Logique avancée

6 - Automates d'arbre

Master Info Nice Sophia Antipolis
E. Lozes

14. mai 2019

Contenu de la séance d'aujourd'hui

- Arbres
- Automates d'arbre
- Propriétés de clôture
- Applications

Arbre finis ordonnés enracinés

On dira plus simplement dans ce cours « arbre ».

Définition

Un *arbre* est un ensemble de suites d'entiers $T \subseteq \mathbb{N}^*$ tel que pour tout $w \in \mathbb{N}^*$, $i \in \mathbb{N}$,

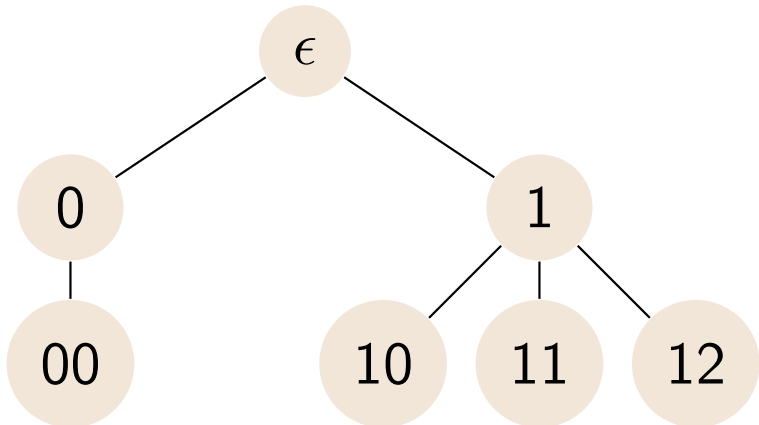
- si $w \cdot (i+1) \in T$, alors $w \cdot i \in T$, et
- si $w \cdot 0 \in T$, alors $w \in T$

Interprétation

- T est l'ensemble des noeuds
- on identifie en fait le noeud et le chemin qui y mène depuis la racine
- ϵ est la racine
- le $i+1$ ème fils de w est $w \cdot i$

Exemple

$T = \{\epsilon, 0, 00, 1, 10, 11, 12\}$



Alphabet avec arité

Définition

Un *alphabet avec arité* est un tuple (Σ, ar), tel que

- Σ est un ensemble fini de symboles
- $\mathrm{ar} : \Sigma \to \mathbb{N}$ associe à chaque symbole une arité

Exemple : un alphabet pour les formules booléenes

- $\Sigma = \{\top, \bot, \vee, \wedge, \neg, p_1, \ldots, p_n\}$
- $\mathrm{ar}(\top) = \mathrm{ar}(\bot) = 0$
- $\mathrm{ar}(\vee) = \mathrm{ar}(\wedge) = 2$
- $\mathrm{ar}(\neg) = 1$
- $\mathrm{ar}(p_1) = \cdots = \mathrm{ar}(p_n) = 0$

Σ-arbre

(aussi appelé Σ-terme)

Définition

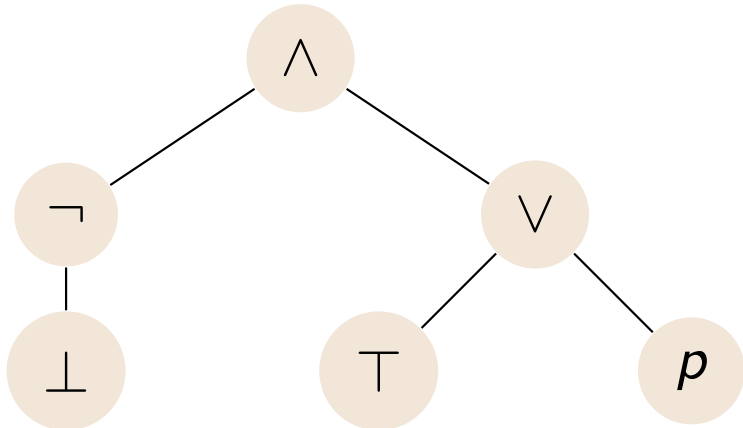
Un Σ-arbre est un tuple (T, t), tel que

- T est un arbre
- $t : T \to \Sigma$ est un étiquetage des noeuds
- l'étiquetage respecte l'arité : pour tout noeud $w \in T$,

$$\mathrm{ar}(t(w)) = n \qquad \text{ssi} \qquad w \text{ a exactement } n \text{ fils.}$$

Exemple

$w \in T$	ϵ	0	00	1	10	11
t(w)	$\wedge$	$\neg$	$\bot$	$\vee$	$\top$	p



Automates d'arbre ascendant (Bottom-Up)

Définition

Un automate d'arbre ascendant ($BUTA$) est un tuple $\mathcal{A} = (Q, \Sigma, \delta, F)$, avec

- un ensemble fini Q d'états
- un alphabet Σ (avec fonction d'arité ar)
- des fonctions de transition $\delta = \{\delta_a | a \in \Sigma\}$, où

$$\delta_a : Q^{\mathrm{ar}(a)} \to \mathcal{P}(Q)$$

- un ensemble fini d'états acceptants $F \subseteq Q$.

Exécution

Une exécution du BUTA (Q, Σ, δ, F) sur l'arbre (T, t) est un Q-arbre (T, q) tel que pour tout noeud $w \in T$ avec n fils, on ait

$$q(w) \in \delta(t(w \cdot 0), t(w \cdot 1), \ldots, t(w \cdot n))$$

Exemple

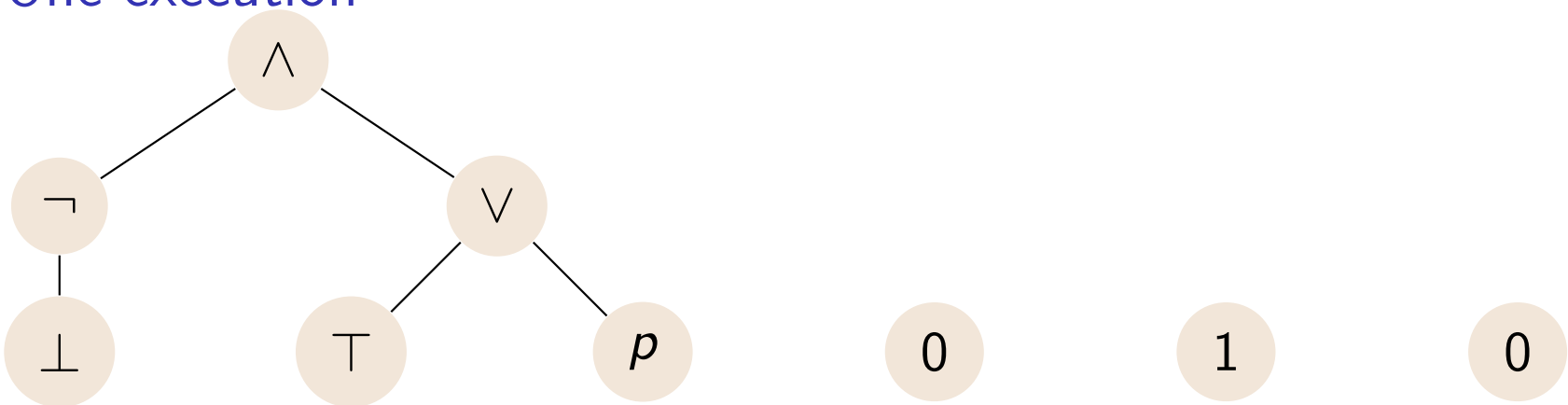
Un BUTA

$$Q = \{0, 1\} \qquad \Sigma = \{\top, \bot, \vee, \wedge, \neg, p\} \qquad F = \{1\} \qquad \delta_\top = \{1\}$$

$$\delta_\bot = \{0\} \qquad \delta_p = \{0, 1\} \qquad \delta_\neg(q) = \{1 - q\}$$

$$\delta_\vee(q_1, q_2) = \{q_1 + q_2 - q_1.q_2\} \qquad \delta_\wedge = \{q_1.q_2\}$$

Une exécution



Exemple

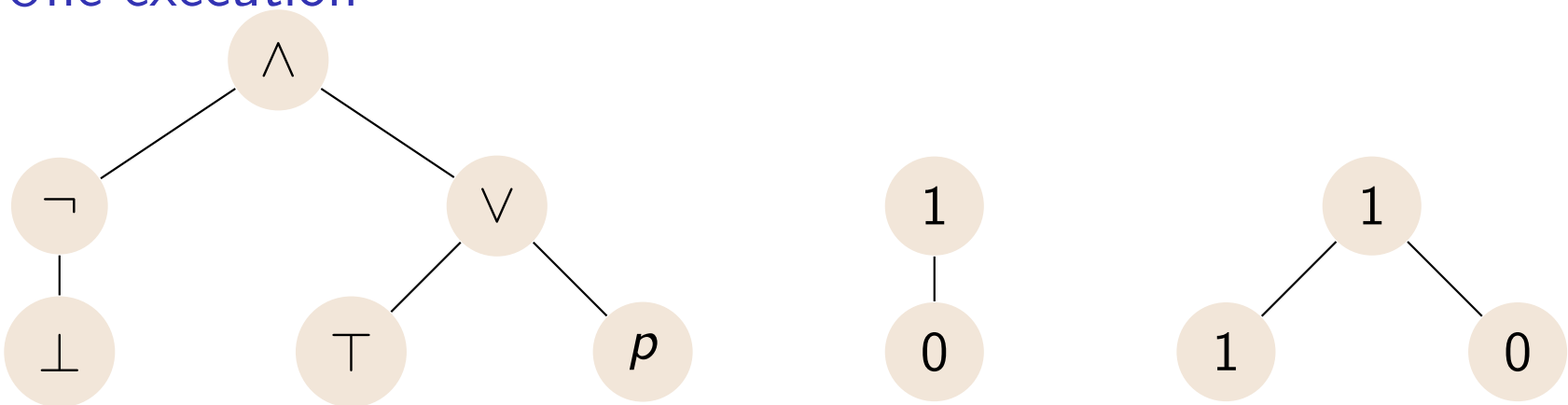
Un BUTA

$$Q = \{0, 1\} \qquad \Sigma = \{\top, \bot, \vee, \wedge, \neg, p\} \qquad F = \{1\} \qquad \delta_\top = \{1\}$$

$$\delta_\bot = \{0\} \qquad \delta_p = \{0, 1\} \qquad \delta_\neg(q) = \{1 - q\}$$

$$\delta_\vee(q_1, q_2) = \{q_1 + q_2 - q_1.q_2\} \qquad \delta_\wedge = \{q_1.q_2\}$$

Une exécution



Exemple

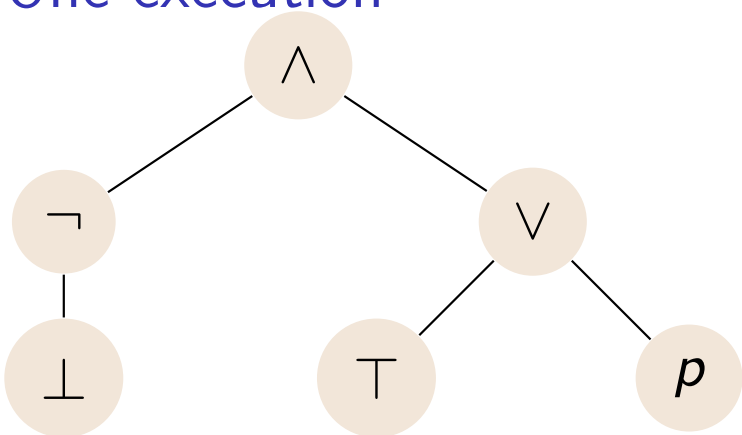
Un BUTA

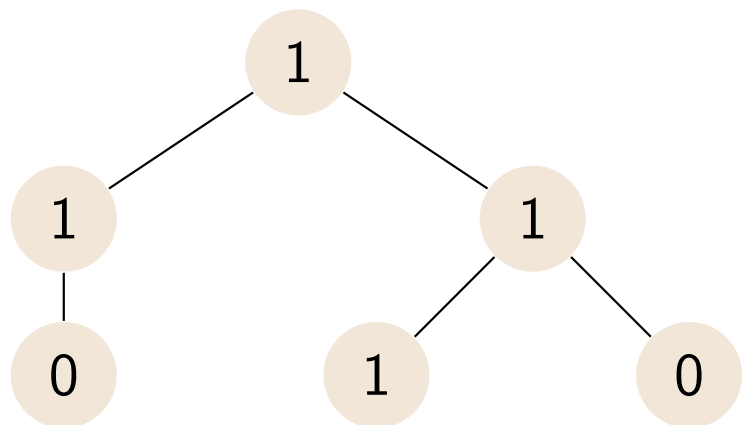
$$Q = \{0, 1\} \qquad \Sigma = \{\top, \bot, \vee, \wedge, \neg, p\} \qquad F = \{1\} \qquad \delta_\top = \{1\}$$

$$\delta_\bot = \{0\} \qquad \delta_p = \{0, 1\} \qquad \delta_\neg(q) = \{1 - q\}$$

$$\delta_\vee(q_1, q_2) = \{q_1 + q_2 - q_1.q_2\} \qquad \delta_\wedge = \{q_1.q_2\}$$

Une exécution





Les automates d'arbre vus comme des systèmes de réécriture de termes particuliers

Une transition = une règle de réécriture de terme

Par exemple

$$\begin{array}{rr} \top \to 1 & 1 \vee x \to 0 \\ \bot \to 0 & 0 \vee x \to x \\ \neg 0 \to 1 & 1 \wedge x \to x \\ \neg 1 \to 1 & 0 \wedge x \to 0 \\ p \to 0 & p \to 1 \end{array}$$

restriction sur la réécriture « version BUTA »

- un seul symbole de Σ dans le membre gauche
- un seul état de Q dans le membre droit

Condition d'acceptation

$t \in L(\mathcal{A})$ ssi $t \to^* q$ et $q \in F$.

Langage d'un BUTA

Définitions

- une exécution (T, q) est *acceptante* si la racine est étiquetée par un état acceptant, i.e. $q(\epsilon) \in F$
- le *langage* $L(\mathcal{A})$ d'un BUTA $\mathcal{A}$ est l'ensemble des Σ-arbres qui admettent une exécution acceptante.

Exercice

1. Pourquoi dans le dernier exemple le langage de l'automate n'est-il pas l'ensemble des formules booléenes avec variables propositionnelles p qui sont satisfaisables ?
2. Donnez un BUTA qui reconnaisse ce langage

Déterminisation

Définition

Un BUTA déterministe (DBUTA) est un BUTA $\mathcal{A} = (Q, \Sigma, \delta, F)$ tel que pour tout $a \in \Sigma$, $\delta_a : Q^{\mathrm{ar}(a)} \to Q$

Théorème

Pour chaque BUTA $\mathcal{A}$ avec n états il existe un DBUTA $\mathcal{A}$ avec 2^n états tel que $L(\mathcal{A}) = L(\mathcal{A}')$.

Preuve :

- $Q' = 2^Q$ (l'ensemble des parties)
- $\delta'_a(M_1, \ldots, M_n) = \bigcup\{\delta_a(q_1, \ldots, q_n) \mid q_1 \in M_1, \ldots q_n \in M_n\}$
- $F' = \{M \mid M \cap F \neq \emptyset\}$

Propriétés de clôture

Théorème

La classe des langages d'arbre reconnaissables par des BUTA est close par union, intersection, et complément.

Un homomorphisme d'arbre est une fonction $h : \Sigma \to \Sigma'$ qui préserve l'arité

L'image $h(t)$ par l'homomorphisme h de l'arbre (T, t) est l'arbre (T, t') où $t'(w) = h(t(w))$ pour tout $w \in T$. Autrement dit, on substitue chaque étiquette de noeud par son image par h.

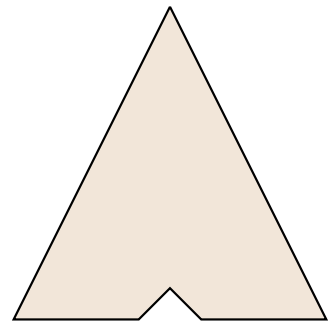
Théorème

La classe des langages d'arbre reconnus par des BUTA est close par homomorphisme.

Contexte

Définition

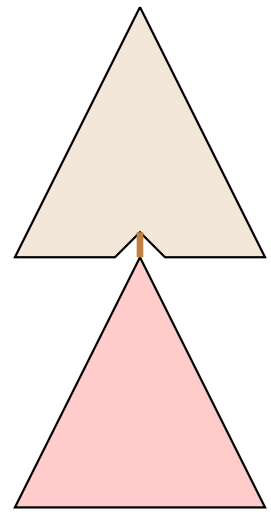
Un context est un tuple $C = (T, t, w)$ où (T, t) est un arbre, et $w \in T$ est une feuille.



Application de contexte

Soit (T', t') un arbre et $C = (T, t, w)$ un contexte. L'arbre $C[t]$ est l'arbre (T'', t'') tel que

- $T'' = T \cup w \cdot T'$
- $t''(u) = t(u)$ pour tout $u \in T \setminus \{w\}$
- $t''(w.u) = t'(u)$ pour tout $u \in T'$



Lemme de pompage

Théorème

Soit L un langage d'arbre reconnaissable par un BUTA. Alors il existe $n \geq 0$, tel que pour tout $t \in L$ de hauteur $> n$, il existe des contextes C_1, C_2 et un arbre t' tels que

1. $t = C_1[C_2[t']]$
2. pour tout $k \geq 0$, $C_1[\underbrace{C_2[\ldots[C_2[}_{k \text{ fois}} t']]\ldots]] \in L$.

Preuve : n est le nombre d'états de l'automate. Sur une branche de longueur $> n$ au moins un état est répété.

Automates d'arbre descendant

Definition

Un automate d'arbre descendant (TDTA) est un tuple $\mathcal{A} = (Q, \Sigma, \delta, q_I)$ où

- Q est un ensemble fini d'états
- Σ est un alphabet avec arités
- $\delta = \{\delta_a \mid a \in \Sigma\}$ sont les fonctions de transition, où $\delta_a : Q \to \mathcal{P}(Q^{\mathrm{ar}(a)})$
- $q_I \in Q$ est l'état initial

Exécution acceptante

Une exécution acceptante de $\mathcal{A}$ sur le Σ-arbre (T, t) est un Q-arbre (T, q) tel que

1. $q(\epsilon) = q_I$, et
2. pour tout $w \in T$, si $t(w) = a$ et $\mathrm{ar}(a) = n$, alors
$$\big(q(w \cdot 0), \ldots, q(w \cdot (n-1))\big) \in \delta_a(q(w)).$$

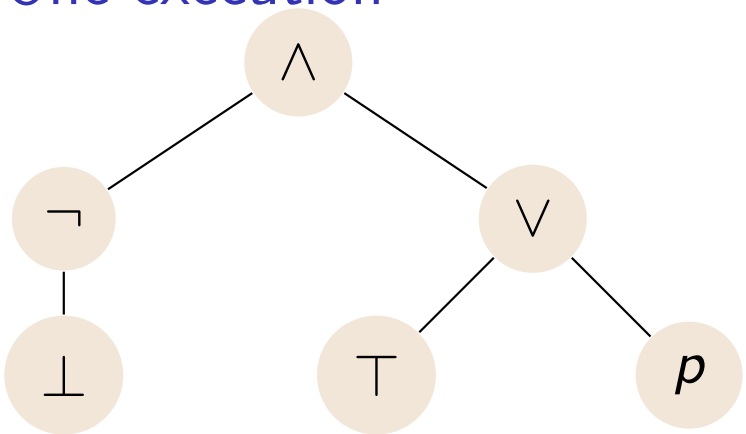
Exemple

Un TDBA

$$Q = \{0, 1\} \qquad \Sigma = \{\top, \bot, \vee, \wedge, \neg, p\} \qquad q_0 = 1$$

	$\top$	$\bot$	$\vee$	$\wedge$	$\neg$	p
0	$\emptyset$	$\{()\}$	$\{(0,0)\}$	$\{(0,0),(0,1),(1,0)\}$	$\{1\}$	$\{()\}$
1	$\{()\}$	$\emptyset$	$\{(0,1),(1,0),(1,1)\}$	$\{(1,1)\}$	$\{0\}$	$\{()\}$

Une exécution



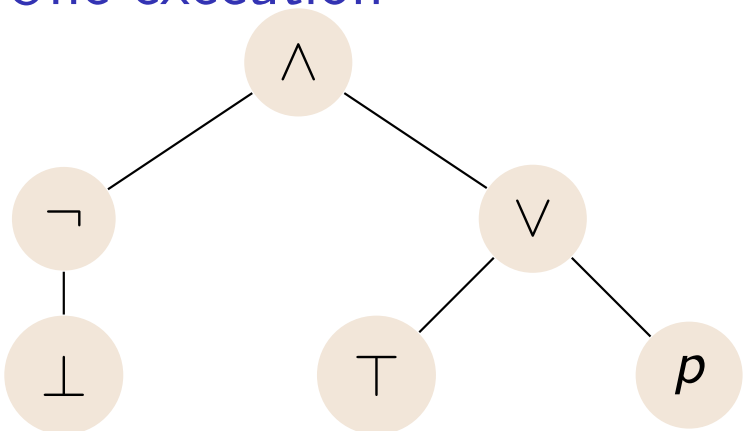
Exemple

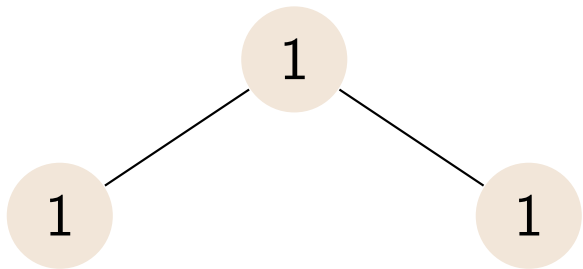
Un TDBA

$$Q = \{0, 1\} \qquad \Sigma = \{\top, \bot, \vee, \wedge, \neg, p\} \qquad q_0 = 1$$

	$\top$	$\bot$	$\vee$	$\wedge$	$\neg$	p
0	$\emptyset$	$\{()\}$	$\{(0,0)\}$	$\{(0,0),(0,1),(1,0)\}$	$\{1\}$	$\{()\}$
1	$\{()\}$	$\emptyset$	$\{(0,1),(1,0),(1,1)\}$	$\{(1,1)\}$	$\{0\}$	$\{()\}$

Une exécution





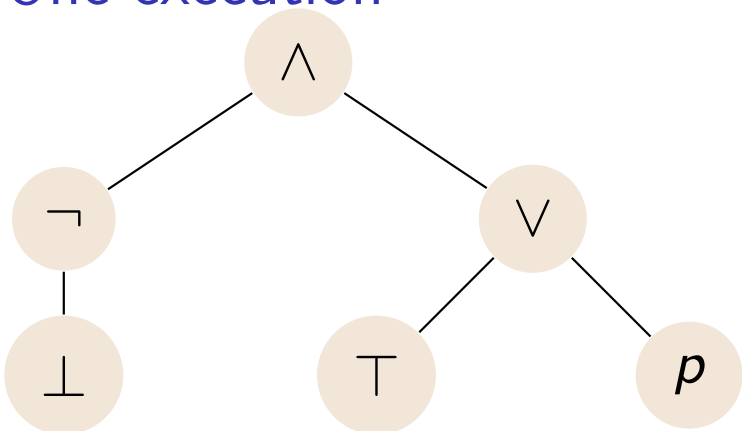
Exemple

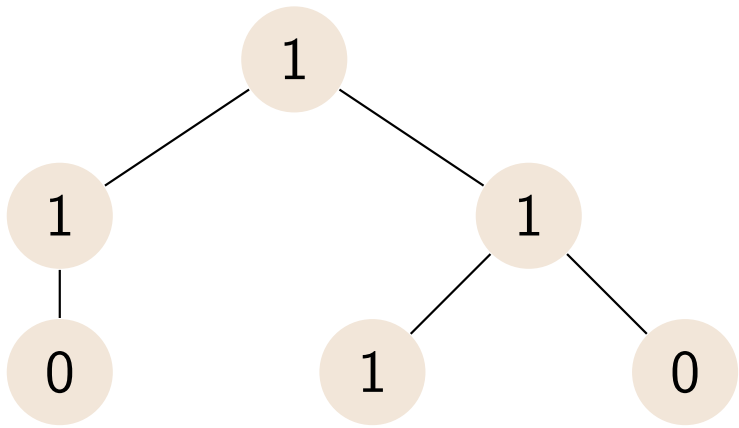
Un TDBA

$$Q = \{0, 1\} \qquad \Sigma = \{\top, \bot, \vee, \wedge, \neg, p\} \qquad q_0 = 1$$

	$\top$	$\bot$	$\vee$	$\wedge$	$\neg$	p
0	$\emptyset$	$\{()\}$	$\{(0,0)\}$	$\{(0,0),(0,1),(1,0)\}$	$\{1\}$	$\{()\}$
1	$\{()\}$	$\emptyset$	$\{(0,1),(1,0),(1,1)\}$	$\{(1,1)\}$	$\{0\}$	$\{()\}$

Une exécution





De Top-Down à Bottom-Up

Théorème

Soit $\mathcal{A}$ un TDTA avec n états. Alors il existe un BUTA $\mathcal{A}'$ avec n états qui reconnait le même langage.

Preuve

- $Q' = Q$
- $\delta'_a = \delta_a$, modulo les identités

$$Q^{\mathrm{ar}(a)} \to \mathcal{P}(Q) \quad = \quad Q \to \mathcal{P}(Q^{\mathrm{ar}(a)}) \quad = \quad \mathcal{P}(Q \times Q^{\mathrm{ar}(a)})$$

- dit autrement, $\delta'_a(q_1, \ldots q_n) = \{q \mid (q_1, \ldots, q_n) \in \delta_a(q)\}$
- $F = \{q_I\}$

De Bottom-Up à Top-Down

Satz

Soit $\mathcal{A}$ un BUTA avec n états. Alors il existe un TDTA $\mathcal{A}'$ avec $n+1$ états qui reconnait le même langage.

preuve

- $Q' = Q \uplus \{q_I\}$
- $(q_1, \ldots, q_n) \in \delta'_a(q)$ ssi $q \in \delta_a(q_1, \ldots, q_n)$ pour tout $q, q_1, \ldots, q_n \in Q$
- $\delta'_a(q_I) = \{(q_1, \ldots q_n) \mid \delta_a(q_1, \ldots, q_n) \in F\}$

Automates descendants déterministes

Définition

Un automate descendant déterministe (DTDTA) est un tuple $\mathcal{A} = (Q, \Sigma, \delta, q_I)$, où $\delta_a : Q \to \mathcal{Q}^{\mathrm{ar}(a)}$ si $\mathrm{ar}(a) \geq 1$, et $\delta_a : Q \to \{\top, \bot\}$, si $\mathrm{ar}(a) = 0$.

Exécution acceptante

Une exécution acceptante de $\mathcal{A}$ sur le Σ-arbre (T, t) est un $Q \cup \{\top, \bot\}$-arbre (T, q), tel que

1. $q(\epsilon) = q_I$, et
2. pour tout $w \in T$, si $t(w) = a$ et $\mathrm{ar}(a) = n$, alors $\delta_a(q(w)) = \big(w \cdot 0), \ldots, q(w \cdot (n-1))\big)$, et
3. pour toute feuille w, $\delta_{t(w)}(q(w)) = \top$.

DTDBA vs TDBA

Théorème

Soit L un langage d'arbres.

1. Si L est reconnu par un TDTA, alors il est aussi reconnu par un DTDTA
2. la réciproque est fausse

Preuve :

1. DTDBA$\rightarrow$ TDBA : clair
2. Soit $\Sigma = \{\top, \bot, \vee\}$. Il n'existe pas de TDTA qui reconnaisse le langage des formules qui s'évaluent à $\top$.

Problème de décision

Vacuité

On peut décider en temps $\mathcal{O}(n)$, si le langage d'un BUTA/TDTA avec n transitions est vide.

Preuve : exercice.

Problème d'universalité

On peut décider en temps $\mathcal{O}(2^n)$, si le langage d'un BUTA/TDTA avec n états est le langage de tous les arbres.

Remarque : Ce problème d'universalité est en fait EXPTIME-difficile, c'est à dire plus difficile que l'universalité d'un NFA, qui n'est $\ll$ que $\gg$ PSPACE-difficile.