I N F O R M A T I K

Subsumption of concepts in DL $\mathcal{FL}_0$ for (cyclic) terminologies with respect to descriptive semantics is PSPACE-complete.

Yevgeny Kazakov and Hans de Nivelle

MPI–I–2003–2–003 April 2003

FORSCHUNGSBERICHT RESEARCH REPORT

M A X - P L A N C K - I N S T I T U T
F Ü R
I N F O R M A T I K

Stuhlsatzenhausweg 85 66123 Saarbrücken Germany

Author's Address

Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken
Germany
Phone: +49 681 9325-215, +49 681 9325-223
Fax: +49 681 9325-299
Email: `ykazakov,nivelle@mpi-sb.mpg.de`

Acknowledgements

The authors would like to thank Franz Baader for reading the draft of the paper and giving important remarks.

Abstract

We prove the PSPACE-completeness of the subsumption problem for (cyclic) terminologies with respect to descriptive semantics in a simple Description Logic $\mathcal{FL}_0$, which allows for conjunctions and universal value restrictions only, thus solving the problem which was open for more than ten years.

Keywords

Description Logic, Automata Theory

1 Introduction

$\mathcal{FL}_0$ is a Description Logic where concepts can be constructed by using conjunctions and universal value restrictions only. The concept subsumption problem in $\mathcal{FL}_0$ for (cyclic) terminologies was investigated in [Baa90], [Baa96] and [Neb91] for the three kinds of semantics: the least fixpoint (lfp), the greatest fixpoint (gfp) and the descriptive semantics. These papers provide a PSPACE decision procedure for the subsumption problem with respect to all three kinds of semantics. In addition, in [Baa90] and [Baa96] it was shown that this problem is PSPACE-hard both for gfp- and lfp-semantics. For the descriptive semantics, however, the highest known lower bound was found to be co-NP [Neb91], which provides a complete characterization for acyclic terminologies. So, the question about the exact complexity of the subsumption problem for the descriptive semantics with respect to (cyclic) terminologies has been open.

In this paper we prove the PSPACE-hardness of this problem (and thus, eliminate the remaining complexity gap) by reduction from the universality problem for automata on infinite words with prefix acceptance condition.

2 Description Logic $\mathcal{FL}_0$

$\mathcal{FL}_0$ is a simple Description Logic, which allows for conjunctions and universal value restrictions of concepts only. Formally, given a *signature* $\Sigma = (\mathcal{A}, \mathcal{R})$ consisting of *concept names* $\mathcal{A}$ and *role names* $\mathcal{R}$, the set of (*generalized*) *concepts* $\mathcal{C}_\Sigma$ of *DL* $\mathcal{FL}_0$ is defined by the grammar:

$$\mathcal{C}_\Sigma \ ::= \ A \mid C_1 \sqcap C_2 \mid \forall R.C$$

were $A \in \mathcal{A}$ are usually called *atomic concepts*; C_1, C_2, C are arbitrary generalized concepts of $\mathcal{FL}_0$ and $R \in \mathcal{R}$.

A *terminology* (or *TBox* for short) is a finite set of *concept definitions* of the form $A \doteq C$, where A is an atomic concept called *defined concept* and C is a generalized concept.

The *semantics* for $\mathcal{FL}_0$ is defined by means of *interpretations* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, were Δ^I is a set called the *domain* of $\mathcal{I}$ and $\cdot^\mathcal{I}$ assigns to every concept name $A \in \mathcal{A}$ a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ and to every role name $R \in \mathcal{R}$ a relation $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The interpretation $\mathcal{I}$ can be extended to generalized concepts of $\mathcal{FL}_0$ by defining:

$(C_1 \sqcap C_2)^\mathcal{I} := C_1^\mathcal{I} \cap C_2^\mathcal{I}$; $\quad (\forall R.C)^\mathcal{I} := \{d \in \Delta^\mathcal{I} \mid \forall\ e \in \Delta^\mathcal{I},\ (d, e) \in R^\mathcal{I} \text{ implies } e \in C^\mathcal{I}\}$

An interpretation $\mathcal{I}$ is a *model* of *TBox* $\mathcal{T}$ *iff* $A^\mathcal{I} = C^\mathcal{I}$ for all definitions $A \doteq B$ of $\mathcal{T}$. Given a terminology $\mathcal{T}$ we say that a concept A *is subsumed* by a concept B w.r.t. *descriptive semantics* (notation: $A \sqsubseteq_\mathcal{T} B$) *iff* $A^\mathcal{I} \subseteq B^\mathcal{I}$ for all models $\mathcal{I}$ of $\mathcal{T}$. The associated decision problem for $\mathcal{T}$, A and B is called the *concept subsumption problem*.

Since we are interested in proving the hardness result for the concept subsumption problem, we may consider restricted forms of terminologies. Thus, in the rest of the paper we assume that *TBox* contains only definitions of the form:

$$A \ \doteq \ \forall R_1.B_1 \sqcap \ldots \sqcap \forall R_l.B_l, \tag{1}$$

were A, B_i are atomic concepts ($1 \leq i \leq l$) and $l \geq 1$. We also assume that *exactly one* definition is given for *every* atomic concept.

With every terminology $\mathcal{T}$ of the form (1) we associate a *non-deterministic semi-automaton* $\mathcal{A}_{\mathcal{T}} = (\Sigma, Q, \delta)$ consisting of the *finite alphabet of letters* Σ, the finite *set of states* Q and the *transition relation* $\delta \subseteq Q \times \Sigma \times Q$. We proceed similarly as in [Baa96], [Neb91]:

- the alphabet Σ of $\mathcal{A}_{\mathcal{T}}$ is the set of role names of $\mathcal{T}$;
- the set of states Q is the set of concept names in $\mathcal{T}$ and
- the transition relation $\delta = \{(A, R, B) \mid A \doteq \ldots \sqcap \forall R.B \sqcap \ldots \in \mathcal{T}\}$.

Note that this construction gives a one-to-one correspondence between terminologies of the form (1) and semi-automata without *blocking states*: for every state $q \in Q$ there exist some $a \in \Sigma$ and $q' \in Q$ such that $(q, a, q') \in \delta$.

A *run* of a semi-automaton $\mathcal{A}$ over an (in)finite *word* $w = a_1 \cdot a_2 \cdots a_i (\cdots) \in \Sigma^{*(\omega)}$ is an (in)finite sequence of states $r: \ q_0, q_1, \ldots, q_i, (\ldots) \in Q^{*(\omega)}$ such that $(q_{i-1}, a_i, q_i) \in \delta$ for any $i \geq 1$. With every two states $q_1, \ q_2 \in Q$ of a semi-automaton $\mathcal{A} = (\Sigma, Q, \delta)$ one can associate the *regular language* $L_{\mathcal{A}}(q_1, q_2) := \{w \in \Sigma^* \mid \text{there exists a run } q_1, \ldots, q_2 \text{ over } w\}$.

Now we give the automata-theoretic characterization of the concept subsumption problem. Theorem 29 in [Baa96] provides the characterization for the general terminologies, however we may give a simplified variant for the restricted form of terminologies.

Theorem 1 (Characterization of concept subsumption) *Let $\mathcal{T}$ be a terminology of the form (1) and $\mathcal{A}_{\mathcal{T}} = (\Sigma, Q, \delta)$ be the corresponding semi-automaton. Then $A_0 \sqsubseteq_{\mathcal{T}} B_0$ iff for every word $w \in \Sigma^\omega$ and for every run*
$r_B: \ B_0, B_1, \ldots, B_i, \ldots$ in $\mathcal{A}_{\mathcal{T}}$ over w there exists a run
$r_A: \ A_0, A_1, \ldots, A_i, \ldots$ in $\mathcal{A}_{\mathcal{T}}$ over w and an integer $k \geq 0$ such that $A_k = B_k$.

Proof. We prove the theorem by inspecting the *tableau algorithm* for checking concept subsumption. We try to refute $A_0 \sqsubseteq_{\mathcal{T}} B_0$ in some model $\mathcal{I}$ of $\mathcal{T}$ with the domain $\mathbb{N}$. Every node of the tableau will describe necessary conditions of the form $n : A$, $n : \neg B$ or $(n, m) : R$ for $A, B \in \mathcal{A}$ and $R \in \mathcal{R}$, which shell be imposed on a model $\mathcal{I}$. The semantical meanings of these restrictions are $n \in A^{\mathcal{I}}$, $n \notin B^{\mathcal{I}}$ and $(n, m) \in R^{\mathcal{I}}$ respectively. We start with the node $\{0 : A_0, \ 0 : \neg B_0\}$ and apply expansion rules. Every definition

$$A \doteq \forall R_1.B_1 \sqcap \ldots \sqcap \forall R_l.B_l$$

of $\mathcal{T}$ enforces two sorts of rules:

$$(\forall^i A) \ \frac{n : A, \ (n, m) : R_i}{m : B_i}; \qquad (\exists A) \ \frac{n : \neg A}{\ldots \mid (n, n+1) : R_i, \ (n+1) : \neg B_i \mid \ldots}$$

A rule is applied to a node by forming a child of this node containing all formulas of parent and the conclusion of the rule; $(\exists A)$-rule assumes branching over $i \leq i \leq l$. The rules are applied *fairly*: the application of a rule cannot be postponed forever. Some branches of the tableau can lead to the *inconsistent* node containing a *clash* $\{n : A,\ n : \neg A\}$. In this case the branch is *closed*, otherwise it is *open*. The tableau is *closed iff* all its branches are closed. The presented tableau procedure is sound and complete for the concept subsumption problem:

Proposition 2 *The tableau for A_0, B_0 and $\mathcal{T}$ is closed iff $A_0 \sqsubseteq_{\mathcal{T}} B_0$.*

Proof. The proof of this proposition can be found in the Appendix A. □

Now, to prove the theorem, observe that for every branch τ of the tableau:

1. There is exactly one negative expression of the form $n : \neg B_n$ for every $n \geq 0$;
2. There is exactly one positive expression of the form $(m, n) : R_n$ for every $n \geq 1$, and only for $m = n - 1$.
3. The sequence $r^{\tau}_{B_0} :\ B_0, \ldots, B_i, \ldots$ is a run over the word $w^{\tau} = R_1 \cdot R_2 \cdots R_i \cdots$ in $\mathcal{A}_{\mathcal{T}}$. Additionally, every run $r :\ B'_0, \ldots, B'_i, \ldots$ corresponds to some branch of the tableau.
4. For every positive expression $m : A'$ in the branch τ either $m = 0$ and $A' = A_0$ or $m > 0$ and $(A'', R_m, A') \in \delta$ for some $A'' \in \tau$, where R_m is the m-th letter of w^{τ}.

Claims 1–4 can be proved by induction on n. Now, to conclude the result of the theorem:
$A_0 \sqsubseteq_{\mathcal{T}} B_0$
  iff *(by soundness and completeness of the tableau procedure)*
every branch τ of the tableau is closed
  iff *(by 3. and by the definition of the closed branch)*
for every run $r^{\tau}_{B_0} :\ B_0, \ldots, B_i, \ldots$ over $w \in \Sigma^{\omega}$ there is some $A'(k) = B_k(k) \in \tau$
  iff *(by 4.)*
for every run $r_{B_0} :\ B_0, \ldots, B_i, \ldots$ over $w \in \Sigma^{\omega}$
there exists a run $r_{A_0} :\ A_0, \ldots, A_k = B_k, B_{k+1}, \ldots$ over w
  iff
for every run $r_{B_0} :\ B_0, \ldots, B_i, \ldots$ over $w \in \Sigma^{\omega}$
there exists a run $r_{A_0} :\ A_0, \ldots, A_i, \ldots$ over w and $k \geq 0$ such that $A_k = B_k$. □

2.1 The reduction.

Now we consider an instance of the concept subsumption problem which suffices to prove PSPACE-hardness. Take a semi-automaton $\mathcal{A} = (\Sigma, Q, \delta)$ and two states $q_1, q_2 \in Q$. We construct a new semi-automaton $\mathcal{A}'$ from $\mathcal{A}$ by adding a new state q' and making it reachable from q_2 and itself by any transition: $\mathcal{A}' = (\Sigma, Q', \delta')$, where $Q' = Q \cup q'$ and $\delta' = \delta \cup \{(q_2, a, q'),\ (q', a, q') \mid a \in \Sigma\}$.

If $\mathcal{A}'$ does not have blocking states then we can consider the terminology T' corresponding to $\mathcal{A}'$, so q_1 corresponds to some concept A of T' and q' corresponds to some concept B of T'. By Theorem 1, B subsumes A *iff* for every run from q' over some word $w \in \Sigma^\omega$ there exists a run in $\mathcal{A}'$ from q_1 over w such that both runs share at least one state. Since every run from q' can contain the state q' only and for every $w \in \Sigma^\omega$ such a run always exists, we obtain: "B subsumes A iff for every $w \in \Sigma^\omega$ there exists a run over w from q_1 containing q'." Note that in the last sentence we can replace q' by q_2. Thus concept subsumption problem is not easier than the problem:

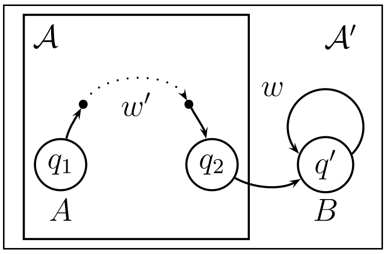

Figure 1: The reduction

> "given a semi-automaton $\mathcal{A} = (\Sigma, Q, \delta)$ and two states $q_1, q_2 \in Q$ such that all states in $Q \setminus \{q_2\}$ are not blocking, check whether any word $w \in \Sigma^\omega$ has a finite prefix $w' \in L_{\mathcal{A}}(q_1, q_2)$."

In the next section we reformulate this problem in terms of automata on infinite words as the *universality problem* and prove that it is PSPACE-hard.

3 Automata on infinite words and the universality problem

Many kinds of finite automata on infinite words (ω-automata) have been investigated in the literature (for a survey see [Tho90]). There is a classification of automata according to acceptance conditions. Büchi automata, for instance, accept an infinite word if there exists a run over this word in which some accepting state is encountered infinitely often.

Although many algorithms for automata are described in the literature, the corresponding complexity issues are usually not well-studied. The (non)universality problem: "given an automaton A check if it does (not) accept all words" is known to be PSPACE-complete for non-deterministic Büchi automata as well as for non-deterministic finite automata on finite words. We introduce a *prefix* acceptance condition for ω-automata and show that the universality problem is also PSPACE-hard for this automata. One of the implications of this result is the PSPACE-hardness of the subsumption problem for the descriptive semantics.

A *non-deterministic finite automaton* (**NFA**) is a tuple $\mathcal{A} = (\Sigma, Q, \delta, Q_0, F)$, which is a semi-automaton (Σ, Q, δ) extended with a set of *initial states* $Q_0 \subseteq Q$ and a set of *accepting states* $F \subseteq Q$. The *size* of the automaton $\mathcal{A} = (\Sigma, Q, \delta, Q_0, F)$ is $|\mathcal{A}| = |Q| + |\delta|$. We distinguish several kinds of non-deterministic finite automata according to the acceptance condition:

1. An *automaton on finite words* $\mathbf{NFA}^*$ is an $\mathbf{NFA} = (\Sigma, Q, \delta, Q_0, F)$ which accepts a finite word $w \in \Sigma^*$ *iff* there exists a run $r:\ q_1, \ldots, q_n$ over w with $q_1 \in Q_0$, $q_n \in F$.

2. A *Büchi automaton* $\mathbf{NFA}_b^\omega$ is an $\mathbf{NFA} = (\Sigma, Q, \delta, Q_0, F)$ on *infinite* words. It accepts $w \in \Sigma^\omega$ *iff* there exists a run $r:\ q_1, \ldots, q_i, \ldots$ over w which repeats some state from F *infinitely often*.

3. We introduce the ω-automaton with the *prefix acceptance condition* $\mathbf{NFA}_p^\omega$ as a $\mathbf{NFA} = (\Sigma, Q, \delta, Q_0, F)$ which accepts $w \in \Sigma^\omega$ *iff* there exist a *finite prefix* w' of w and a run $r:\ q_1, \ldots, q_n$ over w' with $q_1 \in Q_0$ and $q_n \in F$. In other words, $\mathbf{NFA}_p^\omega$ accepts an infinite word if it accepts a finite prefix of this word as $\mathbf{NFA}^*$.

In section 2.1 we have shown that a certain problem for semi-automata $\mathcal{A} = (\Sigma, Q, \delta)$ can be seen as an instance of the concept-subsumption problem and thus should be not harder. After we have introduced the automata with the prefix acceptance condition, we can reformulate this problem as: "given $\mathbf{NFA}_p^\omega = (\Sigma, Q, \delta, \{q_1\}, \{q_2\})$ without blocking states in $Q \setminus \{q_2\}$, check whether all words $w \in \Sigma^\omega$ are accepted." Such a problem appears in the literature as a *(non)universality problem* for finite automata [Var95]. The $\mathbf{NFA}^*$ ($\mathbf{NFA}_b^\omega$, $\mathbf{NFA}_p^\omega$) is *universal* iff it accepts any word $w \in \Sigma^*$ ($w \in \Sigma^\omega$). The associated decision problem is called the *universality problem*. This problem is known to be PSPACE-complete for $\mathbf{NFA}^*$ and $\mathbf{NFA}_b^\omega$ (cf. [Var95]). It is not surprising that we can obtain the similar result for the $\mathbf{NFA}_p^\omega$.

Theorem 3 *The universality problem for* $\mathbf{NFA}_p^\omega$ *is in PSPACE.*

Proof. The proof is by the reduction to the universality problem for Büchi automata. Given $\mathbf{NFA}_p^\omega$ $\mathcal{A} = (\Sigma, Q, \delta, Q_0, F)$ we proceed similarly as in the section 2.1: Consider the Büchi automaton $\mathcal{A}' = (\Sigma, Q', \delta', Q_0, \{q'\})$, where q' is a new state, $Q' = Q \cup \{q'\}$ and $\delta' = \delta \cup \{(q, a, q')\ , (q', a, q') \mid q \in F,\ a \in \Sigma\}$. $\mathcal{A}$ accepts $w \in \Sigma^\omega$ *iff* $\mathcal{A}'$ does, so $\mathcal{A}$ is universal *iff* $\mathcal{A}'$ is universal. □

Theorem 4 *The universality problem for* $\mathbf{NFA}_p^\omega$ *is PSPACE-hard.*

Proof. The proof is given by the reduction from polynomial-space Turing machines. The idea is quite standard for proving such results [??]. For every Turing machine and input we construct the automaton which accepts every word except the legal computation of the Turing machine: given some candidate word it "guesses" the position of the possible error and accepts the word if it is the error indeed. So the constructed automaton is universal *iff* the Turing machine does not accept the input. The details of the proof can be found in the Appendix B. □

Corollary 5 *The universality problem for* $\mathbf{NFA}_p^\omega$ *is PSPACE-complete.*

We have proved the PSPACE-hardness of the universality problem for $\mathbf{NFA}_p^\omega$ $\mathcal{A} = (\Sigma, Q, \delta, Q_0, F)$, however we need to prove the hardness for the instance when we have only one initial, one accepting state and do not have blocking states among the non-accepting states. The next proposition shows that we can assume these restrictions without loss of generality.

Proposition 6 *For any* $\mathbf{NFA}_p^\omega$ $\mathcal{A} = (\Sigma, Q, \delta, Q_0, F)$ *one can construct an* $\mathbf{NFA}_p^\omega$ $\mathcal{A}' = (\Sigma, Q', \delta', \{q_0'\}, \{f'\})$ *without blocking states in* $Q' \setminus \{f'\}$ *in linear size of* $|\mathcal{A}|$ *which accepts exactly the same words as* $\mathcal{A}$.

Proof. We consider two cases:

1. $Q_0 \cap F \neq \emptyset$. Then $\mathcal{A}$ trivially accepts all words and we can take say $A' := \{\Sigma, \{q\}, \emptyset, \{q\}, \{q\}\}$ for some state q.

2. $Q_0 \cap F = \emptyset$. It suffices to construct $\mathcal{A}'$ which accepts exactly the same *finite* words as $\mathcal{A}$ by the $\mathbf{NFA}^*$-acceptance condition. We simply take $\mathcal{A}' = (\Sigma, Q', \delta', \{q_0'\}, \{f'\})$ with the new states q_0' and f', and define $Q' = Q \cup \{q_0',\ f'\}$,
$$\begin{aligned}\delta' = \delta &\cup \{(q_0', a, q) \mid \exists q_0 \in Q_0 : (q_0, a, q) \in \delta\}\\ &\cup \{(q, a, f') \mid \exists f \in F : (q, a, f) \in \delta\}\\ &\cup \{(q_0', a, f') \mid \exists q_0 \in Q_0, \exists f \in F : (q, a, f) \in \delta\}.\end{aligned}$$
If some state $q' \in Q' \setminus \{f'\}$ is blocking then we can remove it together with the involved transitions since no run from q_0' to f' can contain q'. □

Corollary 7 *The concept subsumption problem for DL* $\mathcal{FL}_0$ *with cyclic terminologies w.r.t. descriptive semantics is PSPACE-complete.*

Appendix A.

In this appendix we give a proof of Proposition 2.

Proposition 2 *The tableau for* $A_0,\ B_0$ *and* $\mathcal{T}$ *is closed iff* $A_0 \sqsubseteq_{\mathcal{T}} B_0$.

Proof. To prove the *soundness* ($\Rightarrow$) note that any model $\mathcal{I}$ of $\mathcal{T}$ in which $A_0^{\mathcal{I}} \not\subseteq B_0^{\mathcal{I}}$ can guide an open branch of the tableau.

The *completeness* part ($\Leftarrow$) is more involved. Assume that S is a set of expressions on the open branch of the tableau. Consider the closure $c(S)$ of S under the rules:

$$(c^i A)\ \frac{m : \neg B_i,\ (n, m) : R_i}{n : \neg A}, \quad A \doteq \ldots \sqcap \forall R_i.B_i \sqcap \cdots \in \mathcal{T}.$$

Formally, $c(S) = \cup_{i \geq 0} S^i$, were S^i is obtained from S by adding a *finite number* of conclusions of the rules $(c^i A)$ with $n, m \leq i$. Note that if S did not contain a clash then so is $c(S)$: Otherwise clash first appears in some S^i, $i > 0$. Then consider the first application of the rule $(c^i A)$ which produces a clash $\{n : A,\ n : \neg A\}$ in S^i. Since $n : A \in S$ and S is closed under the rules $(\forall^i A)$, the clash $\{m : B_i,\ m : \neg B_i\}$ should have occurred in S^i before the (presumably first) clash $\{n : A,\ n : \neg A\}$ has appeared. A contradiction.

Since S is a set of formulas of the open branch, we have proved that $c(S)$ does not contain a clash.

The set $c(S)$ defines a model $\mathcal{I} = (\mathbb{N}, \cdot^{\mathcal{I}})$ were:

- $(n,m) \in R^{\mathcal{I}}$ *iff* $(n,m) : R \in c(S)$ (*iff* $(n,m) : R \in S$), $R \in \mathcal{R}$;
- $n \in A^{\mathcal{I}}$ *iff* $n : \neg A \notin c(S)$, $A \in \mathcal{A}$.

$\mathcal{I}$ is indeed a model of $\mathcal{T}$ in which $A_0^{\mathcal{I}} \not\subseteq B_0^{\mathcal{I}}$:

1. $0 \in A_0^{\mathcal{I}}$ since $0 : A_0 \in S \subseteq c(S)$, thus $0 : \neg A_0 \notin c(S)$ ($c(S)$ is clash-free);
2. $0 \notin B_0^{\mathcal{I}}$ since $0 : \neg B_0 \in S \subseteq c(S)$;
3. $A^{\mathcal{I}} \subseteq (\forall R_1.B_1 \sqcap \ldots \sqcap \forall R_l.B_l)^{\mathcal{I}}$ because $c(S)$ is closed under the rules $(\forall^i A)$;
4. $A^{\mathcal{I}} \supseteq (\forall R_1.B_1 \sqcap \ldots \sqcap \forall R_l.B_l)^{\mathcal{I}}$: $n \notin A^{\mathcal{I}}$ *iff* $n : \neg A \in c(S)$ *iff* $n : \neg A \in S$ or $n : \neg A$ is obtained by some $(c^i A)$. In the first case the inclusion holds by the rule $(\exists A)$; In the last case there are some $(n,m) : R_i \in c(S)$, $m : \neg B_i$, which make the right-hand side not to contain n.

Note that we have also proved that the concept subsumption $A_0 \sqsubseteq_{\mathcal{T}} B_0$ has the *linear model property*, i.e. we may consider only *tree-models* $\mathcal{I}$ of $\mathcal{T}$ with branching degree 1. □

Appendix B.

In this appendix we give a proof of Theorem 4.

Theorem 4 *The universality problem for* $\mathbf{NFA}_p^\omega$ *is PSPACE-hard.*

Proof. We prove the theorem by the reduction from polynomial-space Turing machines using the definition:

PSPACE = { L | L *is a language decided by a deterministic Turing machine in polynomial space* }

The details of involved definitions can be found, for instance, in [Sip97], however in order to be self-contained, we give the ones that are needed.

A *Turing machine* is a tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, were Q is the finite *set of states*, Σ is the finite *input alphabet*, Γ is the finite *tape alphabet* containing the special *blank symbol* $\sqcup$ ($\Sigma \subseteq \Gamma \setminus \{\sqcup\}$), $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the *transition function*, $q_0 \in Q$ is the *initial state*, $q_{accept} \in Q$ is the accepting state and $q_{reject} \in Q$ ($q_{reject} \neq q_{accept}$) is the rejecting state.

A *configuration* of the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ is a string of the form: $c = a_1 a_2 \ldots a_{i-1} q a_i \ldots a_k$, were each $a_j \in \Gamma$, $q \in Q$. One could think of the configuration c as the description of the Turing machine in the state q with the head at the i-th cell of the tape with the content $a_1 \cdots a_k$.

The transition function δ can be extended to configurations in the following way: Let $a, b \in \Gamma$, $u, v \in \Gamma^*$ and $[c]$ denote the cut of the configuration c by removing the rightmost blank symbols $\sqcup$ from c. Then

$$\hat{\delta}(uaq_ibv) := \begin{cases} uq_jacv & \text{if } \delta(q_i,b) = (q_j,c,L); \\ uacq_jv & \text{if } \delta(q_i,b) = (q_j,c,R); \end{cases} \quad \hat{\delta}(q_ibv) := \begin{cases} q_jcv & \text{if } \delta(q_i,b) = (q_j,c,L); \\ cq_jv & \text{if } \delta(q_i,b) = (q_j,c,R); \end{cases}$$

$\hat{\delta}(uq_i) := [\hat{\delta}(uq_i\sqcup)]$;

A *computation* of the Turing machine $M = (Q,\Sigma,\Gamma,\delta,q_0,q_{accept},q_{reject})$ *from* $x \in \Sigma^*$ is a sequence of configurations $c_0, c_1, \ldots, c_i, \ldots$ such that $c_0 = q_0x$ and $c_{i+1} = \hat{\delta}(c_i)$. If the computation ends with a configuration c_n then if $q_{accept} \in c_n$, we say that M *accepts* x; if $q_{reject} \in c_n$, we say that M *rejects* x.

The Turing machine M *decides* the language $L \subseteq \Sigma^*$ if for every $x \in \Sigma^*$, $x \in L$ implies M accepts x, and $x \notin L$ implies M rejects x.

We say that M is a *polynomial-space* Turing machine if there exists a polynomial $p(n)$ such that for any input $x \in \Sigma^*$ and the computation $c_0, c_1, \ldots, c_i, \ldots$ from x the length of every configuration $|c_i| \leq p(|x|)$.

Now we give a polynomial-time reduction from the decision problem for any language $L \in$ PSPACE to the universality problem for some set of $\mathbf{NFA}_p^\omega$.

Assume $M = (Q,\Sigma,\Gamma,q_0,q_{accept},q_{reject})$ is a polynomial-space Turing machine that decides L. We give an algorithm which for every $x \in \Sigma^*$ constructs a $\mathbf{NFA}_p^\omega$ $\mathcal{A}_x$ in polynomial size of $|x|$ such that $\mathcal{A}_x$ accepts all words, except the word:

$$w_0 = \#\cdot\#\cdot c_0\cdot(\sqcup)^{l_0}\cdot\#\cdot c_1\cdot(\sqcup)^{l_1}\cdot\#\cdots\#\cdot c_k\cdot(\sqcup)^{l_k}\cdot\#\cdot c_k\cdot(\sqcup)^{l_k}\cdots$$

were $c_0, c_1, \ldots, c_k$ is an accepting computation for x (if any); $l_i = l - |c_i|$, were $l = p(|x|)$ for polynomial $p(n)$ bounding the size of configurations of M; $\#$ is a new symbol ($\# \notin \Gamma$). Thus, $x \in \overline{L}$ *iff* M rejects x *iff* M does not accept x *iff* A_x is universal, and we can obtain the reduction since PSPACE = co-PSPACE.

Consider the word w_0. Note that every three subsequent symbols $\sigma_{i-1}, \sigma_i, \sigma_{i+1}$ at the positions $i-1, i, i+1$ of w_0 uniquely determine the symbol σ_{i+l+1} at the position $i+l+1$ of w_0. To be precise, $\sigma_{i+l+1} = Next(\sigma_{i-1}, \sigma_i, \sigma_{i+1})$, were:

$Next(q_i,a,\sigma_1) = c$, $Next(b,q_i,a) = b$, $Next(\#,q_i,a) = q_j$, $Next(\sigma_1,b,q_i) = q_j$
if $\delta(q_i,a) = (q_j,c,L)$, $q_i \neq q_{accept}$;

$Next(q_i,a,\sigma_1) = q_j$, $Next(\sigma_1,q_i,a) = c$, $Next(\sigma_1,b,q_i) = b$
if $\delta(q_i,a) = (q_j,c,R)$, $q_i \neq q_{accept}$;

$Next(\sigma_1,\sigma_2,\sigma_3) = \sigma_2$ in all other cases;
$(a,b,c \in \Gamma,\ \sigma_i \in \Gamma \cup \{\#\})$.

The informal description of $\mathcal{A}_x$ is as follows: given an infinite string $w \in (\Gamma \cup \{\#\})^\omega$, $\mathcal{A}_x$ accepts w if it can find that $w \neq w_0$, which can be done by detecting one of the following:

1. First $l+2$ symbols of w differ from those of $\#\#x(\sqcup)^{l_0}$ ($l_0 = l - |x|$);
2. For some $i \geq 2$ the symbol $\sigma_{i+l+1} \neq Next(\sigma_{i-1}, \sigma_i, \sigma_{i+1})$;
3. The string w contains the symbol q_{reject}.

Note that since M decides the language $L \subseteq \Sigma^*$, every computation $c_0, c_1, \ldots, c_i, \ldots$ from the $x \in \Sigma^*$ should end either with the accepting state q_{accept} or with the rejecting state q_{reject}. So, $w \neq w_0$ *iff* w satisfies one of the 1-3 above.

Formally, $\mathcal{A}_x = \mathcal{A}_x^1 \cup \mathcal{A}_x^2 \cup \mathcal{A}_x^3$ were: $\mathcal{A}_x^i$ is the $\mathbf{NFA}_p^\omega$ over $(\Gamma \cup \{\#\})^w$ which accepts a word w if the corresponding condition i above is fulfilled ($i = 1, 2, 3$). The *union* of two automata $\mathcal{A}^1 = (\Sigma, Q^1, \delta^1, Q_0^1, F^1)$ and $\mathcal{A}^2 = (\Sigma, Q^2, \delta^2, Q_0^2, F^2)$ is the automaton $\mathcal{A} = (\Sigma, Q^1 \cup Q^2, \delta^1 \cup \delta^2, Q_0^1 \cup Q_0^2, F^1 \cup F^2)$. $\mathcal{A}$ accepts a word *iff* it is accepted by $\mathcal{A}^1$ or $\mathcal{A}^2$. The automata $\mathcal{A}_x^1$, $\mathcal{A}_x^2$ and $\mathcal{A}_x^3$ are constructed as follows:

1. $\mathcal{A}_x^1 = (\Gamma \cup \{\#\}, Q^1, \delta^1, \{q_0^1\}, \{f^1\})$, were $Q^1 = \{q_0^1, q_1^1, \ldots, q_{l+1}^1, f^1\}$;
   $\delta^1 = \{(q_i^1, \sigma, q_{i+1}^1\} \cup \{(q_i^1, \sigma, f^1) \mid 1 \leq i \leq l,\ \sigma \neq (i+1)\text{-th element of } \#\#x(\sqcup)^{l_0}\}$

2. $\mathcal{A}_x^2 = (\Gamma \cup \{\#\}, Q^2, \delta^2, \{q_0^2\}, \{f^2\})$, were
   $Q^2 = \{q_0^2,\ q_{\sigma_1}^2,\ q_{\sigma_1\sigma_2}^2,\ q_{\sigma_1\sigma_2\sigma_3 i}^2,\ q_f^2 \mid \sigma_1, \sigma_2, \sigma_3 \in \Gamma \cup \{\#\};\ 1 \leq i \leq l\}$;
   $\delta^2 = \{(q_0^2, \sigma, q_0^2),\ (q_0^2, \sigma, q_\sigma^2),\ (q_{\sigma_1}^2, \sigma, q_{\sigma_1\sigma}^2),\ (q_{\sigma_1\sigma_2}^2, \sigma, q_{\sigma_1\sigma_2\sigma 1}^2),$
   $(q_{\sigma_1\sigma_2\sigma_3 i}^2, \sigma, q_{\sigma_1\sigma_2\sigma_3(i+1)}^2) \mid \sigma, \sigma_1, \sigma_2, \sigma_3 \in \Gamma \cup \{\#\};\ 1 \leq i < l\} \cup$
   $\{(\sigma_{\sigma_1\sigma_2\sigma_3 l}^2, \sigma, f^2) \mid \sigma, \sigma_1, \sigma_2, \sigma_3 \in \Gamma \cup \{\#\};\ \sigma \neq Next(\sigma_1\sigma_2\sigma_3)\}$

3. $\mathcal{A}_x^3 = (\Gamma \cup \{\#\}, Q^3, \delta^3, \{q_0^3\}, \{f^3\})$, were $Q^3 = \{q_0^3, f^3\}$;
   $\delta^3 = \{(q_0^3, \sigma, q_0^3),\ (q_0^3, q_{reject}, f^3) \mid \sigma \in \Gamma \cup \{\#\}\}$

The size of automata $\mathcal{A}_x^1$, $\mathcal{A}_x^2$ and $\mathcal{A}_x^3$ are linear in $l = p(|x|)$ (Γ is fixed). So, the construction of A_x can be performed in polynomial time of $|x|$.

To summarize, we have constructed a polynomial time reduction from any language $L \in$ PSPACE to the universality problem for $\mathbf{NFA}_p^\omega$ and thus, have proven its PSPACE-hardness. $\square$

References

[Baa90] Franz Baader. "Terminological cycles in KL-ONE-based knowledge representation languages." In *Proc. of the 8th Nat. Conf. on Artificial Intelligence (AAAI'90)*, pages 621–626, Boston (Ma, USA), 1990.

[Baa96] Franz Baader. "Using automata theory for characterizing the semantics of terminological cycles." *Ann. of Mathematics and Artificial Intelligence*, 18:175–219, 1996.

[Neb91] Bernhard Nebel. "Terminological cycles: Semantics and computational properties." In John F. Sowa, editor, *Principles of Semantic Networks*, pages 331–361. Morgan Kaufmann, Los Altos, 1991.

[Sip97] Michael F. Sipser. "Introduction to the Theory of Computation.", *PWS Publishing*, 1997

[Tho90] Wolfgang Thomas. "Automata on infinite objects.", in *Handbook of Theoretical Computer Science* (J. van Leeuwen Ed.), Vol. B (Elsevier, Amsterdam 1990), pages 133–191, 1990.

[Var95] Moshe Y. Vardi. "An Automata-Theoretic Approach to Linear Temporal Logic." *Banff Higher Order Workshop*, pages 238–266, 1995.