# Complexity of Decision Problems for Simple Regular Expressions

Wim Martens Frank Neven Thomas Schwentick

# Main Motivation

To study the complexity of

- inclusion,
- equivalence, and
- intersection

for XML Schema Languages occurring in practice, such as

- Document Type Definitions (DTDs) and
- XML Schema Definitions (XSDs).

# Overview

- XML Schema Languages
- Reducing Problems on XML Trees to Strings
- Simple Regular Expressions
- Inclusion of Simple Regular Expressions
- Equivalence of Simple Regular Expressions
- Intersection of Simple Regular Expressions
- Conclusion

# XML Schema Languages

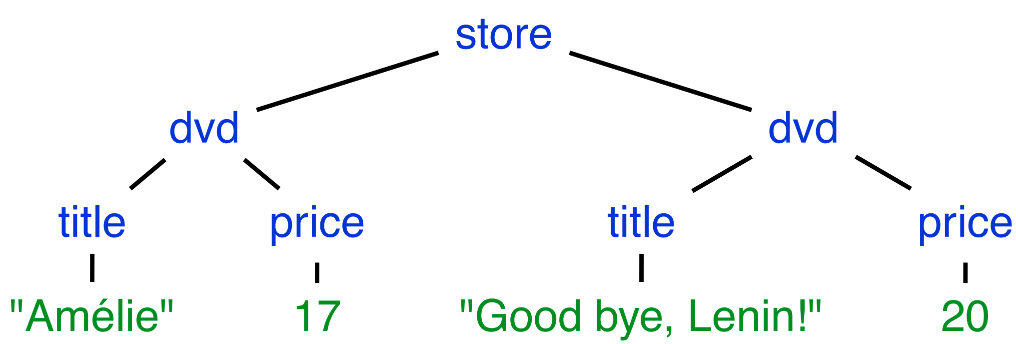
- DTDs (Document Type Definitions):

$$\texttt{store} \rightarrow \texttt{dvd dvd}^*$$
$$\texttt{dvd} \rightarrow \texttt{title price}$$

# XML Schema Languages

- DTDs (Document Type Definitions):

$$\texttt{store} \rightarrow \texttt{dvd}\ \texttt{dvd}^*$$
$$\texttt{dvd} \rightarrow \texttt{title}\ \texttt{price}$$



# XML Schema Languages

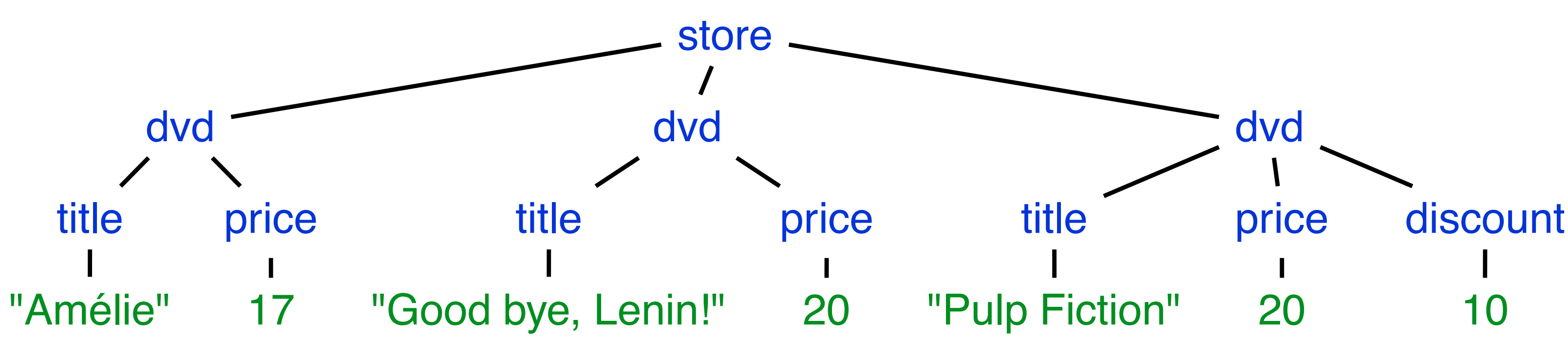
- SDTDs (Specialized DTDs):
  $\equiv$ tree automata on unranked trees

$$
\begin{array}{lcl}
\texttt{store} & \rightarrow & (\texttt{dvd}^1)^* \ \texttt{dvd}^2 \ (\texttt{dvd}^2)^* \\
\texttt{dvd}^1 & \rightarrow & \texttt{title price} \\
\texttt{dvd}^2 & \rightarrow & \texttt{title price discount}
\end{array}
$$

# XML Schema Languages

- SDTDs (Specialized DTDs):
  $\equiv$ tree automata on unranked trees

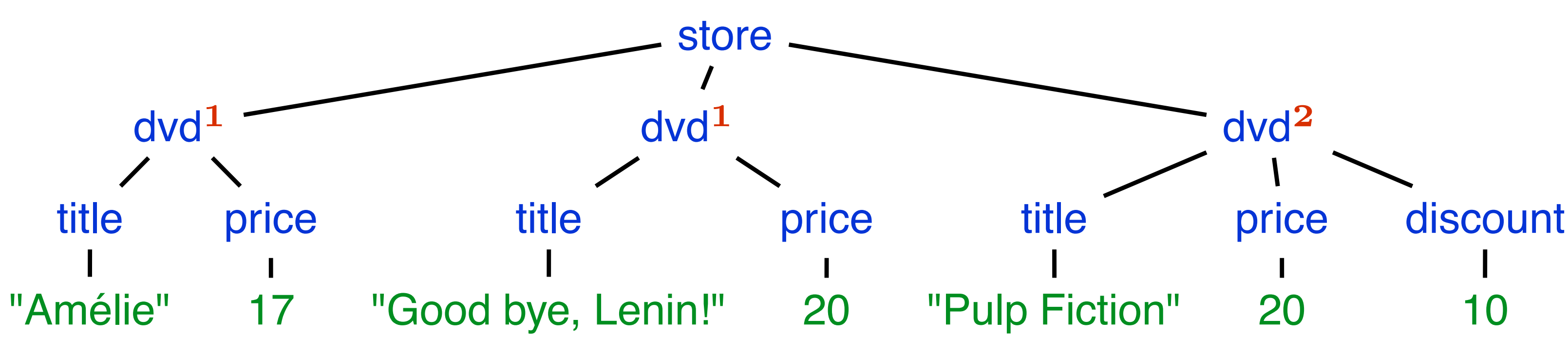
$$
\begin{array}{lcl}
\texttt{store} & \rightarrow & (\texttt{dvd}^1)^* \ \texttt{dvd}^2 \ (\texttt{dvd}^2)^* \\
\texttt{dvd}^1 & \rightarrow & \texttt{title price} \\
\texttt{dvd}^2 & \rightarrow & \texttt{title price discount}
\end{array}
$$



# XML Schema Languages

- SDTDs (Specialized DTDs):
  $\equiv$ tree automata on unranked trees

$$\begin{array}{lll} \texttt{store} & \rightarrow & (\texttt{dvd}^1)^* \ \texttt{dvd}^2 \ (\texttt{dvd}^2)^* \\ \texttt{dvd}^1 & \rightarrow & \texttt{title price} \\ \texttt{dvd}^2 & \rightarrow & \texttt{title price discount} \end{array}$$



# XML Schema Languages

- Single-type SDTDs: different types for one label in one rhs not allowed!

| | | |
|---|---|---|
| Example: | $\texttt{store} \rightarrow (\texttt{dvd}^{1})^{*}\ \texttt{dvd}^{2}\ (\texttt{dvd}^{2})^{*}$ | not allowed |
| | $\texttt{dvd}^{1} \rightarrow \texttt{title}^{2}\ \texttt{price}^{3}$ | is allowed |

$$
\begin{array}{lcl}
\texttt{store} & \rightarrow & \texttt{regulars}^{*}\ \texttt{discounts}\ \texttt{discounts}^{*} \\
\texttt{regulars} & \rightarrow & \texttt{dvd}^{1} \\
\texttt{discounts} & \rightarrow & \texttt{dvd}^{2} \\
\texttt{dvd}^{1} & \rightarrow & \texttt{title price} \\
\texttt{dvd}^{2} & \rightarrow & \texttt{title price discount}
\end{array}
$$

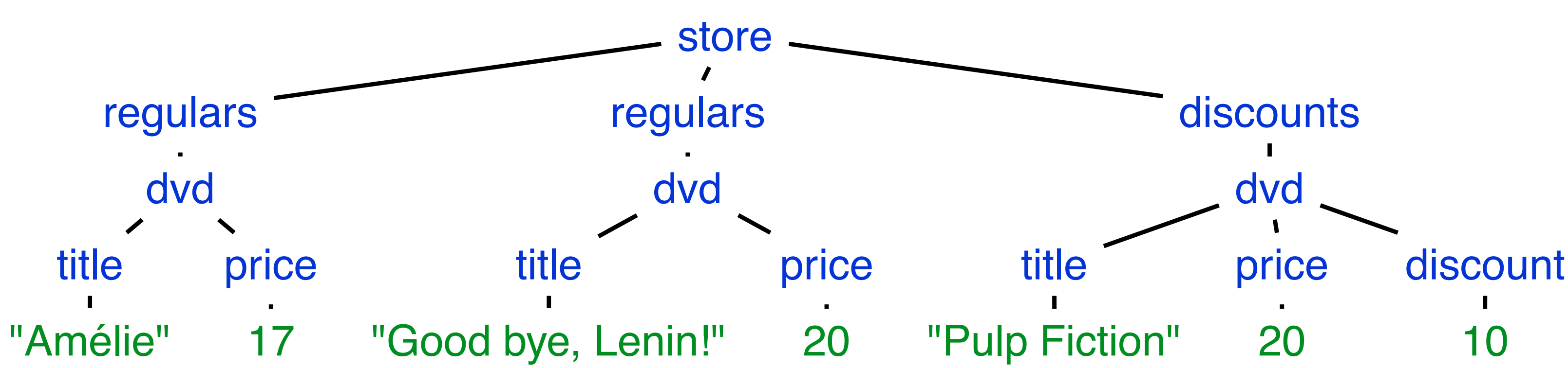
# XML Schema Languages

- Single-type SDTDs: different types for one label in one rhs not allowed!

Example:

| | |
|---|---|
| $\texttt{store} \rightarrow (\texttt{dvd}^{1})^{*}\ \texttt{dvd}^{2}\ (\texttt{dvd}^{2})^{*}$ | not allowed |
| $\texttt{dvd}^{1} \rightarrow \texttt{title}^{2}\ \texttt{price}^{3}$ | is allowed |

$$
\begin{array}{lcl}
\texttt{store} & \rightarrow & \texttt{regulars}^{*}\ \texttt{discounts}\ \texttt{discounts}^{*} \\
\texttt{regulars} & \rightarrow & \texttt{dvd}^{1} \\
\texttt{discounts} & \rightarrow & \texttt{dvd}^{2} \\
\texttt{dvd}^{1} & \rightarrow & \texttt{title price} \\
\texttt{dvd}^{2} & \rightarrow & \texttt{title price discount}
\end{array}
$$

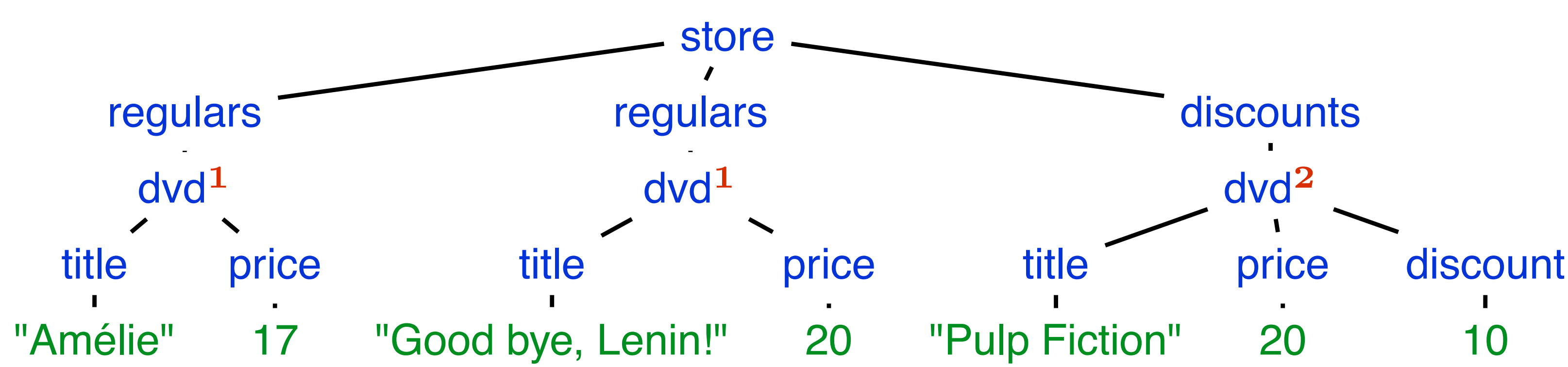


# XML Schema Languages

- Single-type SDTDs: different types for one label in one rhs not allowed!

Example: $\texttt{store} \rightarrow (\texttt{dvd}^1)^* \; \texttt{dvd}^2 \; (\texttt{dvd}^2)^*$ not allowed
$\texttt{dvd}^1 \rightarrow \texttt{title}^2 \; \texttt{price}^3$ is allowed

$$
\begin{array}{lcl}
\texttt{store} & \rightarrow & \texttt{regulars}^* \; \texttt{discounts} \; \texttt{discounts}^* \\
\texttt{regulars} & \rightarrow & \texttt{dvd}^1 \\
\texttt{discounts} & \rightarrow & \texttt{dvd}^2 \\
\texttt{dvd}^1 & \rightarrow & \texttt{title price} \\
\texttt{dvd}^2 & \rightarrow & \texttt{title price discount}
\end{array}
$$




Note: DTD $\subsetneq$ single-type SDTD $\subsetneq$ SDTD

# Decision Problems

Let $\mathcal{M}$ be a subclass of the class of DTDs or SDTDs

- The inclusion problem for $\mathcal{M}$ asks for two given schemas $d, d' \in \mathcal{M}$, whether $L(d) \subseteq L(d')$.
- The equivalence problem for $\mathcal{M}$ asks for two given schemas $d, d' \in \mathcal{M}$, whether $L(d) = L(d')$.
- The intersection problem for $\mathcal{M}$ asks for an arbitrary number of schemas $d_1, \ldots, d_n \in \mathcal{M}$, whether $\bigcap_{i=1}^{n} L(d_i) \neq \emptyset$.

Application: lower and upper bounds for type checking

# Decision Problems: General Complexity

XML Schema Definitions (XSDs) usually modelled as Specialized DTDs (or Tree Automata)

| | DTD | SDTD |
|---|---|---|
| inclusion | **PSPACE**-complete | **EXPTIME**-complete |
| equivalence | **PSPACE**-complete | **EXPTIME**-complete |
| intersection | **PSPACE**-complete | **EXPTIME**-complete |

DTDs: Involved regular expressions

[Murata,Lee,Mani 2001]: XSDs are single-type SDTDs!

# Overview

- XML Schema Languages
- Reducing Problems on XML Trees to Strings
- Simple Regular Expressions
- Inclusion of Simple Regular Expressions
- Equivalence of Simple Regular Expressions
- Intersection of Simple Regular Expressions
- Conclusion

# A Toolbox: From XML trees to strings

$\mathcal{R}$: a class of regular expressions

Notation:

- DTD($\mathcal{R}$): DTDs with regular expressions in $\mathcal{R}$
- single-type DTD($\mathcal{R}$): single-type DTDs with regular expressions in $\mathcal{R}$

# A Toolbox: From XML trees to strings

$\mathcal{R}$: a class of regular expressions
$\mathcal{C}$: a complexity class containing **PTIME**

**THEOREM:** Then the following are equivalent:

- The containment problem for $\mathcal{R}$ expressions is in $\mathcal{C}$.
- The containment problem for DTD($\mathcal{R}$) is in $\mathcal{C}$.
- The containment problem for single-type SDTD($\mathcal{R}$) is in $\mathcal{C}$.

The corresponding statement holds for the equivalence problem.

The above does not hold for SDTDs

# A Toolbox: From XML trees to strings

$\mathcal{R}$: a class of regular expressions
$\mathcal{C}$: a complexity class containing **PTIME**

**THEOREM:** Then the following are equivalent:

- The intersection problem for $\mathcal{R}$ expressions is in $\mathcal{C}$.
- The intersection problem for DTD($\mathcal{R}$) is in $\mathcal{C}$.

**THEOREM:** There is class of regular expressions $\mathcal{R}$ such that:

- The intersection problem for single-type SDTD($\mathcal{R}$) is **EXPTIME**-complete.
- The intersection problem for $\mathcal{R}$ is **NP**-complete.

# Overview

- XML Schema Languages
- Reducing Problems on XML Trees to Strings
- Simple Regular Expressions
- Inclusion of Simple Regular Expressions
- Equivalence of Simple Regular Expressions
- Intersection of Simple Regular Expressions
- Conclusion

# Simple Regular Expressions

- A base symbol is a regular expression $w$, $w?$, or $w^*$ where $w$ is a non-empty string;
- A factor is of the form $e$, $e?$, or $e^*$ where $e$ is a disjunction of base symbols.
- A simple regular expression is $\varepsilon$, $\emptyset$, or a sequence $f_1 \cdots f_k$ of factors.

| Factor | Abbr. | Factor | Abbr. | Factor | Abbr. |
|---|---|---|---|---|---|
| $a$ | $a$ | $(a_1 + \cdots + a_n)$ | $(+a)$ | $(w_1 + \cdots + w_n)$ | $(+w)$ |
| $a?$ | $a?$ | $(a_1 + \cdots + a_n)?$ | $(+a)?$ | $(w_1 + \cdots + w_n)?$ | $(+w)?$ |
| $a^*$ | $a^*$ | $(a_1 + \cdots + a_n)^*$ | $(+a)^*$ | $(w_1 + \cdots + w_n)^*$ | $(+w)^*$ |
| $w?$ | $w?$ | $(a_1^* + \cdots + a_n^*)$ | $(+a^*)$ | $(w_1^* + \cdots + w_n^*)$ | $(+w^*)$ |
| $w^*$ | $w^*$ | | | | |

# Simple Regular Expressions

- A base symbol is a regular expression $w$, $w?$, or $w^*$ where $w$ is a non-empty string;
- A factor is of the form $e$, $e?$, or $e^*$ where $e$ is a disjunction of base symbols.
- A simple regular expression is $\varepsilon$, $\emptyset$, or a sequence $f_1 \cdots f_k$ of factors.

[Bex,Neven,Van den Bussche 2004]: $> 90\%$ of expressions in practical DTDs or XSDs are simple regular expressions

# Simple Regular Expressions: Examples

| Factor | Abbr. | Factor | Abbr. | Factor | Abbr. |
|---|---|---|---|---|---|
| $a$ | $a$ | $(a_1 + \cdots + a_n)$ | $(+a)$ | $(w_1 + \cdots + w_n)$ | $(+w)$ |
| $a?$ | $a?$ | $(a_1 + \cdots + a_n)?$ | $(+a)?$ | $(w_1 + \cdots + w_n)?$ | $(+w)?$ |
| $a^*$ | $a^*$ | $(a_1 + \cdots + a_n)^*$ | $(+a)^*$ | $(w_1 + \cdots + w_n)^*$ | $(+w)^*$ |
| $w?$ | $w?$ | $(a_1^* + \cdots + a_n^*)$ | $(+a^*)$ | $(w_1^* + \cdots + w_n^*)$ | $(+w^*)$ |
| $w^*$ | $w^*$ | | | | |

$((abc)^* + b^*)(a+b)?(ab)^*(ac+b)^*$ OK

$a^*((abc)^* + c^*)^*$ OK

$(ac + (abc)^*)$ NOK

$(ab^*c)^*$ NOK

# Related Work on Strings

- [Stockmeyer, Meyer, STOC 1973]
- [Hunt III, Rosenkrantz, Szymanski, JCSS 1976]
- [Kozen, FOCS 1977]

Interesting complexity results on fragments of regular expressions.

These fragments are more general than Simple Regular Expressions.

# Overview

- XML Schema Languages
- Reducing Problems on XML Trees to Strings
- Simple Regular Expressions
- Inclusion of Simple Regular Expressions
- Equivalence of Simple Regular Expressions
- Intersection of Simple Regular Expressions
- Conclusion

# Inclusion

**THEOREM:** The inclusion problem

- is **CONP**-hard for $RE(a, a^*)$ and $RE(a, a?)$;
- is in **CONP** for $RE(\text{All} - \{(+a)^*, (+w)^*\})$;
- is **PSPACE**-hard for $RE(a, (+a)^*)$;
- is in **PSPACE** for $RE(\text{All})$; and,
- is in **PTIME** for $RE^{\leq k}$.

[Abdullah et al. 1998]: inclusion of $RE(a?, (+a)^*)$ can be solved in linear time

[Milo, Suciu 1999]: inclusion for $RE(a, \Sigma, \Sigma^*)$ is in **PTIME**

# Inclusion

Hint: **CONP**-hardness for $RE(a, a^*)$ and $RE(a, a?)$
Reduction from VALIDITY:

$(x_1 \wedge \neg x_2 \wedge x_3) \vee (\neg x_1 \wedge x_3 \wedge \neg x_4)$ reduces to testing

$$\#a|a|a|a\# \quad a?a?|a?a?|a?a?|a?a? \quad \#a|a|a|a\#$$

$$\subseteq$$

$$\#?a?|?a?|?a?|?a?\#?$$
$$aa?|a?|aa?|a?a?\#a?|a?a?|aa?|a?$$
$$\#?a?|?a?|?a?|?a?\#?$$

Intuition: $\varepsilon \equiv false$, $aa \equiv true$

# Overview

- XML Schema Languages
- Reducing Problems on XML Trees to Strings
- Simple Regular Expressions
- Inclusion of Simple Regular Expressions
- Equivalence of Simple Regular Expressions
- Intersection of Simple Regular Expressions
- Conclusion

# Equivalence

**THEOREM:** The equivalence problem is in **PTIME** for $RE(a, a?)$, and $RE(a, a^*)$.

Idea: equivalent expressions have identical normal form

Not trivial!
Example: $a^+ b^* a^* b^+ a^+$ and $a^+ b^+ a^* b^* a^+$

# Overview

- XML Schema Languages
- Reducing Problems on XML Trees to Strings
- Simple Regular Expressions
- Inclusion of Simple Regular Expressions
- Equivalence of Simple Regular Expressions
- Intersection of Simple Regular Expressions
- Conclusion

# Intersection

**THEOREM:** The intersection problem is

- **NP**-hard for $RE(a, a^*)$ and $RE(a, a?)$;
- in **NP** for $RE(\text{All} - (+w)^*)$;
- **PSPACE**-hard for $RE^{\leq 3}$; and
- in **PTIME** for $RE(a, a^+)$.

# Overview

- XML Schema Languages
- Reducing Problems on XML Trees to Strings
- Simple Regular Expressions
- Inclusion of Simple Regular Expressions
- Equivalence of Simple Regular Expressions
- Intersection of Simple Regular Expressions
- Conclusion

# Conclusion

- DTDs, XML Schema Definitions:
  - Inclusion, equivalence: complexity carries over from string case
  - Intersection: complexity only carries over to DTDs
- Simple Regular Expressions:
  - Inclusion, intersection: hard surprisingly quickly
  - Equivalence: seems easier than inclusion
- One unambiguous regular expressions:
  - Inclusion, equivalence: **PTIME** (DFA)
  - Intersection: **PSPACE**-hard

# Overview

| RE-fragment | Inclusion | Equivalence | Intersection |
|---|---|---|---|
| $a, a^+$ | in **PTIME** (DFA!) | in **PTIME** | in **PTIME** |
| $a, a^*$ | **CONP**-complete | in **PTIME** | **NP**-complete |
| $a, a?$ | **CONP**-complete | in **PTIME** | **NP**-complete |
| All $-\{(+a)^*, (+w)^*\}$ | **CONP**-complete | in **CONP** | **NP**-complete |
| $a, (+a)^*$ | **PSPACE**-complete | in **PSPACE** | **NP**-complete |
| All $-\{(+w)^*\}$ | **PSPACE**-complete | in **PSPACE** | **NP**-complete |
| All | **PSPACE**-complete | in **PSPACE** | in **PSPACE** |
| RE$^{\leq k}$ $(k \geq 3)$ | in **PTIME** | in **PTIME** | **PSPACE**-complete |
| one-unambiguous | in **PTIME** | in **PTIME** | **PSPACE**-complete |