Construction of Thompson's Automaton from a Regular Tree Expression *

Ahlem Belabbaci
LIM - University of Laghouat, Algeria
ah.belabbaci@lagh-univ.dz

Hadda Cherroun
LIM - University of Laghouat, Algeria
h.cherroun@lagh-univ.dz

Djelloul Ziadi
LITIS - University of Rouen, France
Djelloul.Ziadi@univ-rouen.fr

Loek Cleophas
Umeå University, Sweden and
FASTAR Research Group,
Stellenbosch University, South Africa
loek@fastar.org

In this paper we deal with tree automata, particularly the construction of a tree automaton from a regular tree expression. In the late 1960s, Ken Thompson proposed an algorithm for the construction of an automaton from a regular expression over strings. In this work we try to generalize the former algorithm to trees. For this, we modify the general form of the tree automaton in order to facilitate the operations of union, closure and concatenation. This construction allows us to perform pattern matching over trees.

1 Introduction

Trees are widely used in applications nowadays, hence the importance of dealing with tree automata. Among the fields of handling of tree automata appears tree pattern matching, term rewriting, model checking, XML, ...

In the case of words, several algorithms were proposed in order to convert a regular expression into an automaton. The most common construction is the standard or position automaton [7, 12]. Brzozowski's construction [2] of a deterministic finite automaton uses derivatives of regular expressions. This approach was modified by Antimirov [1] who defined partial derivatives to construct a non-deterministic automaton from a regular expression E. Another construction was proposed by Thompson [13] based on induction over the structure of a regular expression.

By analogy to words, some algorithms were proposed for trees. Among these works is the one of Sebti and Ziadi [9], who gave an algorithm to compute the position tree automaton. The work of Kuske and Meinecke [8] consists of the definition of partial derivatives for regular tree expressions and then building a non-deterministic finite tree automaton recognizing the language denoted by such an expression. They adapt and modify the approach of Champarnaud and Ziadi [3, 4] in the word case. Tree derivatives were introduced by Levine in [10, 11] and extend the concept of Brzozowski's string derivatives. He used tree derivatives for minimizing and characterizing tree automata. Tree derivatives are used as a basis of inference of tree automata from finite samples of trees.

In this paper we present the construction of Thompson's automaton for the case of trees. In the next section, preliminaries about trees, tree automata and regular tree expressions are presented. Afterwards we recall the process of the construction of the Thompson's automaton for strings in Section 3. Our

*This work is supported by a South Africa-Algeria Cooperation Project funded by the South African National Research Foundation and the Algerian MESRS-DGRSDT.

Submitted to:
TTATT 2015

construction of the Thompson's automaton for trees is detailed with illustrations in Section 4 with a discussion about the size and features of the proposed construction. The proofs of the equivalence between the different constructions and regular tree expressions are presented in appendices.

2 Preliminaries

A ranked alphabet is a set of symbols provided with an arity (rank) function: $\#\Sigma :\rightarrow \mathbb{N}$. For $a \in \Sigma$, # a is called the arity (rank) of a. A symbol of arity 0 is called a constant, terminal node or leaf. We denote by Σ_n the set of all symbols of arity n of Σ. Σ_0 is the set of leaves. We consider $\varepsilon \in \Sigma_1$ as the empty node.

An ordered labeled tree t with arity over an alphabet Σ can be a constant or of the form $g(t_1,t_2...,t_{\#g})$, such that $t_i,\ 1 < i < \#g$, is itself a tree.

A tree automaton (TA) $\mathscr{A}$ is a tuple $(Q,Q_f,Q_{\Sigma_0},\Delta)$ such that Q is a finite set of states, Q_{Σ_0} is the set of initial states, $Q_f \subset Q$ is the acceptance state and Δ is the set of transition rules. A transition rule is a triplet $((q_1,...,q_n),a,q)$ where $q_1,...,q_n \in Q,\ a \in \Sigma$ *and* $\#a = n$. We use $a(q_1,...,q_n) \rightarrow q$ and $a(q) \rightarrow (q_1,...,q_n)$ to denote respectively bottom-up or top-down automaton transition rules. For $n = 0$ leaves rules are represented by $a \rightarrow q$ or $q \rightarrow a$. The notation $\sigma^*(t) = Q_f$ is used to indicate that there is a sequence of transitions in the automaton *recognizing* the tree t. The language recognized by an automaton is the set of all trees that can be recognized using this automaton.

The graphical representation of the tree automaton is similar to the one of strings. The states are represented by circles with double circles for the final states, and transitions between states by edges labeled with a symbol of the alphabet Σ_0 or ε [5]. For tree automata, some changes are made in the representation of transitions. A transition from state q_0 to states $q_1,q_2,...,q_n$ with a symbol or an ε-transition is represented by i) an edge connecting the state q_0 to a small circle unlabeled with a symbol or ε, ii) n edges connecting the small circle to the state q_i labeled by i where $i : 1..n$. In the case of directed automata (top-down or bottom- up) edges are directed.

A regular tree expression over a ranked alphabet Σ is inductively defined by $E = \varepsilon$, $E \in \Sigma_0$, $E = g(E_1,...,E_n)$, $E = (E_1 + E_2)$, $E = (E_1 {}_c E_2)$, $E = (E_1{}^{*,c})$, where $c \in \Sigma_0$, $n \in N$, $g \in \Sigma_n$ and $E_1,E_2,...,E_n$ are any regular tree expressions over Σ. $\|E\|$ is the number of occurrences of symbols from the ranked alphabet Σ in a regular tree expression E.

3 Thompson's Automaton for Strings

In his 1968 article [13], Ken Thompson describes a technique for the construction of a deterministic finite automaton from a given regular expression.

According to this technique, the regular expression must first be converted into post-fixed notation. Then the automaton is built by successive composition of automata. These automata are assembled according to the basic operations of regular expressions, namely concatenation, union and closure. A Thompson's automaton has the following properties:

- only one initial state and one final state; these two states are distinct;
- there are neither incoming transitions to the initial state nor outgoing transitions to the final one;
- any state is the origin of at most one transition labeled by a letter and of at most two ε-transitions.

Formally, there are five sorts of regular expression: the empty string ε, any character a, the union $E + F$, the concatenation $E \mid F$ and the closure E^*. Figure 3 illustrates the different constructions.

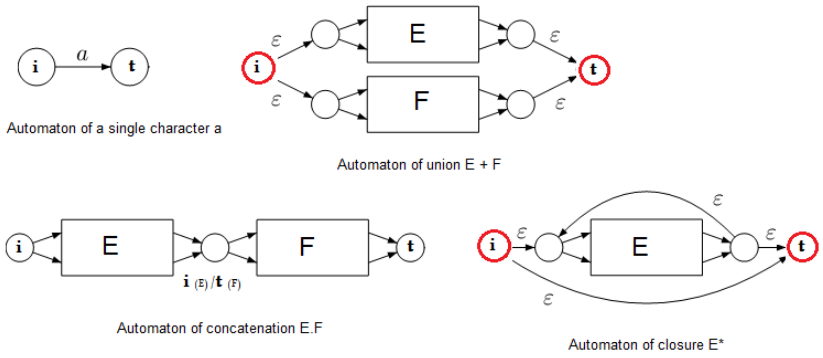

Figure 1: Thompson's Automata for Strings

4 Thompson's Automaton for Trees

Given the difference between strings and trees concerning the concatenation and closure operations, we should take care when constructing the tree automaton. Indeed, we have designed a special form of tree automaton that allows us to inductively construct tree automata in a straightforward way from a regular tree expression.

The basic idea of our construction is to build from a given regular tree expression E, a finite bottom-up tree automaton which has the form illustrated by Figure 2. The main characteristic of this automaton is that it contains one initial state for each element of Σ_0 (the frame Q_{Σ_0}). This condition makes more sense in the operation of concatenation, since we have to perform concatenation just in one state.
In order to keep this form, some ε-transitions are added during the inductive constructions.

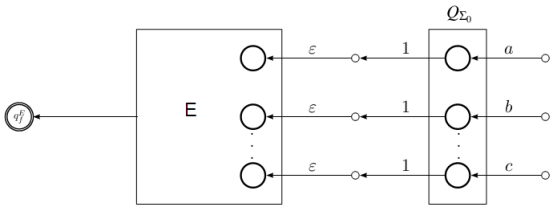

Figure 2: General form of a Thompson tree automaton.

Formally, a (bottom-up) TA Th_E for a regular expression E is a tuple $(Q^E, q_f^E, Q_{\Sigma_0}^E, \Delta^E)$, where:

Q^E : set of all states of Th_E
q_f^E : final state of Th_E
$Q_{\Sigma_0}^E$: set of initial states of Th_E, $Q_{\Sigma_0}^E = \{q_a | a \in \Sigma_0\}$
Δ^E : set of transition rules of Th_E, $\Delta^E : (Q^E)^n \rightarrow Q^E$.

4.1 Construction

In this section, the different constructions of Thompson's tree automata are presented with a formal definition and graphical representation of each automaton.

Elementary Automaton, Leaf Tree Automaton: It is a one-state automaton that consists of an initial state which is itself the accepting state (Figure 3). $Q^E = q_f^E = q_a^E$ and $\Delta^E = \{a \to q_a^E\}$

Automaton of Arity $E = g(E_1, E_2, ..., E_n)$**:** The automaton is built from n other automata E_i by adding a new final state Q_f^E (Figure 4). To keep the basic structure of Thompson's tree automata, we merge sets $Q_{\Sigma_0}^{E_i}$. This is done by:

- adding k initial states q_a^E as $k = |\Sigma_0|$ and $a \in \Sigma_0$,
- adding the transition $a \to q_a^E$ for all $a \in \Sigma_0$,
- removing transitions $a \to q_a^{E_i}$, such that $a \in \Sigma_0$ and $i : 1...n$,
- adding ε-transitions between q_a^E and $q_a^{E_i}$.

This way of merging sets Q_{Σ_0} is also used for the union and concatenation operations. So:

$$\begin{aligned} Q^E &= (\bigcup_{i=1...n} Q^{E_i}) \cup \{q_f^E\} \cup \{q_a^E | a \in \Sigma_0\} \\ \Delta^E &= \{\bigcup_{i=1...n} \Delta^{E_i}\} \setminus \{a \to q_a^{E_i}, a \in \Sigma_0, i = 1 \dots n\} \\ &\quad \cup \{a \to q_a^E | a \in \Sigma_0\} \cup \{g(q_f^{E_1}, q_f^{E_2}, ..., q_f^{E_n}) \to q_f^E\} \cup \{\varepsilon(q_a^E) \to q_a^{E_i} | a \in \Sigma_0, i = 1 \dots n\} \end{aligned}$$

Automaton of Union $E = F + G$**:** The union of two Thompson's automata F and G is built by adding a new final state connecting the final states of the two automata q_f^F and q_f^G with ε-transitions and merging Q_{Σ_0} sets. See graphical representation for Th_E in Figure 5.

Formally:

$$\begin{aligned} Q^E &= Q^F \cup Q^G \cup \{q_f^E\} \cup \{q_a^E | a \in \Sigma_0\} \\ \Delta^E &= \{\Delta^F\} \setminus \{a \to q_a^F, a \in \Sigma_0\} \cup \{\Delta^G\} \setminus \{a \to q_a^G, a \in \Sigma_0\} \\ &\quad \cup \{\varepsilon(q_a^E) \to q_a^F | a \in \Sigma_0\} \cup \{\varepsilon(q_a^E) \to q_a^G | a \in \Sigma_0\} \cup \{\varepsilon(q_f^F) \to q_f^E |\} \\ &\quad \cup \{\varepsilon(q_f^G) \to q_f^E |\} \cup \{a \to q_a^E | a \in \Sigma_0\} \end{aligned}$$

Automaton of Concatenation $E = F_{.c}G$**:** The construction of the concatenation automaton (Figure 6) is a little bit different. We remove the transition $c \to q_c$ (for c the concatenation symbol) from Δ^F, then we add a ε-transition from the final state of G to q_c. Finally, we merge sets Q_{Σ_0} of F and G.

Then:

$$\begin{aligned} Q^E &= Q^F \cup Q^G \cup \{q_f^E\} \cup \{q_a^E | a \in \Sigma_0\} \\ \Delta^E &= \{\Delta^F\} \setminus \{a \to q_a^F, a \in \Sigma_0\} \cup \{\Delta^G\} \setminus \{a \to q_a^G, a \in \Sigma_0\} \\ &\quad \cup \{a \to q_a^E | a \in \Sigma_0\} \cup \{\varepsilon(q_a^E) \to q_a^G | a \in \Sigma_0\} \cup \{\varepsilon(q_a^E) \to q_a^F | a \in (\Sigma_0 \setminus \{c\})\} \cup \{\varepsilon(q_f^G) \to q_f^c\} \\ Q_f^E &= Q_f^F \end{aligned}$$

Automaton of Closure $E = F^{*,c}$ For the construction of Thompson's automaton of the closure operation $(*)$, we extend the basic automaton with three ε-transitions like in Figure 7:

- a transition from the final state of F to the state representing the concatenation symbol in F,
- a transition from the state representing the concatenation symbol in E to the final state of E,
- and another transition from the final state of F to the final state of E.

Formally:

$$\begin{aligned} Q^E &= Q^F \cup \{q_f^E\} \cup \{q_a^E | a \in \Sigma_0\} \\ \Delta^E &= \{\Delta^F\} \setminus \{a \to q_a^F, a \in \Sigma_0\} \\ &\quad \cup \{a \to q_a^E | a \in \Sigma_0\} \cup \{\varepsilon(q_c^E) \to q_f^E\} \cup \{\varepsilon(q_c^F) \to q_f^F\} \cup \{\varepsilon(q_f^F) \to q_f^E\} \end{aligned}$$

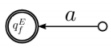

Figure 3: The leaf tree automaton.

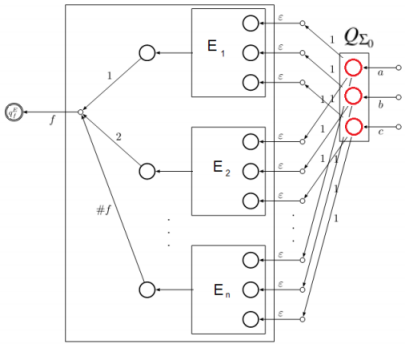

Figure 4: The Automaton for $E = f(E_1, E_2, ..., E_n)$.

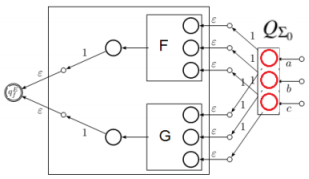

Figure 5: The automaton for $E = F + G$.

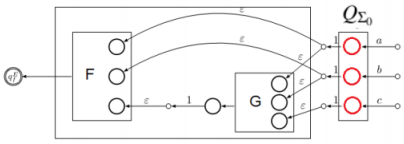

Figure 6: The automaton for $E = \mathrm{F}._c\mathrm{G}$.

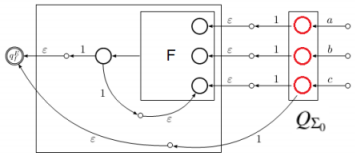

Figure 7: The automaton for $E = F^{*,c}$.

4.2 Validity of the Construction

We should prove the equivalence between the regular language of the constructed Thompson's automaton and the regular language recognized by E.

Definition 1. *Let t be a tree, we define the function ε-closure(t) that removes ε-nodes from t as follows:*

ε-*closure*$(a) = a$ *for each* $a \in \Sigma_0$

ε-*closure*$(\varepsilon(t)) = t$

ε-*closure*$(g(t_1, t_2, ..., t_n)) = g(\varepsilon$-*closure*$(t_1)$, ε-*closure*(t_2), ..., ε-*closure*$(t_n))$

From the definition of ε-*closure*(t) we can deduce the following properties:

Property 1. *For each trees two t_1, t_2, we have ε-closure$(t_1._c t_2) = [\varepsilon$-closure$(t_1)]._c$ ε-closure(t_2).*

This property can be extended to sets of trees.

Property 2. ε-*closure*$(t.{}_c\{t_1, t_2, ... t_k\}) = [\varepsilon$-*closure*$(t)]._c\{\varepsilon$-*closure*$(t_1), \varepsilon$-*closure*$(t_2), ...\varepsilon$-*closure*$(t_k)\}$

Theorem 1. *Let E be a regular tree expression, Th_E is Thompson's automaton created for E, $[\![E]\!]$ is the regular language recognizing E and $\mathscr{L}(Th_E)$ is the regular language recognized by Th_E.*

Then, we have ε-closure$(\mathscr{L}(Th_E)) \equiv [\![E]\!]$.

In order to prove this theorem, we prove the two next lemmas (in Appendices A and B).

Lemma 1. *For each $t \in [\![E]\!]$, there exists a tree $t' \in \mathscr{L}(Th_E)$ so that ε-closure$(t') = t$.*

Lemma 2. *If $t \in \mathscr{L}(Th_E)$, then ε-closure$(t) \in [\![E]\!]$.*

We now consider features and size of our construction of a tree automaton from a regular tree expression. The number of states and transitions generated for each construction in section 4.1 is around $\|E\|$, since we can consider each regular tree expression as a tree where leaves represent symbols of the alphabet and internal nodes represent regular tree operators (arity, union, concatenation and closure).

5 Conclusion

In this paper, we have presented a method for the construction of Thompson's automaton from a regular tree expression: we have proposed a general form of the automaton to facilitate the operations of closure and concatenation. Proofs of the equivalence between the automaton generated and the original regular expression are provided for the different operations on the automaton. Future work will be focused on tree pattern matching using this automaton.

References

[1] Valentin Antimirov (1995): *Partial Derivatives of Regular Expressions and Finite Automata Constructions*. *Theoretical Computer Science* 155, pp. 291–319.

[2] Janusz A. Brzozowski (1964): *Derivatives of Regular Expressions*. *J. ACM* 11(4), pp. 481–494.

[3] Jean-Marc Champarnaud & Djelloul Ziadi (2001): *From C-Continuations to New Quadratic Algorithms for Automaton Synthesis*. *IJAC* 11(6), pp. 707–736.

[4] Jean-Marc Champarnaud & Djelloul Ziadi (2002): *Canonical derivatives, partial derivatives and finite automaton constructions*. *Theor. Comput. Sci.* 289(1), pp. 137–163.

[5] Loek Cleophas (2008): *Tree Algorithms: Two Taxonomies and a Toolkit*. PhD thesis, Technische Universiteit Eindhoven, Department of Mathematics and Computer Science.

[6] H. Comon, M. Dauchet, R. Gilleron, C. Loding, F. Jacquemard, D. Lugiez, S. Tison & M. Tommasi (2008): *Tree Automata Techniques and Applications*. Available on: `http://www.grappa.univ-lille3.fr/tata`. Release November, 18th 2008.

[7] Victor M. Glushkov (1961): *The Abstract Theory of Automata*. *Russian Mathematical Surveys* 16(5), pp. 1–53.

[8] Dietrich Kuske & Ingmar Meinecke (2008): *Construction of Tree Automata from Regular Expressions*. In: *Developments in Language Theory, 12th International Conference, DLT 2008, Kyoto, Japan, September 16-19, 2008. Proceedings*, pp. 491–503.

[9] Éric Laugerotte, Nadia Ouali Sebti & Djelloul Ziadi (2013): *From Regular Tree Expression to Position Tree Automaton*. In: *Language and Automata Theory and Applications - 7th International Conference, LATA 2013, Bilbao, Spain, April 2-5, 2013. Proceedings*, pp. 395–406.

[10] Barry Levine (1981): *Derivatives of Tree Sets with Applications to Grammatical Inference*. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* PAMI-3(3), pp. 285–293.

[11] Barry Levine (1982): *The Use of Tree Derivatives and a Sample Support Parameter for Inferring Tree Systems*. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* PAMI-4(1), pp. 25–34.

[12] R. McNaughton & H. Yamada (1960): *Regular Expressions and Finite State Graphs for Automata*. *IRE Trans. on Electronic Comput* EC-9(1), pp. 38–47.

[13] Ken Thompson (1968): *Programming Techniques: Regular Expression Search Algorithm*. *Commun. ACM* 11(6), pp. 419–422.

Appendix A: Proof of Lemma 1

The proof is accomplished by induction on the construction of the tree automaton.

Leaf Tree

$E = a$. This case is obvious because: $\mathscr{L}(\mathrm{Th_E}) = \{a\}$, $[\![E]\!] = \{a\}$, so $\mathscr{L}(\mathrm{Th_E}) \equiv [\![E]\!]$, so for all $t \in [\![E]\!]$, there exists $t' \in \mathscr{L}(\mathrm{Th_E})$ such as $\varepsilon\text{-}closure(t') = t$. □

Arity

$E = g(E_1, E_2, ..., E_n)$. where n is the arity of the function f.
Let $t \in [\![E]\!]$, so $t = g(e_1, e_2, ..., e_n)$, such as $e_1 \in [\![E_1]\!]$, $e_2 \in [\![E_2]\!]$, ..., $e_n \in [\![E_n]\!]$.

According to the induction hypothesis, there exists $t'_i \in \mathscr{L}(\mathrm{Th}_{E_i})$ such as
$\varepsilon\text{-}closure(t'_i) = t_i$ with $i = 1...n$.
Let us construct a term $t' = g(t'_1, t'_2, ..., t'_n)$ and replacing each symbol x of arity 0 by $\varepsilon(x)$. This means that $t' = (\ g(t'_1, t'_2, ..., t'_n)\ )_{.a}\varepsilon(a)...._c\varepsilon(c)....$
We now show that $t' \in \mathscr{L}(\mathrm{Th_E})$ and that $\varepsilon\text{-}closure(t') = t$.

According to the construction of Thompson's automaton for the arity, transition $a \to q_a^{E_i}$ in the path of recognizing t'_i in Th_{E_i} is replaced by $a \to q_a^E$ and $\varepsilon(q_a^E) \to q_a^{E_i}$ for each $i = 1...n$ and $a \in \Sigma_0$.
As we have $\Delta^{E_i} \subset \Delta^E$ and $t'_i \in \mathscr{L}(\mathrm{Th}_{E_i})$, it means $\sigma^*_{E_i}(t'_i) = q_f^{E_i}$.

So: $\sigma^*_E(t'_i) = q_f^{E_i}$, $i = 1...n$.
Moreover we have $g(q_f^{E_1}, q_f^{E_2}, ..., q_f^{E_n}) \to q_f^E \in \Delta^E$, so: $\sigma^*_E(t') = q_f^E$.
Therefore t' is recognized by $\mathscr{L}(\mathrm{Th_E})$.

On the other hand we have:

$$\begin{aligned}
\varepsilon\text{-}closure(t') &= \varepsilon\text{-}closure(g(t'_1, t'_2, ..., t'_n)_{.a}\varepsilon(a)...._c\varepsilon(c)...) \\
&= \varepsilon\text{-}closure(g(t'_1, t'_2, ..., t'_n))_{.a}(\varepsilon\text{-}closure(\varepsilon(a)))...._c\varepsilon(\varepsilon\text{-}closure(\varepsilon(c)))... \\
&= \varepsilon\text{-}closure(g(t'_1, t'_2, ..., t'_n))_{.a}a, ...,_c c. \\
&= \varepsilon\text{-}closure(g(t'_1, t'_2, ..., t'_n)) \\
&= g(\varepsilon\text{-}closure(t'_1), \varepsilon\text{-}closure(t'_2), ..., \varepsilon\text{-}closure(t'_n)) \\
&= g(t_1, t_2, ..., t_n) = t.
\end{aligned}$$

Thus there exists $t' \in \mathscr{L}(\mathrm{Th_E})$ such that $\varepsilon\text{-}closure(t') = t$. □

Union

$E = F + G$. We have $[\![E]\!] = [\![F]\!] \cup [\![G]\!]$.
Let $t \in [\![E]\!]$. Without loss of generality, we suppose that $t \in [\![F]\!]$.

According to the induction hypothesis, there exists $t'_F \in \mathscr{L}(\mathrm{Th_F})$: $\varepsilon\text{-}closure(t'_F) = t$.
Let us construct a term $t' = \varepsilon(t'_F)$, Then we replace each symbol x of arity 0 by $\varepsilon(x)$, that is to say: $t' = (t'_F)_{.a}\varepsilon(a)...._c\varepsilon(c)....$
Let us show that $t' \in \mathscr{L}(\mathrm{Th_E})$ and that $\varepsilon\text{-}closure(t')= t$

According to the construction of Thompson for the union, we replace each transition $a \to q_a^F$ by $a \to q_a^E$ and $\varepsilon(q_a^E) \to q_a^F$ such as $a \in \Sigma_0$.
As $\Delta^F \subset \Delta^E$ ans $t'_i \in \mathscr{L}(\mathrm{Th_F})$, ie. $\sigma_F^*(t'_F) = q_f^F$.
So: $\sigma_E^*(t'_F) = q_f^F$.

Moreover we have $\varepsilon(q_f^F) \to q_f^E \in \Delta^E$, then: $\sigma_E^*(t'_F) = q_f^E$, which means that t' is recognized by $\mathscr{L}(\mathrm{Th_E})$. Furthermore we have:

$$\begin{array}{rcl} \varepsilon\text{-}closure(t') & = & \varepsilon\text{-}closure(\varepsilon(t'_F)_{.a}\varepsilon(a)\ldots_{.c}\varepsilon(c)\ldots) \\ & = & \varepsilon\text{-}closure(t'_F)_{.a}a,\ldots,_{.c}c. \\ & = & \varepsilon\text{-}closure(t'_F) = t. \end{array}$$

Thus there exists $t' \in \mathscr{L}(\mathrm{Th_E})$ such that $\varepsilon\text{-}closure(t') = t$. □

Concatenation

$E = F._cG$. Let $t \in [\![E]\!]$, $t \in [\![F]\!]._c[\![G]\!]$, it means $t \in \{(t_F)_{.c}\{t_1,\ldots,t_k\}\}$, such that $t_i \in [\![G]\!]$, $i = 1\ldots k$.

According to the induction hypothesis, there exists $t'_F, t_1, \ldots, t_k$ with $t'_F \in \mathscr{L}(\mathrm{Th_F})$ and $t'_i \in \mathscr{L}(\mathrm{Th_G})$, $i = 1\ldots k$, such as $\varepsilon\text{-}closure(t'_F) = t_F$ and $\varepsilon\text{-}closure(t'_k) = t_k$.

Let us build two terms t''_F and t''_j by replacing each symbol x of arity 0 by $\varepsilon(x)$, ie.
$t''_F = (t'_F)_{.a}\varepsilon(a)\ldots_{.c}\varepsilon(c)\ldots.$
$t''_j = (t'_j)_{.a}\varepsilon(a)\ldots_{.c}\varepsilon(c)\ldots.$
$t' \in \{\varepsilon(t''_F)_{.c}\{t''_1, t''_2, \ldots, t''_k\}\}$.

Let us prove that $t' \in \mathscr{L}(\mathrm{Th_E})$.

According to the construction of Thompson's automata of concatenation we replaced the transition $a \to q_a^G$ dans Th_G by $a \to q_a^E$ and $\varepsilon(q_a^E) \to q_a^G$ in Th_E by $a \in \Sigma_0$.
And as we have $\Delta^G \subset \Delta^E$ and $\sigma_G^*(t'_j) = q_f^G$, then $\sigma_E^*(t''_F) = q_f^G$.

According to the same construction, we replaced the transition $a \to q_a^F$ by $a \to q_a^E$ and $\varepsilon(q_a^E) \to q_a^F$ for $a \neq c$, where c is the symbol of concatenation.
If $a = c$, this transition $(c \to q_c^F)$ is replaced by $\varepsilon(q_f^G) \to q_c^F$.

Since t'_F is recognized by Th_F by the induction hypothesis, then: $\sigma_F^*(t'_F) = q_f^F$.
As we have $\varepsilon(q_a^E) \to q_a^F \in \Delta^E$ for $a \neq c$ and $\varepsilon(q_f^G) \to q_c^F \in \Delta^E$ otherwise (if $a = c$) and $(\Delta^F \setminus \{c \to q_c^F\}) \subset \Delta^E$, alors nous avons $\sigma_E^*(t') = q_f^F$.

Therefore $\sigma_E^*(\varepsilon(t')) = q_f^E$ because $\varepsilon(q_f^F) \to q_f^E \in \Delta^E$. So t' is recognized by $\mathscr{L}(\mathrm{Th_E})$.

Furthermore we have:

$$\begin{array}{rcl} \varepsilon\text{-}closure(\varepsilon(t')) & = & \varepsilon\text{-}closure(t'). \\ \varepsilon\text{-}closure(t') & \in & \{\varepsilon\text{-}closure(\varepsilon(t''_F)_{.c}(t''_1,...,t''_k))\} \\ & \in & \{\varepsilon\text{-}closure(t''_F)_{.c}\varepsilon\text{-}closure(t''_1),\ ...,\ \varepsilon\text{-}closure(t''_k)\} \end{array}$$

Let us replace terms t''_j and t''_F respectively by $\varepsilon\text{-}closure(t'_j)_{.a}\varepsilon(a),...,_{.c}\,\varepsilon(c)$ and $\varepsilon\text{-}closure(t'_F)_{.a}\varepsilon(a),...,_{.c}\,\varepsilon(c)$.

.

we will have: $\varepsilon\text{-}closure(t') \in \varepsilon\text{-}closure(t'_F)_{.c}\{\varepsilon\text{-}closure(t'_1),\ ...,\ \varepsilon\text{-}closure(t'_k)\}$.

So $t \in \{(t_F)_{.c}\{t_1,...,t_k\}\}$. □

Closure

$E = F^{*c}$. Let $t \in [\![F^{*c}]\!]$:

$$t \in \begin{cases} [\![F^{0,c}]\!] & \text{if } n = 0, \\ [\![F^{n,c}]\!], n \in \mathbb{N}^* & \text{otherwise.} \end{cases}$$

We will show that for all $t \in F^{*c}$, there exists $t' \in \mathscr{L}(\mathrm{Th_E})$, such as
$\varepsilon\text{-}closure(t')= t$.

For the case $n = 0$, $t \in [\![F^{0,c}]\!]$, $t^0 = c$, this case is obvious because we have:
$\{c \to q_c^E, \varepsilon(q_c^E) \to q_f^E\} \in \Delta^E$.
Then $t'^0 = \varepsilon(t^0)$.
So t'^0 is recognized by the automaton Th_E and $\varepsilon\text{-}closure(t'^0) = t^0$.

For $n = 1$, $t \in [\![F^{1,c}]\!]$, $t^1 = \mathrm{c}._c[\![F]\!]$, we demonstrate that there exists
$t'^1 \in \mathscr{L}(\mathrm{Th_E})$ and $\varepsilon\text{-}closure(t'^1)= t^1$.
$t'^1 \in \{\mathrm{c}._c\{t'^1_i\}/t'^1_i \in [\![F]\!]\}$ then $t'^1 \in \{\mathrm{c}._c\{t'^1_1,...,t'^1_j\}\}$.

According to the induction hypothesis $\{c, t'^1_i\} \in \mathscr{L}(\mathrm{Th_F})$, it means $\sigma^*_F(t'^1_i) = q_f^F$. And depending on the construction of Thompson's automata for the closure operation, we have $\varepsilon(q_f^F) \to q_f^E \in \Delta^E$, so $\sigma^*_E(t'^1_i) = q_f^E$. Therefore, $t'^1 \in \mathscr{L}(\mathrm{Th_E})$.

Furthermore, we have:

$$\begin{array}{rcl} \varepsilon\text{-}closure(t'^1_i) & \in & \{\varepsilon\text{-}closure(\mathrm{c}._c\{t'^1_1,...,t'^1_j\})\} \\ & \in & \{\varepsilon\text{-}closure(c)._c\{\varepsilon\text{-}closure(t'^1_1,...,t'^1_j)\}\} \\ & \in & \{c._c\{t^1_1,...,t^1_j\}\} \end{array}$$

For the case $(1 < k \leq n)$, the closure operation comes down to a concatenation. $t^k = t^{k-1}._c[\![F]\!]$.
Thus there exists $t'^k \in \mathscr{L}(\mathrm{Th_E})$ such that $\varepsilon\text{-}closure(t'^k) = t^k$. □

Appendix B: Proof of Lemma 2

We remind that proofs are accomplished by induction on the construction of the tree automaton.

Leaf tree

$E = a$. This case is obvious because $t = a$ and $\varepsilon\text{-}closure(t) = [\![E]\!]$. □

Arity

$E = g(E_1, E_2, ..., E_n)$. We have $t \in \mathscr{L}(\mathrm{Th_E})$ wich means that $t = g(t_1, t_2, ..., t_n)$.
According to the induction hypothesis, there exists $t_1, t_2, ..., t_n$ such as:
$t_1 \in \mathscr{L}(\mathrm{Th}_{E_1})$ and $\varepsilon\text{-}closure(t_1) \in [\![E_1]\!]$.
$t_2 \in \mathscr{L}(\mathrm{Th}_{E_2})$ and $\varepsilon\text{-}closure(t_2) \in [\![E_2]\!]$.
$\vdots$
$t_n \in \mathscr{L}(\mathrm{Th}_{E_n})$ and $\varepsilon\text{-}closure(t_n) \in [\![E_n]\!]$.

We have:
$g(\varepsilon\text{-}closure(t_1), \varepsilon\text{-}closure(t_2), ..., \varepsilon\text{-}closure(t_n)) \in [\![g(E_1, E_2, ..., E_n)]\!]$.
According to the definition 1 of the function $\varepsilon\text{-}closure(t)$, we have:
$g(\varepsilon\text{-}closure(t_1), \varepsilon\text{-}closure(t_2), ..., \varepsilon\text{-}closure(t_n)) = \varepsilon\text{-}closure(g(t_1, t_1, ..., t_n))$.
So, $\varepsilon\text{-}closure(g(t_1, t_1, ..., t_n)) \in [\![g(E_1, E_2, ..., E_n)]\!]$.
Therefore, $\varepsilon\text{-}closure(g(t_1, t_1, ..., t_n)) \in [\![E]\!]$.

Furthermore, we have: $t = g(t_1, t_2, ..., t_n)$, then:
$\varepsilon\text{-}closure(t) \in [\![E]\!]$. □

Union

$E = F + G$. We have $t \in \mathscr{L}(\mathrm{Th_E})$ wich means that $t = t_f$ or $t = t_g$, where
$t_f \in \mathscr{L}(\mathrm{Th_F})$ and $t_g \in \mathscr{L}(\mathrm{Th_G})$.

According to the induction hypothesis:
$t_f \in \mathscr{L}(\mathrm{Th_F})$ and $\varepsilon\text{-}closure(t_f) \in [\![F]\!]$.
$t_g \in \mathscr{L}(\mathrm{Th_G})$ and $\varepsilon\text{-}closure(t_g) \in [\![G]\!]$.
Without loss of generality, we suppose that $t = t_f$.

We have $\varepsilon\text{-}closure(t_f) \in [\![F]\!]$, then $\varepsilon\text{-}closure(t_f) \in [\![F + G]\!]$.

So, $\varepsilon\text{-}closure(t) \in [\![F + G]\!]$.
As we have $E = F + G$, then $\varepsilon\text{-}closure(t) \in [\![E]\!]$. □

Concatenation

$E = F._c G$. If $t \in \mathscr{L}(\mathrm{Th_E})$ then t is of the form $t = t_{f._c} t_g$.

According to the induction hypothesis:
$t_f \in \mathscr{L}(\mathrm{Th_F})$ et ε-$closure(t_f) \in [\![F]\!]$.
$t_g \in \mathscr{L}(\mathrm{Th_G})$ et ε-$closure(t_g) \in [\![G]\!]$.

We have ε-$closure(t_f)_{.c}\ \varepsilon$-$closure(t_g) \in [\![F_{.c}G]\!]$.
According to the property 2, we have
ε-$closure(t_f{}_{.c}t_g) = \varepsilon$-$closure(t_f)_{.c}\ \varepsilon$-$closure(t_g)$.
So, ε-$closure(t) \in [\![F_{.c}G]\!]$.

Then ε-$closure(t) \in [\![E]\!]$. □

Closure

$E = F^{*c}$. If $t \in \mathscr{L}(\mathrm{Th_E})$, then t is of the form: $t = {t_f}^{n,c}$, $t_f \in [\![F]\!]$ and

$$ {t_f}^{n,c} = \begin{cases} {t_f}^{0,c} = c & \text{si } n = 0, \\ {t_f}^{1,c} = ({t_f}^{0,c})_{.c}t_f & \text{si } n = 1, \\ {t_f}^{i,c} = ({t_f}^{i-1,c})_{.c}t_f & \text{si } n > 2. \end{cases} $$

According to the induction hypothesis, we have $t_f \in \mathscr{L}(\mathrm{Th_F})$ and
ε-$closure(t_f) \in [\![F]\!]$.

For the case $t = {t_f}^{0,c} = c$, we have ε-$closure(t_f) = \varepsilon$-$closure(c) = c$.

We also have $c \in [\![F]\!]$, then $c \in [\![F^{0,c}]\!]$.

Therefore, ε-$closure(t) \in [\![E]\!]$.

For $t = {t_f}^{1,c} = ({t_f}^{0,c})_{.c}t_f = t_f$, we have ε-$closure(t) = \varepsilon$-$closure(t_f)$.
ε-$closure(t_f) \in [\![F]\!]$ according to the induction hypothesis, then
ε-$closure(t_f) \in [\![F^{1,c}]\!]$, because $[\![F^{1,c}]\!] = [\![F^{0,c}]\!] \cup [\![F^{0,c}_{.c}F]\!]$ according to the closure property of regular languages.

Therefore, ε-$closure(t) \in [\![E]\!]$.

For ${t_f}^{n,c}$ with $n > 1$, closure is a special case of the concatenation, and this was proven before.

Then, for $t = {t_f}^{n,c}$ and $E = F^{*c}$, ε-$closure(t) \in [\![E]\!]$. □