THLR - Théorie des langages rationnels

Maxime Bridoux

Septembre 2020

Automate fini

NFA - Non-deterministic Finite Automaton

Un automate fini est un quintuplé $(\Sigma, Q, I, F, \delta)$ où:

- Σ est l'alphabet utilisé;
- Q est un ensemble fini d'états;
- $I \subseteq Q$ est l'ensemble des états initiaux;
- $F \subseteq Q$ est l'ensemble des états finaux;
- $\delta \subseteq Q \times \Sigma \times Q$ est l'ensemble des transitions

Vision graphique

Graphe orienté où certains noeuds sont marqués comme initiaux et/ou finaux et où les arcs sont étiquetés par des lettres

Calcul

Transitions

Soit une transition $(q, a, r) \in \delta$, alors:

- q est l'origine
- a est l'étiquette
- r est la destination

Calcul dans un automate

Un calcul dans un automate fini A est une suite de transitions $e_1 \dots e_n$ telle que pour tout $e_i = (p_1, a_1, q_1)$ et $e_{i+1} = (p_2, a_2, q_2)$, on a $q_1 = p_2$.

Reconnaissance

Calcul réussi

Un calcul $e_1 \dots e_n$ dans un automate fini A est réussi si e_1 a pour origine un état initial de A et e_n a pour destination un état final de A

Langage reconnu

Le langage reconnu par un automate fini A, noté $L(A)$, est l'ensemble des étiquettes des calculs réussis

Étapes de calcul

Étape élémentaire

On peut réécrire une étape de calcul utilisant la transition (q, a, r) sous la forme $(q, av) \vdash_A (r, v)$ où v est un mot quelconque

Clôture

On note $\vdash_A^*$ la clôture réflexive et transitive de $\vdash_A$ i.e. $(q_1, u_1 \dots u_n v) \vdash_A^* (q_n, v)$ s'il existe des états $q_1 \dots q_n$ tels que $(q_1, u_1 \dots u_n v) \vdash_A (q_2, u_2 \dots u_n v) \vdash_A \cdots \vdash_A (q_n, v)$

Reconnaissance

$$L(A) = \{u \in \Sigma^* \mid (q_0, u) \vdash_A^* (q, \epsilon) \text{ où } q_0 \in I \text{ et } q \in F\}$$

Reconnaissance

Langages reconnaissables

Un langage L appartient à $Rec(\Sigma)$ l'ensemble des langages reconnaissables par un automate fini i.e. il existe un automate A sur Σ qui reconnaît le langage L.

Automates équivalents

Deux automates A_1 et A_2 sont équivalents s'il reconnaissent le même langage i.e. $L(A_1) = L(A_2)$

Reconnaissance

Langages reconnaissables

Un langage L appartient à $Rec(\Sigma)$ l'ensemble des langages reconnaissables par un automate fini i.e. il existe un automate A sur Σ qui reconnaît le langage L.

Automates équivalents

Deux automates A_1 et A_2 sont équivalents s'il reconnaissent le même langage i.e. $L(A_1) = L(A_2)$

Objectifs

- Montrer que $Rec(\Sigma) = Rat(\Sigma^*)$ (théorème de Kleene)
- Minimiser la taille d'un automate en trouvant un équivalent de taille minimale

Automate complet

Définition

Un automate fini $A = (\Sigma, Q, I, F, \delta)$ est complet si pour tout $q \in Q$ et $a \in \Sigma$ il existe $r \in Q$ tel que $(q, a, r) \in \delta$

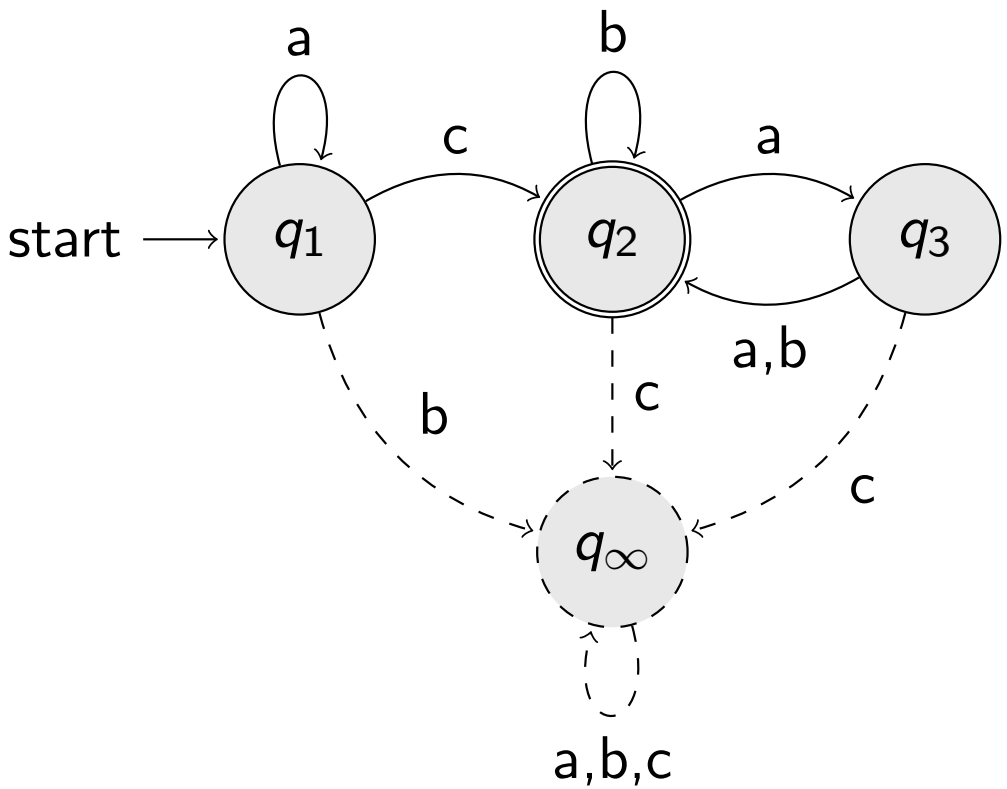
Complétion

Ajout d'un état puits pour créer les transitions manquantes

Compléter ne coûte rien

Tout automate est équivalent à un automate complet

Automate complet



États utiles

Définitions

- Un état q de A est accessible s'il existe un chemin dans A de i à q avec $i \in I$
- Un état q de A est co-accessible s'il existe un chemin dans A de q à f avec $f \in F$
- Un état q de A est utile s'il est accessible et co-accessible

Automates

Un automate A est dit accessible (resp. co-accessible, émondé) si tous ses états sont accessibles (resp. co-accessibles, utiles)

Émonder ne coûte rien

Tout automate est équivalent à un automate émondé

États utiles

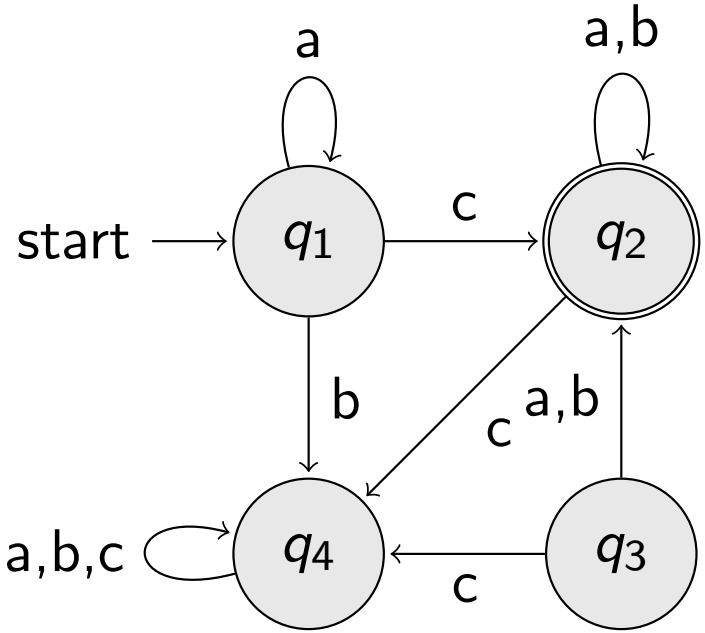


Figure: Automate initial complet

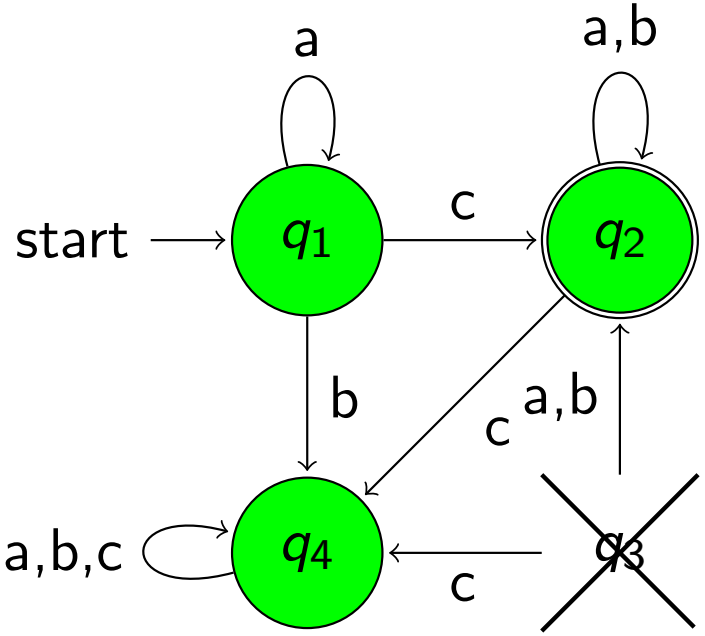


Figure: États accessibles

États utiles

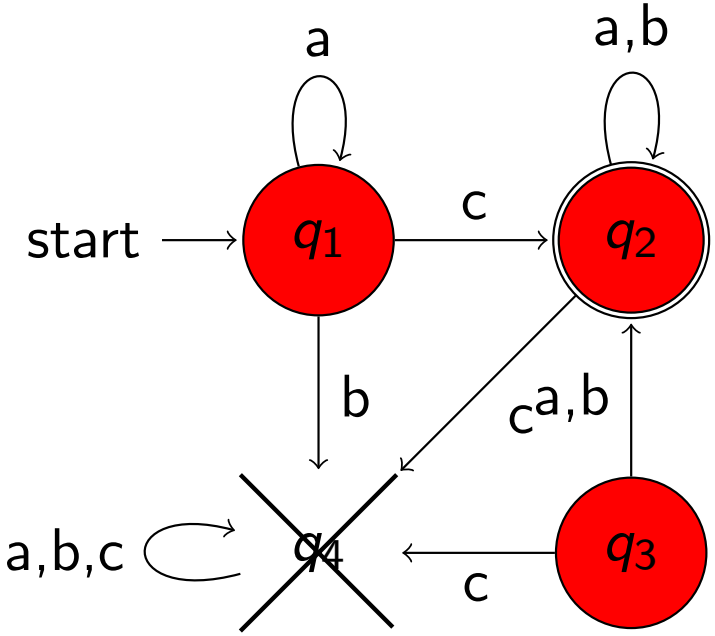


Figure: États co-accessibles

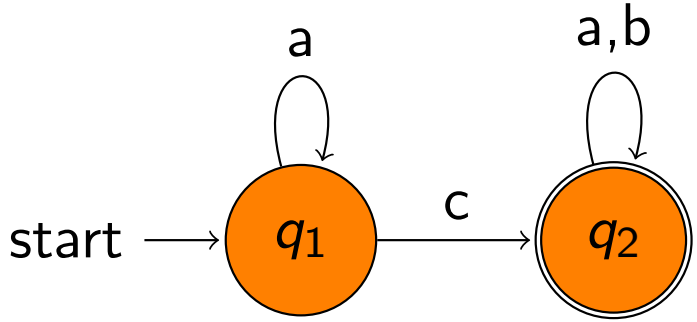


Figure: Automate final emmondé

Automate normalisé

Définition

Un automate A est dit normalisé s'il possède:

- un unique état initial qui n'est l'arrivée d'aucune transition
- un unique état final qui n'est le départ d'aucune transition

Transitions spontanées

ϵ-NFA

Un automate avec transitions spontanées (ou ϵ-transition) est un automate où $\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Q$

Calcul

Il est possible de changer d'état sans consommer de symbole à la lecture d'un mot

Les ϵ-transitions n'augmentent pas l'expressivité

On peut transformer tout automate avec ϵ-transitions en en automate équivalent sans ϵ-transitions

Transformation

Fermeture

On appelle ϵ-fermeture d'un état q l'ensemble

$$\epsilon\text{-closure}(q) = \{p \in Q \mid (q, v) \vdash_A^* (p, v)\}$$

On note alors $q \xrightarrow{\epsilon^*} p$ lorsque $p \in \epsilon\text{-closure}(q)$

Automate sans ϵ-transition (fermeture arrière)

Soit un automate avec ϵ-transition $A = (\Sigma, Q, I, F, \delta)$, on pose $A' = (\Sigma, Q, I, F', \delta')$ où :

- $F' = \{q \in Q \mid \epsilon\text{-closure}(q) \cap F \neq \emptyset\}$
- $\delta' = \{(p, a, q) \mid \exists p' \in Q, p \xrightarrow{\epsilon^*} p', (p', a, q) \in \delta\}$

On a alors $L(A) = L(A')$

Algorithme de Thompson

Rappel

- $\emptyset$ et a pour $a \in \Sigma$ sont des expressions rationnelles;
- si e_1 et e_2 sont des langages rationnels, alors $(e_1 + e_2)$, $(e_1 e_2)$, (e_1^*) sont des expressions rationnelles