

1. Introduction

Bruno Grenet

Université Grenoble Alpes – IM²AG
L3 Informatique
UE Modèles de calcul – Machines de Turing



<https://membres-ljk.imag.fr/Bruno.Grenet/MCAL-MT.html>

Contents

1. Informations générales

2. Introduction

3. Machines de Turing

Contents

1. Informations générales

2. Introduction

3. Machines de Turing

Présentation

Nom : Bruno Grenet

Email : ...@univ-grenoble-alpes.fr

Insérer [MCAL-MT] en sujet

Page web : <https://membres-ljk.imag.fr/Bruno.Grenet/>

Affiliation : Université Grenoble Alpes

UFR IM²AG

Laboratoire Jean Kuntzmann (LJK)

Téléphone : +33 457 421 721

Bureau : Bâtiment IMAG (150 pl. du Torrent); 1^{er} étage ; bureau 106

Organisation

Équipe pédagogique

Cours: Bruno Grenet

TD G1: Linda Gutsche

TD G2: Thomas Rahab-Lacroix

TD G3: Linda Gutsche

TD G4: Bruno Grenet

Contacts: <Prenom>. <Nom>@univ-grenoble-alpes.fr

Planning sur huit semaines

Cours: 1 séance par semaine

TD: 1 séance par semaine

Communications

- ▶ Email + mattermost
- ▶ Consultez ADE!

Évaluations

EC_{MT}: Évaluation continue (*quick*)

Durée: 1h

Date: lundi 9 février 9h45

$$EC = \frac{1}{2}(EC_{MT} + EC_{LC})$$

Coefficient: 0,4

Contenu: cours/TD 1 à 4

ET_{MT}: Évaluation terminale

Durée: 1h

Date: jeudi 23 avril (?)

$$ET = \frac{1}{2}(ET_{MT} + ET_{LC})$$

Coefficient: 0,6

Contenu: tout

ES_{MT}: Évaluation supplémentaire

- ▶ Similaire à l'évaluation terminale
- ▶ mardi 23 juin (?)

$$ES = \frac{1}{2}(ES_{MT} + ES_{LC})$$

Note finale

commune avec MCAL-LC

$$\text{Session 1: } \max(ET; 0,4 \cdot EC + 0,6 \cdot ET)$$

$$\text{Session 2: } \max(ES; 0,4 \cdot EC + 0,6 \cdot ES)$$

Page du cours



<https://membres-ljk.imag.fr/Bruno.Grenet/MCAL-MT.html>

Contenu

- ▶ Diapositives de cours annotées, sujets de TD, annales
- ▶ Code et exemples
- ▶ Bibliographie :
 - ▶ livres, notes de cours
 - ▶ liens : vidéos, articles de vulgarisation, sites web, ...

→ **Très utile à consulter !**

Contents

1. Informations générales

2. Introduction

3. Machines de Turing

De quoi va-t-on parler ?



Résumé (personnel) de la vidéo

Origines

- ▶ Mécanisation du raisonnement ? Leibniz (xvii^e siècle)
- ▶ *Entscheidungsproblem*: existe-t-il un algorithme pour décider si un énoncé mathématique est correct ? Hilbert & Ackermann (1928)
- ▶ Années 1930 : intuition de l'absence d'algorithme → comment la prouver ?

Formalisation des algorithmes

- ▶ Fonctions μ -récursives Herbrand-Gödel (1931-34)
- ▶ λ -calcul Kleene-Church (1932-36)
- ▶ Machines de Turing Turing (1936)

Trois définitions prouvées équivalentes → c'est la « bonne » définition !

Incalculabilité

- ▶ Pas d'algorithme pour le problème de la décision, le problème de l'arrêt, ...
- Résultats négatifs mais une des origines de l'informatique

Objectifs du cours

Machines de Turing

- ▶ Étude d'un des modèles de calcul
- ▶ (Autre modèle – λ -calcul – étudié dans l'autre partie du cours)
- ▶ Comparaison avec d'autres modèles

Calculabilité

- ▶ Qu'est-ce que veut dire *calculer* ?
- ▶ Comment démontrer l'*inexistence* d'algorithme pour un problème donné ?
- ▶ Liens avec l'informatique *de tous les jours* ?

Avant / après

- ▶ Automates et Langages (L2) ; Analyse syntaxique (L3)
 - ▶ Automates finis ; automates à pile
- ▶ Complexité Algorithmique ; Fundamentals of Computer Science (M1 Mosig / INFO)
 - ▶ Théorie de la complexité

Plan (approximatif)

1, 2. Machines de Turing et variantes

- ▶ Exemples de machines
- ▶ Définition (informelle et formelle)
- ▶ Robustesse : les variantes ne changent pas grand chose

3. Thèse de Church-Turing

- ▶ Algorithme = Machine de Turing (ou λ -calcul, ou ...)
- ▶ Équivalence entre modèles

4, 5, 6. Incalculabilité

- ▶ Il existe des fonctions non calculables (*diagonalisation*)
- ▶ Exemples concrets

7. Fonctions primitives récursives

- ▶ Et si on interdisait le `while`?

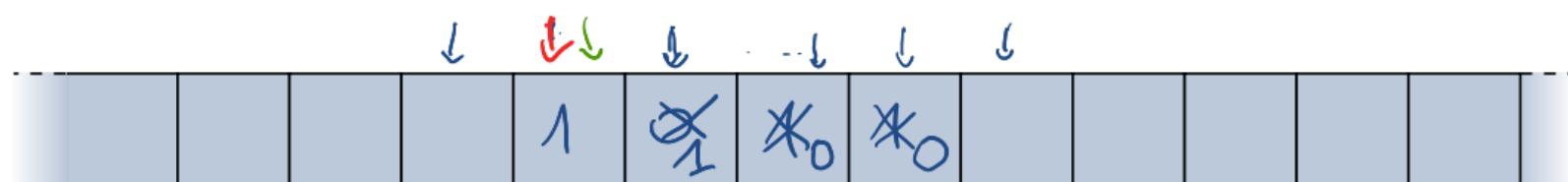
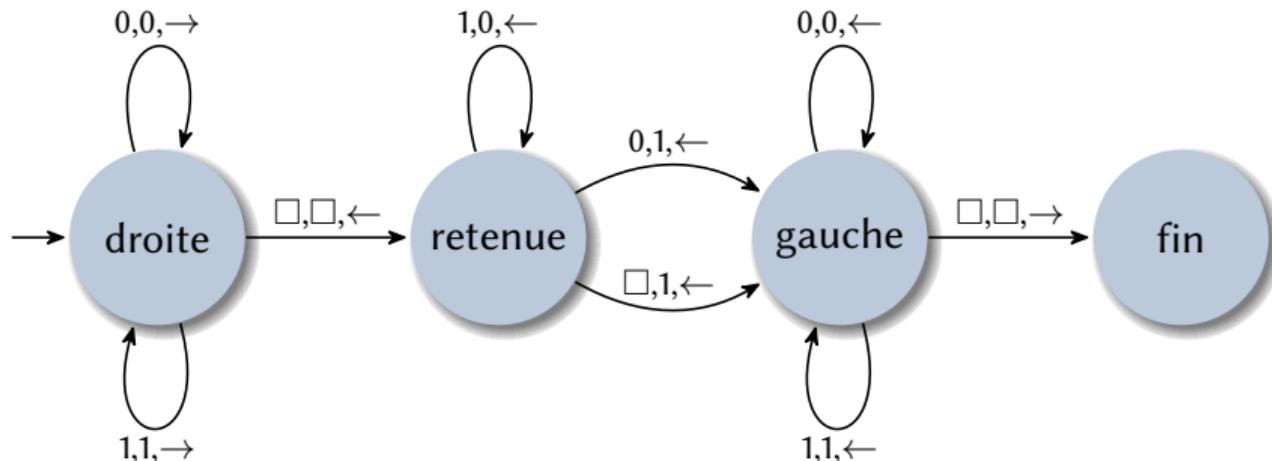
Contents

1. Informations générales

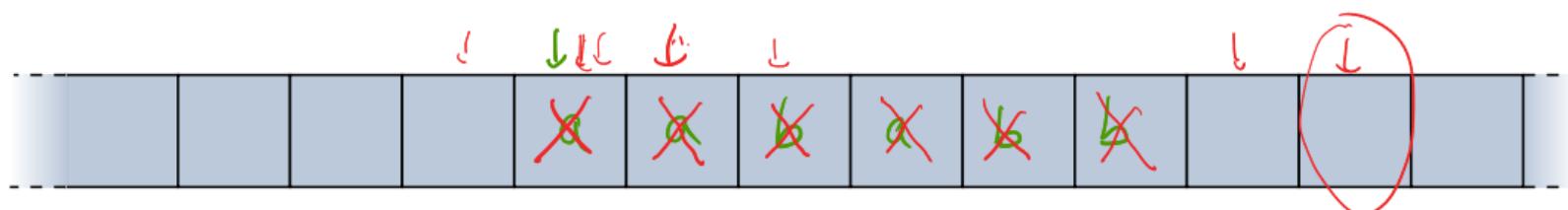
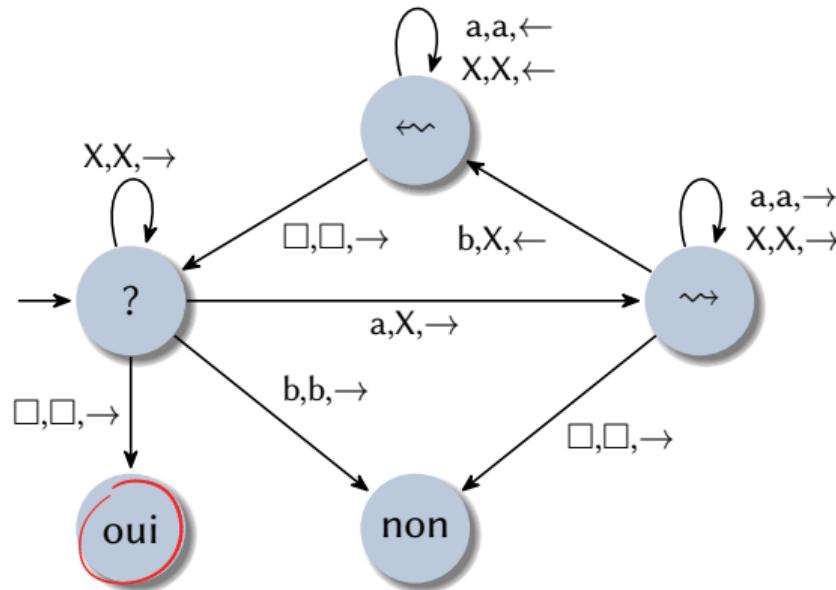
2. Introduction

3. Machines de Turing

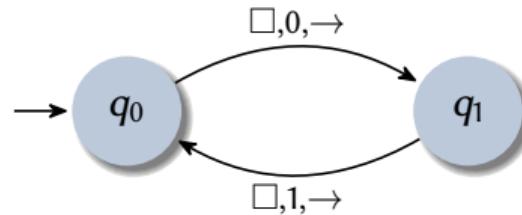
Premier exemple



Deuxième exemple

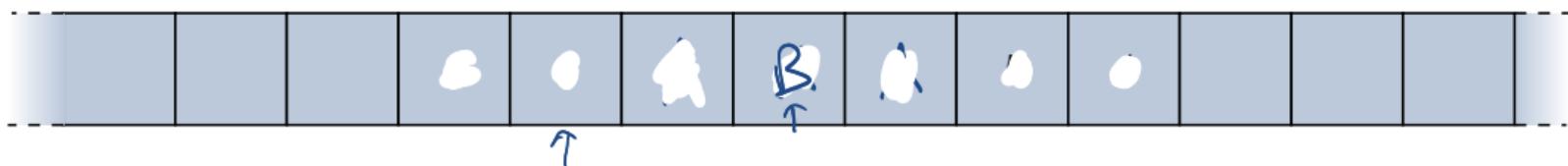
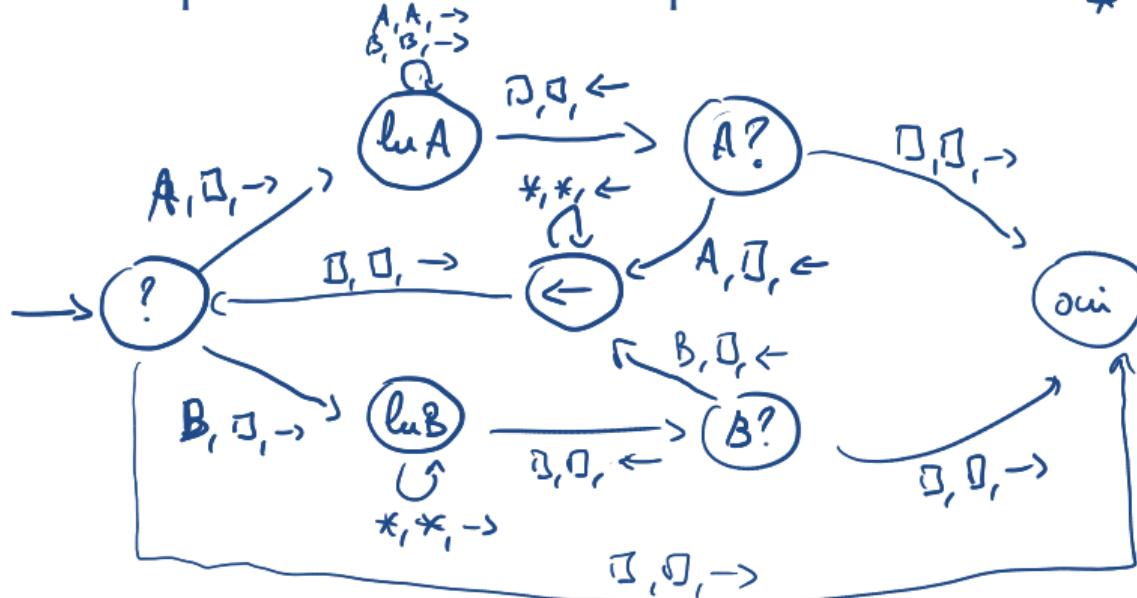


Troisième exemple



Quatrième exemple : reconnaître des palindromes

$$* = A \text{ ou } B$$



Description des machines de Turing

Constituants d'une machine de Turing

- ▶ Un ensemble d'états Q , dont un état initial q_0
- ▶ Une table de transition δ
- ▶ Un ruban constitué d'une infinité de cases
- ▶ Une tête de lecture/écriture

sommets

automate

Le ruban

- ▶ Chaque case du ruban contient un symbole d'un alphabet Σ
- ▶ L'alphabet contient un symbole spécial « blanc » \square
- ▶ Le ruban ne contient qu'un nombre fini de symboles $\neq \square$

case vide

La table de transition

- ▶ Pour certains couples $(q, x) \in Q \times \Sigma$, $\delta(q, x) = (q', y, \leftrightarrow)$ où
 - ▶ $q, q' \in Q$ sont l'ancien et le nouvel état
 - ▶ $x \in \Sigma$ est le symbole lu
 - ▶ $y \in \Sigma$ est le symbole écrit
 - ▶ $\leftrightarrow \in \{\leftarrow, \rightarrow\}$ est le déplacement
- ▶ Sinon, $\delta(q, x)$ est indéfini

Formalisation : machine de Turing

Définition

Une machine de Turing est un quadruplet $\mathcal{M} = (Q, \Sigma, q_0, \delta)$ où

Q est un ensemble fini d'états

Σ est l'*alphabet* contenant le blanc \square

$q_0 \in Q$ est l'*état initial*

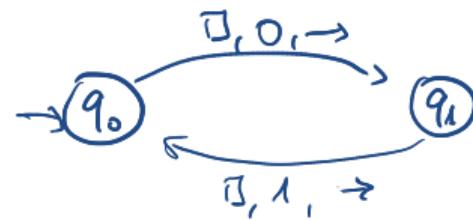
$\delta : Q \times \Sigma \rightarrow Q \times \Sigma \times \{\rightarrow, \leftarrow\}$ est la *table de transition*

Remarques

- ▶ Ruban implicite et *non borné* plutôt qu'infini
- ▶ Table de transition : fonction *partielle* $\rightarrow \delta(q, x)$ peut être *indéfini*
- ▶ Il existe d'autres formalisations équivalentes

Exemple

- ▶ $Q = \{q_0, q_1\}$
- ▶ $\Sigma = \{\square, 0, 1\}$
- ▶ $\delta(q_0, \square) = (q_1, 0, \rightarrow), \delta(q_1, \square) = (q_0, 1, \rightarrow)$



Formalisation : machine de Turing

Définition

Une machine de Turing est un quadruplet $\mathcal{M} = (Q, \Sigma, q_0, \delta)$ où

Q est un ensemble fini d'états

Σ est l'*alphabet* contenant le blanc \square

$q_0 \in Q$ est l'*état initial*

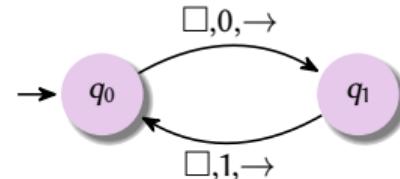
$\delta : Q \times \Sigma \rightarrow Q \times \Sigma \times \{\rightarrow, \leftarrow\}$ est la *table de transition*

Remarques

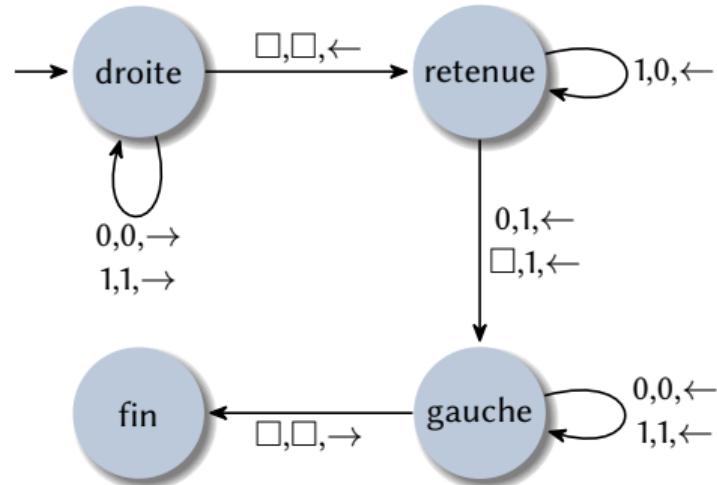
- ▶ Ruban implicite et *non borné* plutôt qu'infini
- ▶ Table de transition : fonction *partielle* $\rightarrow \delta(q, x)$ peut être *indéfini*
- ▶ Il existe d'autres formalisations équivalentes

Exemple

- ▶ $Q = \{q_0, q_1\}$
- ▶ $\Sigma = \{\square, 0, 1\}$
- ▶ $\delta(q_0, \square) = (q_1, 0, \rightarrow), \delta(q_1, \square) = (q_0, 1, \rightarrow)$



Représentation de la table de transition



Etat	Lu	Écrit	Dépl.	Nv. état
droite	0	0	→	droite
droite	1	1	→	droite
droite	□	□	←	retenue

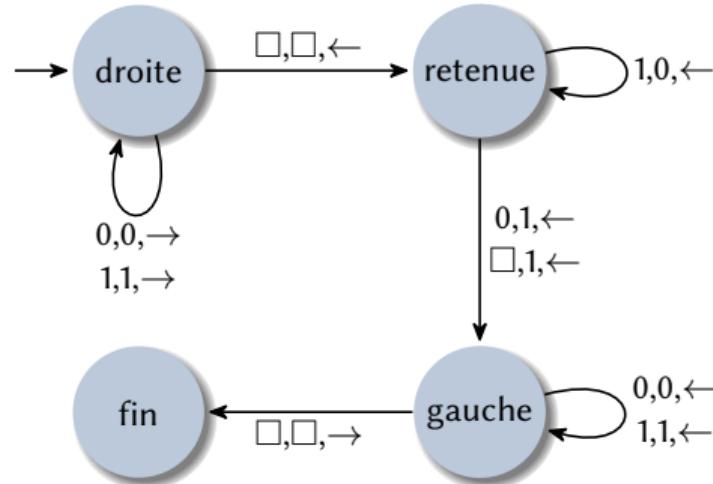
retenue	1	0	←	retenue
retenue	0	1	←	gauche
retenue	□	1	←	gauche

gauche	0	0	←	gauche
gauche	1	1	←	gauche
gauche	□	□	→	fin

- Le tableau de δ suffit à représenter \mathcal{M} :
- Q : liste des états dans la table
- Σ : liste des symboles dans la table
- q_0 : 1^{er} état de la table (par convention)

autre formalisation

Représentation de la table de transition



État	Lu	Écrit	Dépl.	Nv. état
droite	0	0	→	droite
	1	1	→	droite
	□	□	←	retenue
retenue	1	0	←	retenue
	0	1	←	gauche
	□	1	←	gauche
gauche	0	0	←	gauche
	1	1	←	gauche
	□	□	→	fin

- Le tableau de δ suffit à représenter \mathcal{M} :
- Q : liste des états dans la table
- Σ : liste des symboles dans la table
- q_0 : 1^{er} état de la table (par convention)

autre formalisation

Fonctionnement d'une machine de Turing

Une étape de calcul

Machine dans l'état q :

1. Lecture de la lettre x écrite sur le ruban
2. Si $\delta(q, x)$ n'est pas défini : arrêt (définitif) de la machine
3. Si $\delta(q, x) = (q', y, \leftrightarrow)$:
 - i. écriture de y à la place de x
 - ii. déplacement de la tête à gauche (\leftarrow) ou à droite (\rightarrow)
 - iii. nouvel état : q'

Calcul complet

- ▶ *Configuration initiale* : état q_0 , entrée w sur le ruban, tête sur la 1^{ère} lettre de w
- ▶ Tant que c'est possible, appliquer une étape de calcul
 - ▶ Soit la machine se bloque : calcul terminé $\delta(q, x)$ indéfini
 - ▶ Sinon : la machine *boucle*
- ▶ Remarque : pas de *sémantique* → reconnaissance de langage, calcul de fonction, ...

Concept important : configurations

Configuration : état global de la machine à un instant donné

Définition

- ▶ Configuration $C = (q, w, i)$:
 - ▶ état q
 - ▶ mot w sur le ruban
 - ▶ position i de la tête sur w
- ▶ Représentation graphique: $C = q : w_{[0]} \cdots \check{w}_{[i]} \cdots w_{[n-1]}$ $i < 0$ ou $i \geq |w|$: tête hors de w
- ▶ Configuration initiale $C_0 = q_0 : \check{w}_{[0]} w_{[1]} \cdots w_{[n-1]}$ $\check{\cdot}$: position de la tête
 w : entrée

Exemples

D: 101č0



q: Xabb



0: 01č



=		I	O	I		O		=
=	X	a	b	b				=
=		O	A					=

Suites de configurations

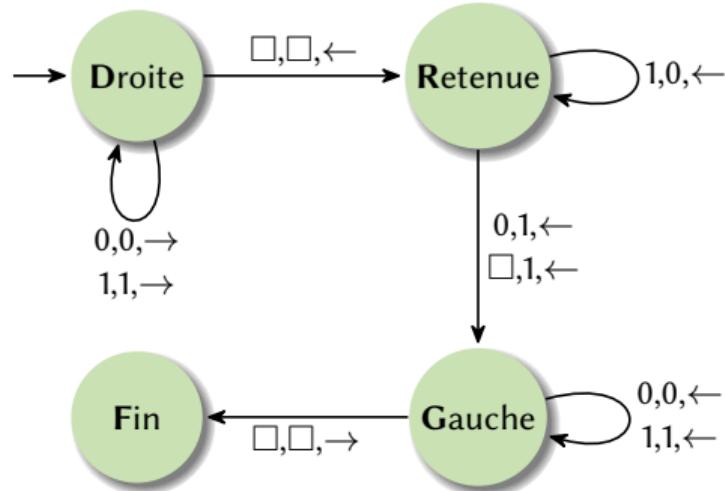
Notations

- ▶ $C \vdash C'$ si on passe de C à C' en une étape de calcul
- ▶ $C \vdash^* C'$ si on passe de C à C' en une ou plusieurs étapes de calcul
 - ▶ $C = C_1 \vdash C_2 \vdash \dots \vdash C_{k-1} \vdash C_k = C'$
- ▶ $C \downarrow$ si aucune règle de la table de transition ne s'applique à C (calcul terminé)

Calcul d'une machine

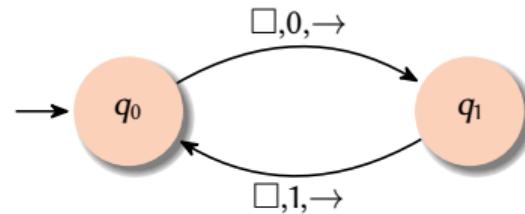
- ▶ Deux possibilités :
 - ▶ si la machine boucle : suite infinie $C_0 \vdash C_1 \vdash C_2 \vdash \dots$
 - ▶ sinon : suite finie $C_0 \vdash C_1 \vdash \dots \vdash C_k \downarrow$
- ▶ Résultat du calcul de \mathcal{M} sur w :
 - ▶ $\mathcal{M}(w) = C_k$ si $C_0 \vdash^* C_k \downarrow$ *configuration finale*
 - ▶ Notation $\mathcal{M}(w) \uparrow$ si la suite est infinie aucune configuration finale

Exemples



$D : \check{1}011$
 $\vdash D : 1\check{0}11$
 $\vdash D : 10\check{1}1$
 $\vdash D : 101\check{1}$
 $\vdash D : 1011\check{1}$
 $\vdash R : 10\check{1}1$

$\vdash R : 10\check{1}0$
 $\vdash R : 1\check{0}00$
 $\vdash G : \check{1}100$
 $\vdash G : \check{\square}1100$
 $\vdash F : \check{1}100 \downarrow$



$q_0 : \check{\square}$
 $\vdash q_1 : 0\check{\square}$
 $\vdash q_0 : 01\check{\square}$
 $\vdash q_1 : 010\check{\square}$
 $\vdash q_0 : 0101\check{\square}$
 \vdots

Utilisation d'une machine de Turing

Reconnaissance de langage

- ▶ Hypothèse : \mathcal{M} contient un état spécial : **oui**
- ▶ Langage $L(\mathcal{M})$ **reconnu** par \mathcal{M} :
 $w \in L(\mathcal{M})$ **ssi** le calcul termine et l'état final est l'état « oui »

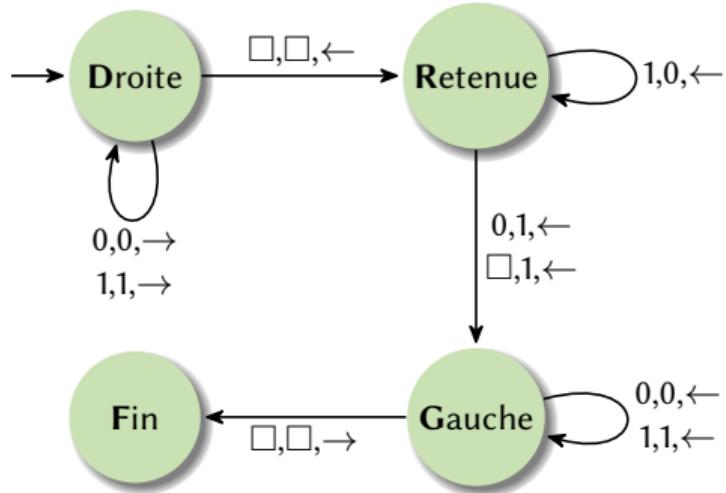
Calcul de fonction

- ▶ Fonction $f_{\mathcal{M}}$ **calculée** par \mathcal{M} :
 $f_{\mathcal{M}}(w) = w'$ si le calcul termine, et que le mot w' est écrit sur le ruban
- ▶ $f_{\mathcal{M}}(w)$ est indéfini si $\mathcal{M}(w) \uparrow$, noté $f_{\mathcal{M}}(w) \uparrow$

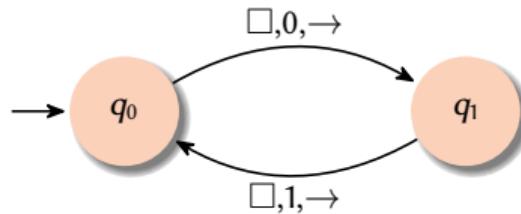
Remarques

- ▶ Autres conventions possibles
 - ▶ États d'acceptation et de rejet
 - ▶ Prise en compte de l'état final et du contenu final du ruban en même temps
- ▶ Définition possible d'**énumération** (infinie)

Exemples de fonctions

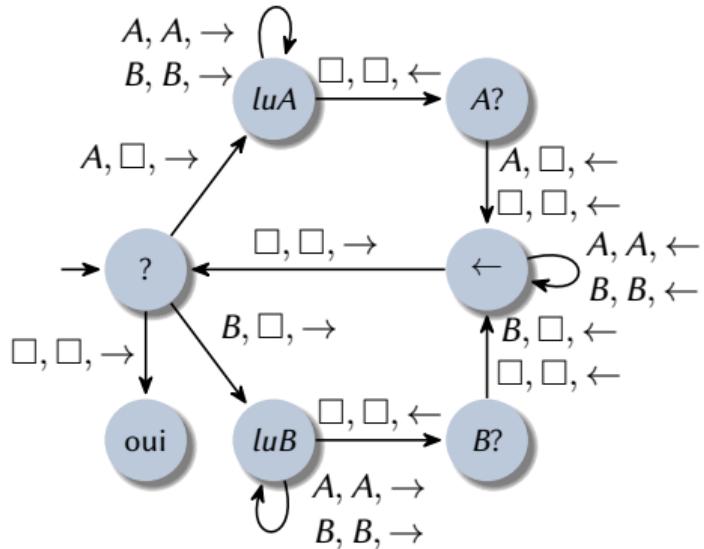


$$f_\pi(x) = x + 1 \text{ (en binaire)}$$

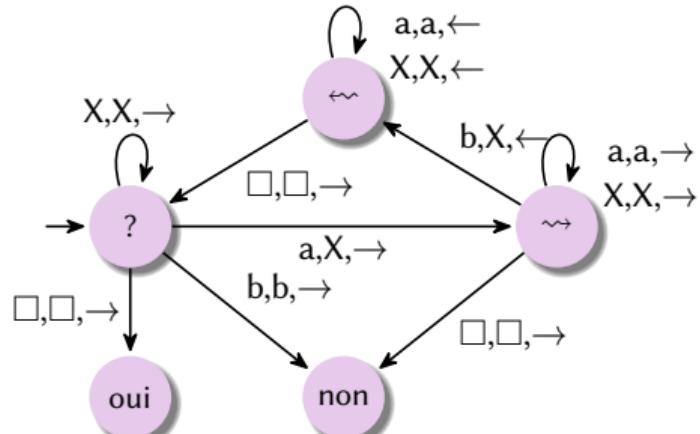


$$f_q(w) = \begin{cases} w & \text{si } |w| > 0 \\ \text{indefini} & \text{si } |w| = 0 \end{cases}$$

Exemples de langages

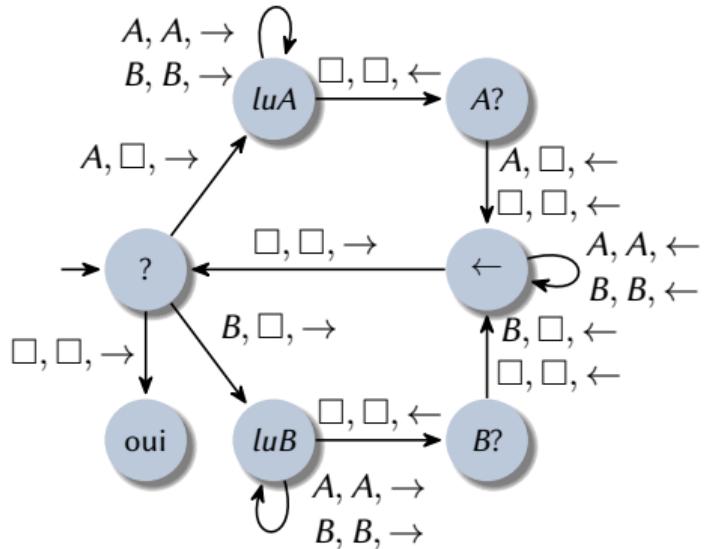


$$L(AB) = \left\{ w \in \{A, B\}^*: w \text{ est un palindrome} \right\}$$

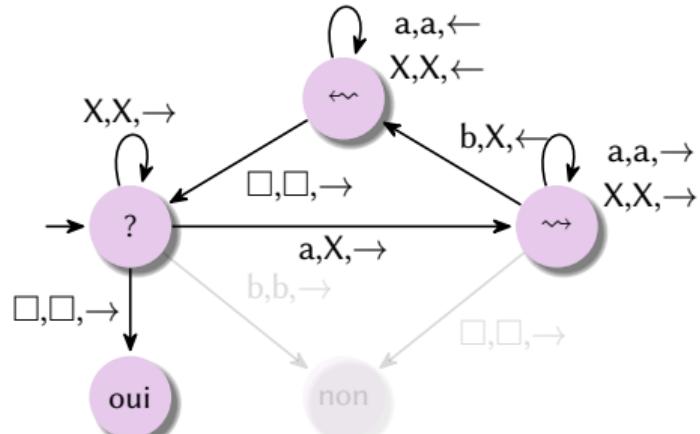


$$L(BA) = \left\{ w \in \{a, b\}^*: w \text{ est bien parenthésé} \right\}$$

Exemples de langages



$$L(\text{pal}) = \left\{ w \in \{A, B\}^*: w \text{ est un palindrome} \right\}$$



$$L(\text{bal}) = \left\{ w \in \{a, b\}^*: w \text{ est bien parenthésé} \right\}$$

Conclusion

Machine de Turing

- ▶ Extension des automates avec un ruban non borné et une table de transition
- ▶ Définitions formelles :
 - ▶ Machine de Turing : $\mathcal{M} = (Q, \Sigma, q_0, \delta)$
 - ▶ Table de transition : δ = tableau à cinq colonnes $(q, x, q', y, \leftrightarrow)$
 - ▶ Configuration : $C = (q, w, i)$
 - ▶ Calcul : $C_0 \vdash C_1 \vdash C_2 \vdash \dots$

$q : w_{[0]} \cdots \check{w}_{[i]} \cdots w_{[n-1]}$

Utilisation des machines de Turing

- ▶ Reconnaissance : $L(\mathcal{M}) = \{w : \text{sur l'entrée } w, \mathcal{M} \text{ termine dans l'état « oui »}\}$
- ▶ Calcul : $f_{\mathcal{M}}(w) = \text{mot sur le ruban en fin de calcul, ou } \uparrow \text{ si la machine boucle}$

À savoir faire

- ▶ Exécuter une machine (simple) sur une entrée
- ▶ Décrire le calcul effectué / le langage reconnu par une machine (simple)
- ▶ Créer une machine pour un problème simple