Automates d'arbre

TD 5 : Alternation and Two-ways

Charlie Jacomme

October 24, 2019

1 Alternation

Exercise 1 : SUCH AWA

Definition 1 *If $\mathcal{X}$ is a set of propositional variables, let $\mathbb{B}(\mathcal{X})$ be the set of positive propositional formulae on $\mathcal{X}$, i.e., formulae generated by the grammar $\phi ::= \bot \mid \top \mid \phi \vee \phi \mid \phi \wedge \phi$.*

Definition 2 *A AWA (Alternating Word Automata) is a tuple $\mathcal{A} = (Q, \Sigma, Q_0, Q_f, \delta)$ where Σ is a finite set (alphabet), Q is a finite set (of states), $Q_0 \subset Q$ (initial states), $Q_f \subseteq Q$ (final states) and δ is a function from $Q \times \Sigma$ to $\mathbb{B}(Q)$ (transition function). A run of $\mathcal{A} = (Q, \Sigma, Q0, Q_f, \delta)$ on a word w is a tree t labelled by Q such that :*
— *if $w = \varepsilon$, then $t = q_0$ with $q_0 \in Q_0$.*
— *if $w = a.w'$, then $t = q_0(t_1, \dots, t_n)$ $q_0 \in Q_0$ and such that for all i, t_i is a run of w' on $(Q, \Sigma, q_i, Q_f, \delta)$ and $\{q_1, \dots, q_n\} \models \delta(q_0, a)$.*

Definition 3 *We say that a run is accepting if every leaf of the form q satisfies that $q \in Q_f$.*

1. Show how to reduce the emptiness problem for an AWA on a one letter alphabet $\{a\}$ with formalas that are in positive disjunctive normal form to the emptiness problem of a tree automaton .
2. Show how to reduce the emptiness problem for a tree automaton to the emptiness problem of an AWA on a one letter alphabet $\{a\}$. Conclude on the complexity of the emptiness problem for an AWA on a one letter alphabet.

Solution:

1. Let $\mathcal{A} = (Q, \{a\}, q_0, Q_f, \delta)$ an AWA. Notice that δ only contains one rule. The accepted trees then have a very particular form, which we can recognize using an NFTA. We construct an NFTA of the form $(Q, \{f_k(k) \mid 0 \leq k \leq n\}, F, \Delta')$ with $F = Q_0$:
$$\delta(q, a) = \bigvee_{i=1}^{n} \bigwedge_{j=1}^{k_i} (q_{i,j}, i) \Rightarrow \forall i, f_i(q_{i,1}, ..., q_{i,k_i}) \longrightarrow q \in \delta'$$
2. Given $\mathcal{A} = (Q, \mathcal{F}, Q_f, \delta)$ an NFTA, we construct an AWA whose alphabets corresponds to labeling of a tree accepted by A. We define the AWA $\mathcal{A}' = (Q \times \mathcal{F}, \{a\}, I, F, \delta')$ with :
$$F = \{(q, f) \mid f \longrightarrow q \in \Delta\}$$
$$I = \{(q, f) \mid q \in Q_f\}$$
$$\delta((q, f), a) = \bigvee_{f(q_1, \dots, q_n) \longrightarrow q \in \delta} \bigwedge_{i=1}^{n} \bigvee_{f_j \in \mathcal{F})} ((q_i, f_j), i)$$
.
We deduce that emptiness for AWA on singleton alphabet is P-hard.

Exercise 2 : Membership

1. Recall the complexity of the uniform membership problem for DFTAs and NFTAs.
2. Prove that **(AlternatingUMembership)** :
   Instance : an AWA $\mathcal{A}$ and a word w
   Question : $w \in L(\mathcal{A})$?
   is in PTIME.

Solution:

1. In the case of DFTA, from a term t and the automaton A, we can compute a run in $O(|t| + |A|)$. In the nondeterministic case, the idea is similar to the word case : the algorithm determinizes along the computation, i.e. for each node of the term, we compute the set of reached states. The complexity of this algorithm will be in $O(|t| \times |A|)$.
   For NFHAs, the idea is similar to the NFTA, and we obtain a PTIME algorithm.
2. Let n be the size of w. Define S_i with $0 \leq i \leq n$ by decreasing induction :
   — $S_n = Q_f$
   — $S_i = \{q \in Q \mid \exists q, w_i \to \phi \in \mathcal{A}, S_{i+1} \models \phi\}$
   Then $w \in L(A)$ iff $I \cap S_0 \neq \varnothing$. All this can be done in PTIME.

Exercise 3 : Two ways

Definition 4 *A two way alternating tree automata $\mathcal{A}$ is given by a finite set of states Q, a set of final statesQ_f, and a transition function δ which associates to each pair $(q, f) \in Q \times \mathcal{F}$ a formula in $\mathbb{B}(Q \times \{-1, 0, \ldots, n\})$ where n is the arity of f.*
A run of A on t is a tree ρ labelled by $Q \times Pos(t)$ such that :
— *$\epsilon \in Pos(\rho)$ and $\rho(\epsilon) = (q, \epsilon)$*
— *If $\omega \in Pos(\rho)$, $\rho(\omega) = (p, q)$ and $\delta(q, t(p)) = \phi$, then there exists $S = \{(q_1, d_1), \ldots, (q_n, d_n)\}$ such that $S \models \phi$ and for all $(q_i, d_i) \in S$,*
  — *$\omega \cdot i \in Pos(\rho)$*
  — *if $d_i > 0$ then $p \cdot d_i \in Pos(t)$ and $\rho(\omega \cdot i) = (p \cdot d_i, q_i)$*
  — *if $d_i = 0$, then $\rho(\omega \cdot i) = (p, q_i)$*
  — *if $d_i = -1$, then $\rho(\omega \cdot i) = (p', q_i)$ with $p = p' \cdot i$.*
A run is accepting if the root is labelled with a final state.

Give an example of an automaton according to the previous Definition and one of its accepting runs.

Exercise 4 : Horn and Two ways

We first recall the notion of two way automatons of TATA.

Definition 5 *A clause $P(u) \leftarrow P_1(x_1), \ldots, P_n(x_n)$ where u is a linear term and $x_1, \ldots, x_n$ are (not necessarily distinct) variables occuring in u, is called a push clause. A clause $P(x) \leftarrow Q(t)$ where x is a variable and t is a linear term, is called a pop clause. A clause $P(x) \leftarrow P_1(x), \ldots, P_n(x)$ is called an alternating clause (or an intersection clause).*

An alternating two-way tree automaton is a tuple $(Q, Q_f, \mathcal{F}, C)$ where Q is a finite set of unary function symbols, Q_f is a subset of Q and C is a finite set of clauses each of which is a push clause, a pop clause or an alternating clause.

Such an automaton accepts a tree t if t belongs to the interpretation of some $P \in Q_f$ in the least Herbrand model of the clauses.

We restrict ourselves to the case where $\mathcal{F}$ only contains unary symbols and constants, and automaton (according to def 4) have a single accepting state q_f, and do not have non deterministic or alternating rules : transitions are of te form $\delta(q, a) = (q', \delta')$ where $d \in \{-1, 1\}$ or $\delta(q, a) = \top$.
We consider a translation from automatons to horn clauses such that given $\mathcal{A}$, we define $C_{\mathcal{A}}$ the minimal set with :
— for any $\delta(q, a) = (q', 1)$, $q'(x) \to q(a(x)) \in C_{\mathcal{A}}$
— for any $f \in \mathcal{F}$ and $\delta(q, a) = (q', -1)$,$q'(f(a(x))) \to q(a(x)) \in C_{\mathcal{A}}$
— for anu $\delta(q, a) = \top$, $\emptyset \to q(a) \in C_{\mathcal{A}}$
Notice the difference between $C_{\mathcal{A}}$, and the clauses defining a two way alternating automata according to Definition 5. We are going to show that this translation, which is the most natural one, does define an automaton in the sense of Definition 4, but that it is however incorrect.

1. Given $\mathcal{A}$ according to Definition 4, provides a two way alternating automata according to Definition 5 which accepts the interpretation of q_f in the smallest Herband model of $C_{\mathcal{A}}$.
2. Give an automaton according to Definition 4 which accepts the empty language but such that the interpretation of q_f in the least Herbrand model of $C_{\mathcal{A}}$ is non empty.

Solution:

1. Assuming that q_a^S and q_a for any symbol a do not appear inside the original automaton We replace each clause inside $C_{\mathcal{A}}$ of the form $q'(f(a(x))) \to q(a(x))$ by the clauses :

— $q'(f(y)) \to q_a^S(y)$
— $q_a^S(a(x)) \to q_a(x)$
— $q_a(x) \to q(a(x))$

2. The idea is to show that automatons from Definition 4 can travel along the same tree, while the clauses only allow to work on set of trees. We set, given three distinct symbols a, b :

   $\delta(q_0, 0) = \top \quad \delta(q_f, a) = (q_1, 1) \quad \delta(q_1, b) = (q_2, -1)$
   $\delta(q_0, b) = (q_0, 1) \quad \delta(q_2, b) = (q_0, 1)$

   Any run of $\mathcal{A}$ must start with the $\delta(q_f, a)$ and then $\delta(q_1, b) = (q_2, -1)$, i.e recognize a term of the form $a(b(u))$. However, as there is no rule for $\delta(q_2, a)$, no tree can be accepted.

   However, the interpretation of q_f inside the least Herbrand model of $C_\mathcal{A}$ contains $a(b(0))$. Indeed, we have inside $C_\mathcal{A}$ the rules :

   (a) $\top \to q_0(0)$
   (b) $q_0(x) \to q_2(b(x))$
   (c) $q_0(x) \to q_0(b(x))$

   (d) $q_2(a(b(x))) \to q_1(b(x))$
   (e) $q_2(b(b(x))) \to q_1(b(x))$

   (f) $q_1(x) \to q_f(a(x))$

   And thus we conclude with :

$$\begin{aligned}\top &\xrightarrow{(a)} q_0(0)\\ &\xrightarrow{(c)} q_0(b(0))\\ &\xrightarrow{(b)} q_2(b(b(0)))\\ &\xrightarrow{(e)} q_1(b(0))\\ &\xrightarrow{(f)} q_1(a(b(0)))\end{aligned}$$