \psi_1 \mathsf{U} \psi_2 \text{ or } \psi^j = \Box \psi_1 \text{ or } \psi^j = \Diamond \psi_1 \end{array} \right\}$$

For $\psi^j = \Box\psi_1$ and $\psi^j = \Diamond\psi_1$, by definition $\mathsf{eval}^n(x_{i-1}^{\psi^j}) = \mathsf{Ans}(\psi^j, \mathfrak{I}^{(n)}, i)$. Since Φ_i^{j-1} is correct for i, this is the same set as $\mathsf{eval}^n(\Phi_i^{j-1}(\psi^j)) = \mathsf{eval}^n(\mathsf{update}(x_{i-1}^{\psi^j}))$.

It remains to show i-boundedness of $\Phi_i = \Phi_i^k$. In [1] this is again proven by induction on j. It therefore suffices to add the missing cases. It is enough to show that $\mathsf{update}(x_{i-1}^{\psi^j})$ contains only variables from $\mathsf{Var}_i^{\psi^j}$. If $\psi^j = \Box\psi_1$ or $\psi^j = \Diamond\psi_1$, then $\mathsf{update}(x_{i-1}^{\psi^j}) = \Phi_i^{j-1}(\psi^j)$. Since Φ_i^{j-1} differs from Φ_i^0 only in the replacement of some variables with index $i-1$

$$\Phi_i^{j-1}(\psi^j) = \Phi_i^{j-1}(\psi_1) \cap x_i^{\psi^j}$$

or

$$\Phi_i^{j-1}(\psi^j) = \Phi_i^{j-1}(\psi_1) \cup x_i^{\psi^j}, \text{ respectively.}$$

By the induction hypothesis $\Phi_i^{j-1}(\psi_1)$ contains only variables from $\mathsf{Var}_i^{\psi_1} = \mathsf{Var}_i^{\psi^j} \setminus \{x_i^{\psi^j}\}$ and $\mathsf{Var}_{i-1}^{\psi_1} \cap \{x_{i-1}^{\psi^j}, ..., x_{i-1}^{\psi^k}\}$. Since every variable $x_{i-1}^{\psi'} \in \mathsf{Var}_{i-1}^{\psi_1}$ must satisfy $\psi' \in \mathsf{FSub}(\psi_1)$ the second set $\mathsf{Var}_{i-1}^{\psi_1} \cap \{x_{i-1}^{\psi^j}, ..., x_{i-1}^{\psi^k}\}$ is empty. This follows from the total order $\psi^1 < ... < \psi^k$ on the set $\mathsf{FSub}(\phi) = \{\psi^1, ..., \psi^k\}$ presented in [1], i.e., $\psi' \in \mathsf{FSub}(\psi^j) \setminus \{\psi^j\}$, and thus $\psi' < \psi^j$. $\Box$

References

[1] Stefan Borgwardt, Marcel Lippmann, and Veronika Thost. Temporalizing rewritable query languages over knowledge bases. *Journal of Web Semantics*, 33:50–70, 2015.