UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL
EPISEN

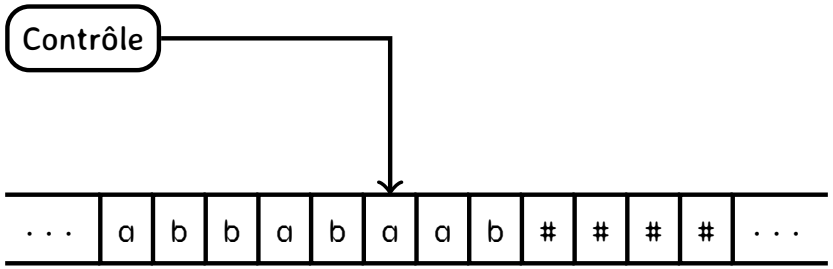
Machines de Turing

Calculabilité et Complexité

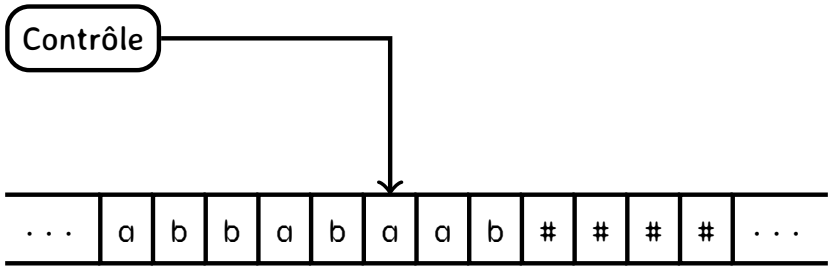
Paul Brunet

Machines de Turing
Plan

Dispositif automatique
Machine de Turing



Dispositif automatique
Machine de Turing

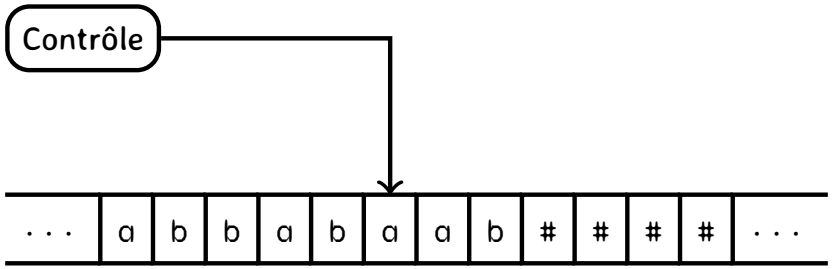


Mémoire

La mémoire d'une machine de Turing est appelée un **ruban**.
Chaque case (ou cellule) du ruban contient :

- soit un symbole tiré d'un alphabet fini (par exemple $\{0, 1\}$, $\{a, b\}$, ou bien l'alphabet `ASCII`) ;
- soit le symbole spécial #, indiquant que la case est vide.

Dispositif automatique
Machine de Turing



Mémoire

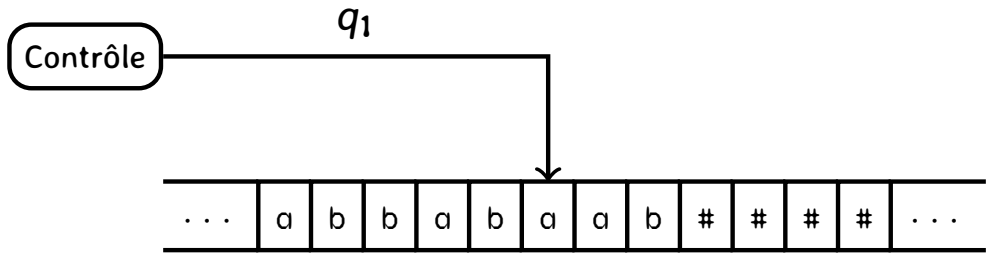
La mémoire d'une machine de Turing est appelée un **ruban**.
Chaque case (ou cellule) du ruban contient :

- soit un symbole tiré d'un alphabet fini (par exemple $\{0, 1\}$, $\{a, b\}$, ou bien l'alphabet `ASCII`) ;
- soit le symbole spécial #, indiquant que la case est vide.

Contrôle

Le "module de contrôle" contient un **ensemble fini d'états**, dont un état initial et un sous-ensemble d'états acceptants, ainsi que des règles de calcul.

Dispositif automatique
Machine de Turing



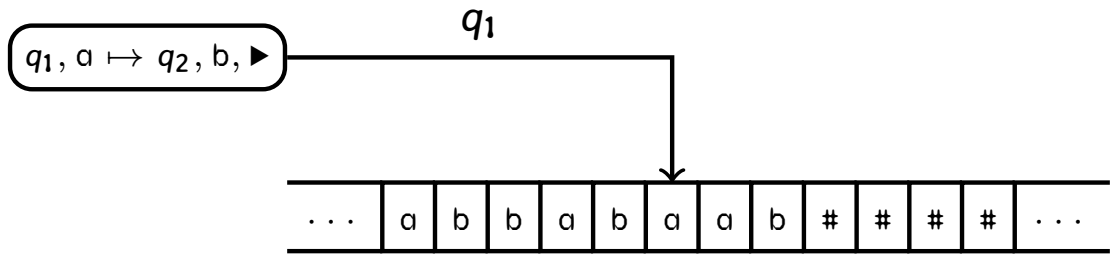
Configuration

Une configuration de la machine correspond à son état courant. Cela inclut l'état du module de contrôle, le contenu du ruban, et la position de la tête de lecture.

À chaque étape :

☞ la machine lit le symbole contenu dans la case courante ;

Dispositif automatique
Machine de Turing



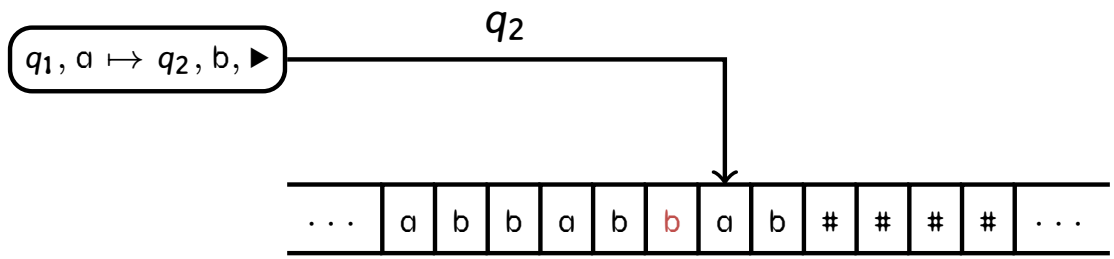
Configuration

Une configuration de la machine correspond à son état courant. Cela inclut l'état du module de contrôle, le contenu du ruban, et la position de la tête de lecture.

À chaque étape :

- la machine lit le symbole contenu dans la case courante ;
- en fonction de ce symbole et de l'état courant, si une règle s'applique :

Dispositif automatique
Machine de Turing



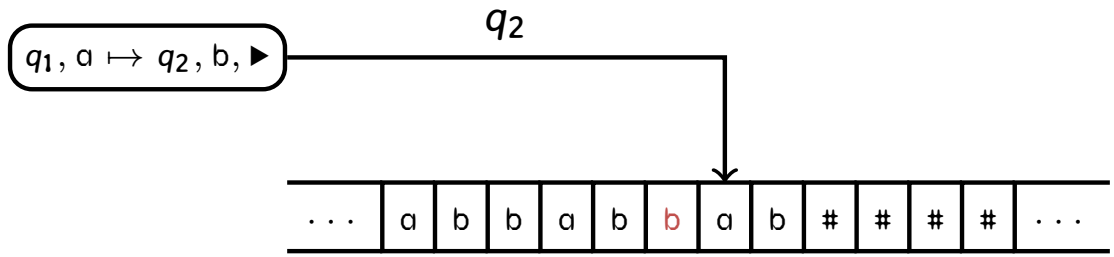
Configuration

Une configuration de la machine correspond à son état courant. Cela inclut l'état du module de contrôle, le contenu du ruban, et la position de la tête de lecture.

À chaque étape :

- la machine lit le symbole contenu dans la case courante ;
- en fonction de ce symbole et de l'état courant, si une règle s'applique :
  - on change d'état de contrôle ;

Dispositif automatique
Machine de Turing



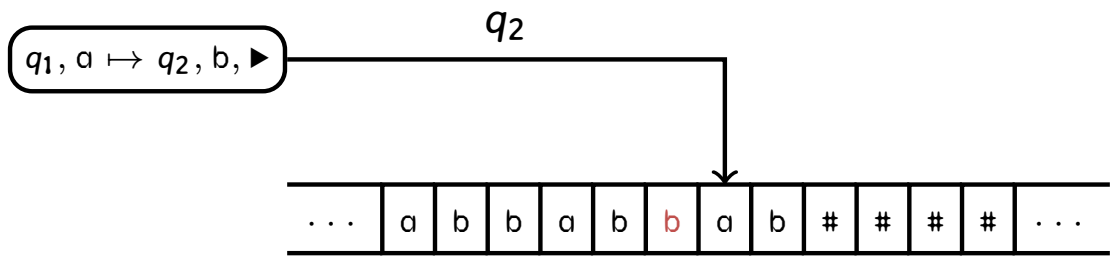
Configuration

Une configuration de la machine correspond à son état courant. Cela inclut l'état du module de contrôle, le contenu du ruban, et la position de la tête de lecture.

À chaque étape :

- la machine lit le symbole contenu dans la case courante ;
- en fonction de ce symbole et de l'état courant, si une règle s'applique :
  - on change d'état de contrôle ;
  - on écrit un nouveau symbole dans la case courante ;

Dispositif automatique
Machine de Turing



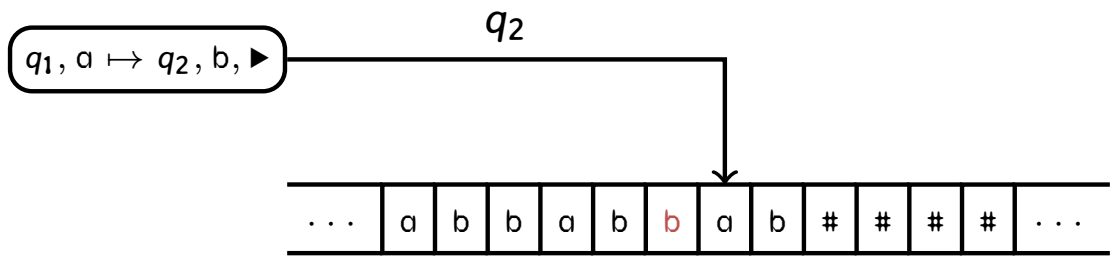
Configuration

Une configuration de la machine correspond à son état courant. Cela inclut l'état du module de contrôle, le contenu du ruban, et la position de la tête de lecture.

À chaque étape :

- la machine lit le symbole contenu dans la case courante ;
- en fonction de ce symbole et de l'état courant, si une règle s'applique :
  - on change d'état de contrôle ;
  - on écrit un nouveau symbole dans la case courante ;
  - on déplace le pointeur d'un cran, vers la droite ou vers la gauche.

Dispositif automatique
Machine de Turing



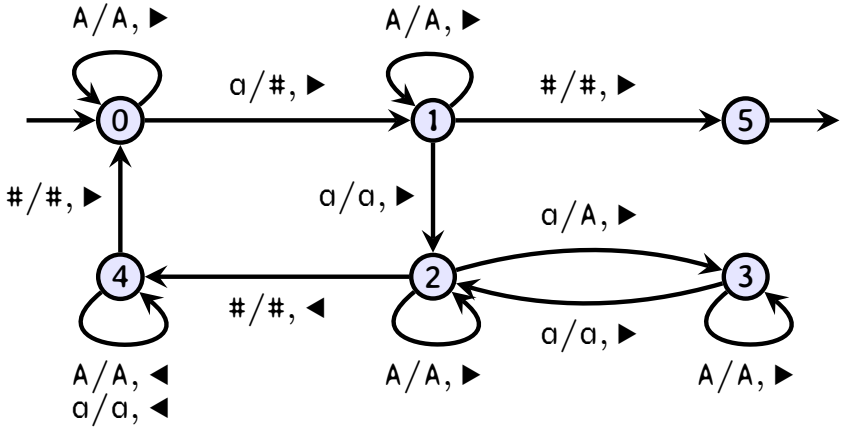
Configuration

Une configuration de la machine correspond à son état courant. Cela inclut l'état du module de contrôle, le contenu du ruban, et la position de la tête de lecture.

À chaque étape :

- la machine lit le symbole contenu dans la case courante ;
- en fonction de ce symbole et de l'état courant, si une règle s'applique :
  - on change d'état de contrôle ;
  - on écrit un nouveau symbole dans la case courante ;
  - on déplace le pointeur d'un cran, vers la droite ou vers la gauche.
- si aucune règle ne s'applique, ou si l'état courant est final, la machine s'arrête.

Machines de Turing
Définition formelle

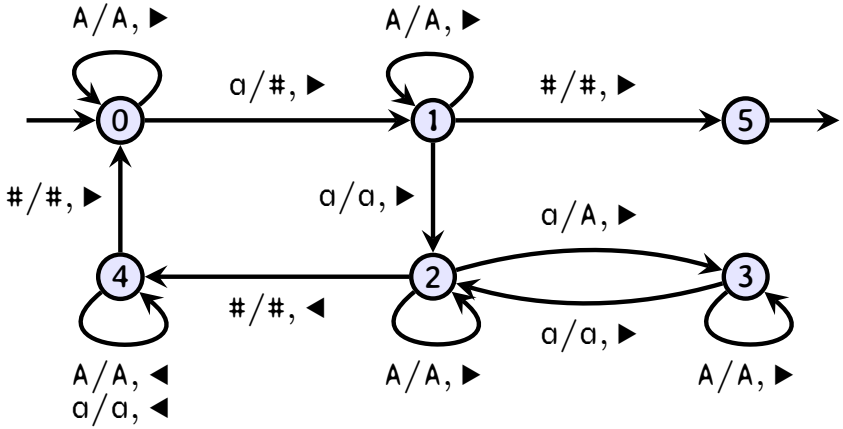


définition (Machine de Turing)

Une machine de Turing est une structure $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ où :

☞ Q est un ensemble fini d'états ;

Machines de Turing
Définition formelle

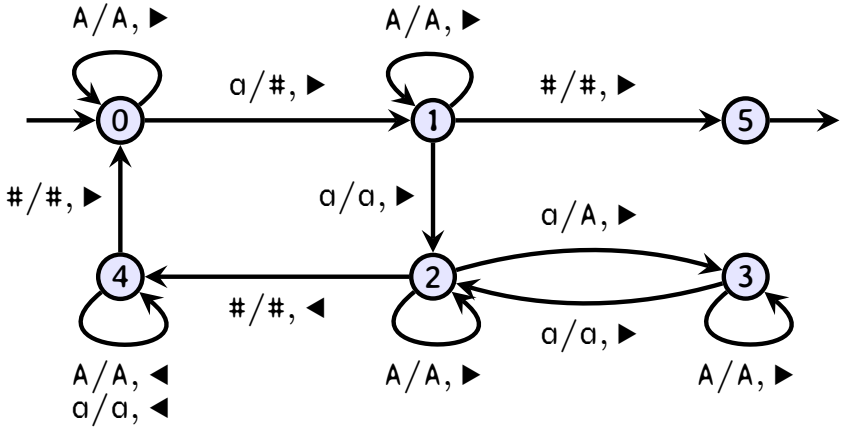


définition (Machine de Turing)

Une machine de Turing est une structure $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ où :

- Q est un ensemble fini d'états ;
- Σ est un alphabet d'entrée ;

Machines de Turing
Définition formelle

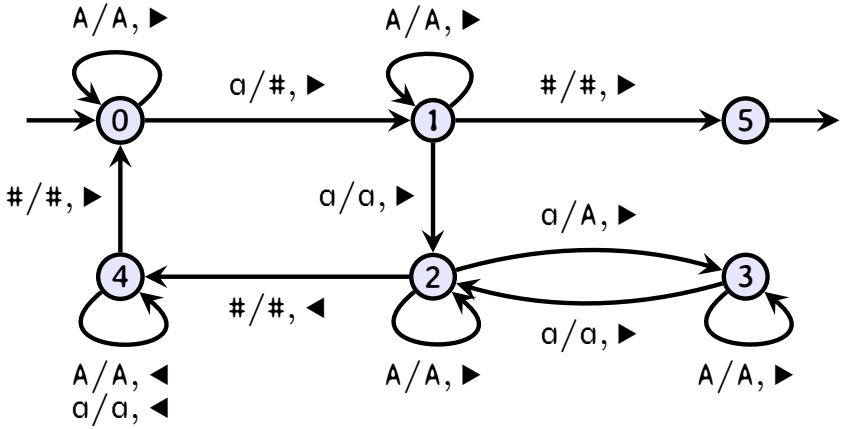


définition (Machine de Turing)

Une machine de Turing est une structure $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ où :

- Q est un ensemble fini d'états ;
- Σ est un alphabet d'entrée ;
- Γ est un alphabet de bande, tel que $\{\#\} \cup \Sigma \subseteq \Gamma$;

Machines de Turing
Définition formelle

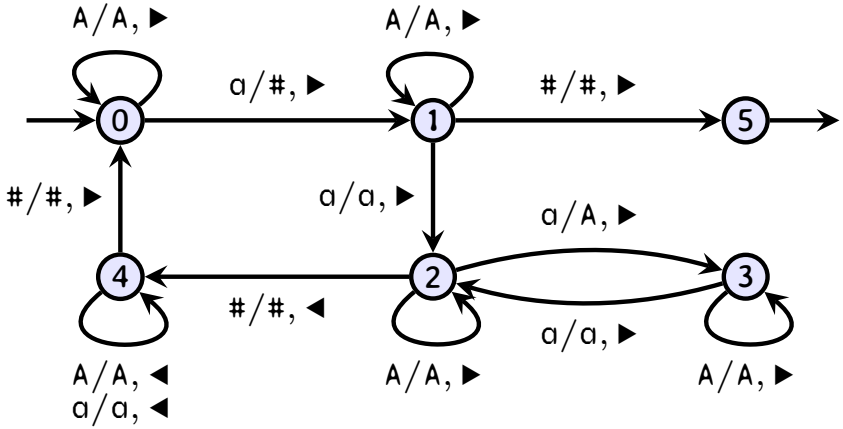


définition (Machine de Turing)

Une machine de Turing est une structure $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ où :

- Q est un ensemble fini d'états ;
- Σ est un alphabet d'entrée ;
- Γ est un alphabet de bande, tel que $\{\#\} \cup \Sigma \subseteq \Gamma$;
- $\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangleright\} \times Q$ est un ensemble de transitions ;

Machines de Turing
Définition formelle

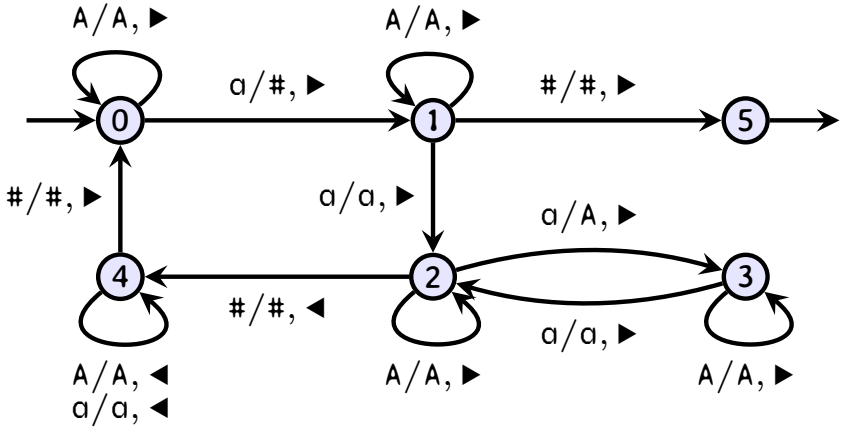


définition (Machine de Turing)

Une machine de Turing est une structure $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ où :

- Q est un ensemble fini d'états ;
- Σ est un alphabet d'entrée ;
- Γ est un alphabet de bande, tel que $\{\#\} \cup \Sigma \subseteq \Gamma$;
- $\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangleright\} \times Q$ est un ensemble de transitions ;
- $q_0 \in Q$ est un état initial, et $F \subseteq Q$ est un ensemble d'états finaux.

Machines de Turing
Définition formelle



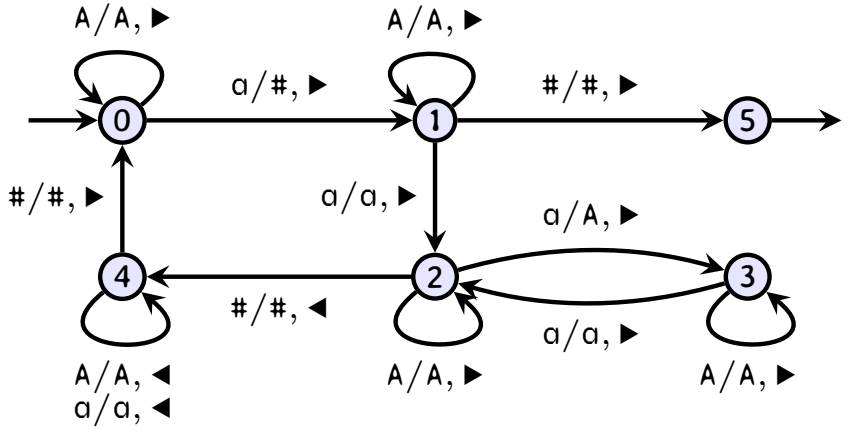
définition (Machine de Turing)

Une machine de Turing est une structure $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ où :

- Q est un ensemble fini d'états ;
- Σ est un alphabet d'entrée ;
- Γ est un alphabet de bande, tel que $\{\#\} \cup \Sigma \subseteq \Gamma$;
- $\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangleright\} \times Q$ est un ensemble de transitions ;
- $q_0 \in Q$ est un état initial, et $F \subseteq Q$ est un ensemble d'états finaux.

On utilise la notation $p \xrightarrow{a/b,d} q$ pour désigner une transition $\langle p, a, b, d, q \rangle \in \Delta$.

Machines de Turing
Configurations

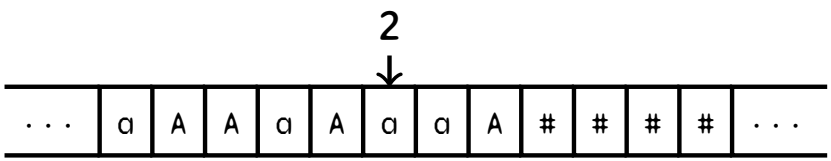


définition (Configuration)

Une configuration représente l'**état global** de la machine $\mathcal{M}$ à un instant donné. Cela comprend :

- ☞ l'état courant $q \in Q$;
- ☞ le contenu du ruban ;
- ☞ la position courante de la tête de lecture.

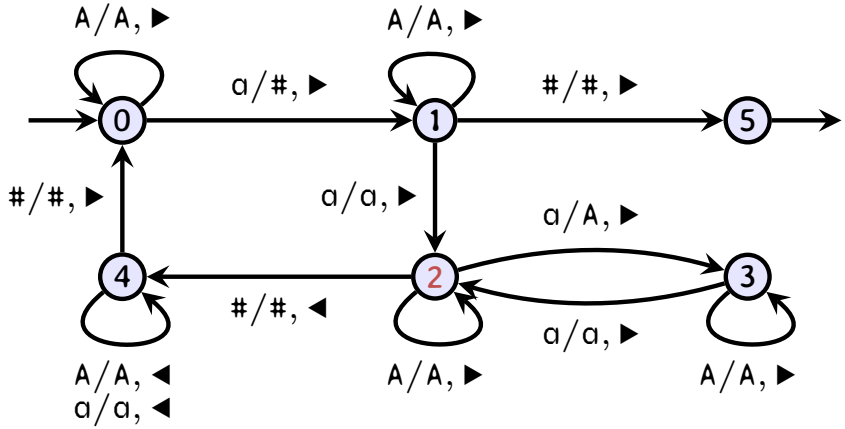
On représente une configuration par uqv, avec q l'état courant, et u et v le contenu du ruban resp. à gauche et à droite de la tête de lecture.



$\longmapsto$ aAAaA 2 aaA

Pour gérer le fait que le ruban est infini, on considère les configurations à équivalence près, en posant : $u\,q\,v \approx u\,q\,v\# \approx \#u\,q\,v$.

Machines de Turing
Configurations

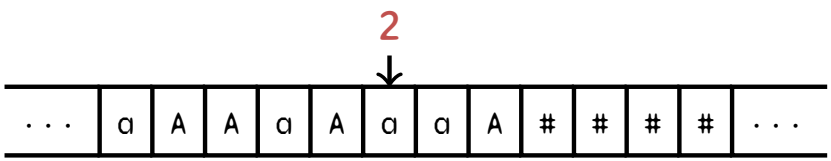


définition (Configuration)

Une configuration représente l'**état global** de la machine $\mathcal{M}$ à un instant donné. Cela comprend :

- l'état courant $q \in Q$;
- le contenu du ruban ;
- la position courante de la tête de lecture.

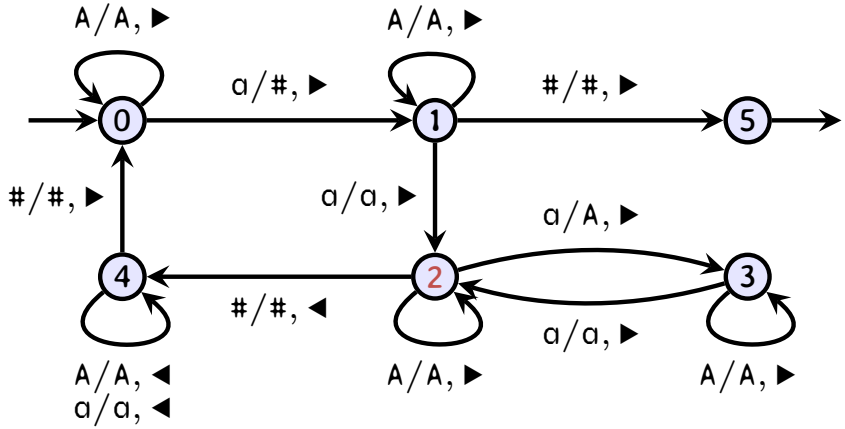
On représente une configuration par uqv, avec q l'état courant, et u et v le contenu du ruban resp. à gauche et à droite de la tête de lecture.



$\longmapsto$ aAAaA 2 aaA

Pour gérer le fait que le ruban est infini, on considère les configurations à équivalence près, en posant : $u\,q\,v \approx u\,q\,v\# \approx \#u\,q\,v$.

Machines de Turing
Configurations

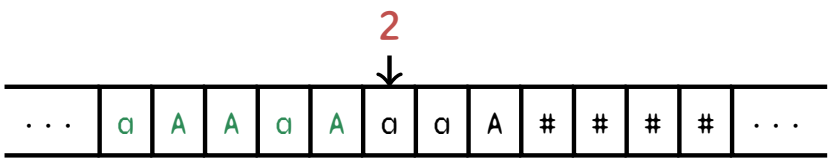


définition (Configuration)

Une configuration représente l'**état global** de la machine $\mathcal{M}$ à un instant donné. Cela comprend :

- l'état courant $q \in Q$;
- le contenu du ruban ;
- la position courante de la tête de lecture.

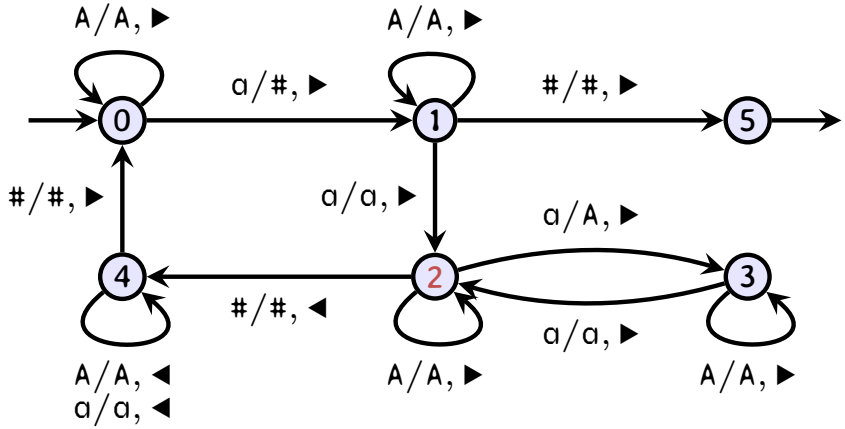
On représente une configuration par uqv, avec q l'état courant, et u et v le contenu du ruban resp. à gauche et à droite de la tête de lecture.



$\mapsto$ aAAaA 2 aaA

Pour gérer le fait que le ruban est infini, on considère les configurations à équivalence près, en posant : $u\,q\,v \approx u\,q\,v\# \approx \#u\,q\,v$.

Machines de Turing
Configurations

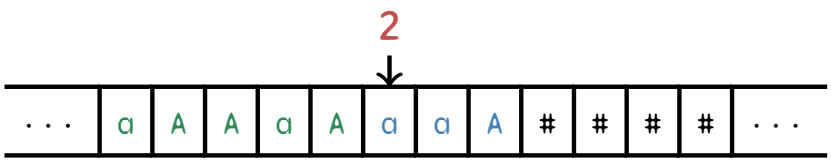


définition (Configuration)

Une configuration représente l'**état global** de la machine $\mathcal{M}$ à un instant donné. Cela comprend :

- l'état courant $q \in Q$;
- le contenu du ruban ;
- la position courante de la tête de lecture.

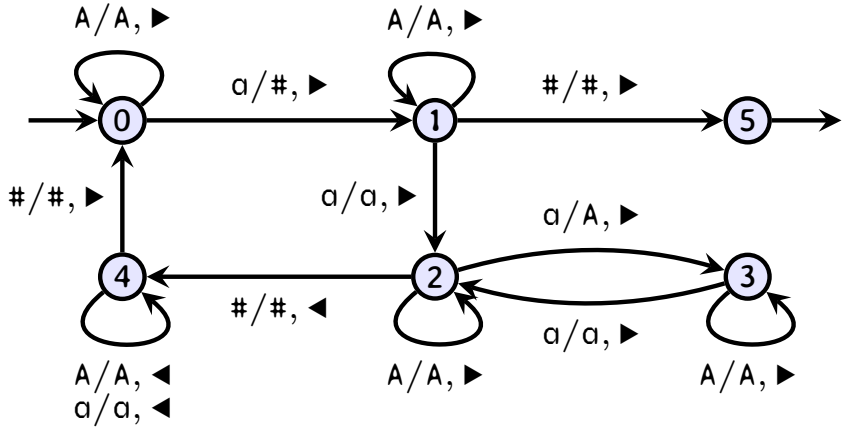
On représente une configuration par uqv, avec q l'état courant, et u et v le contenu du ruban resp. à gauche et à droite de la tête de lecture.



$\mapsto$ aAAaA 2 aaA

Pour gérer le fait que le ruban est infini, on considère les configurations à équivalence près, en posant : $u\,q\,v \approx u\,q\,v\# \approx \#u\,q\,v$.

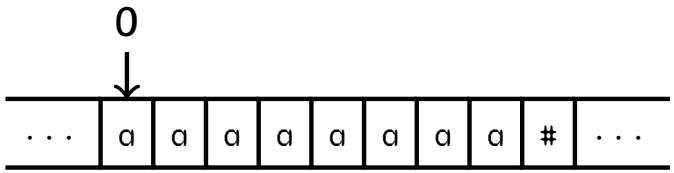
Machines de Turing
Configurations



définition (Configurations initiales/finales)

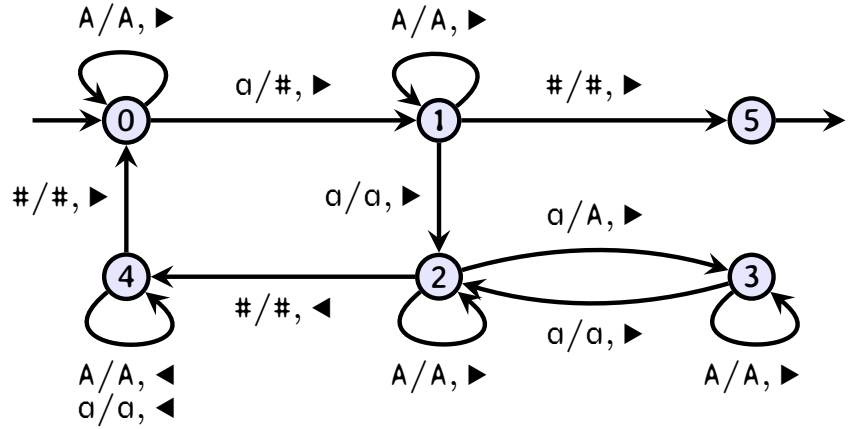
☞ Une configuration est initiale si elle est de la forme $q_0\, w$ avec $w \in \Sigma^\star$.

☞ Configuration initiale :



0 aaaaaaaa

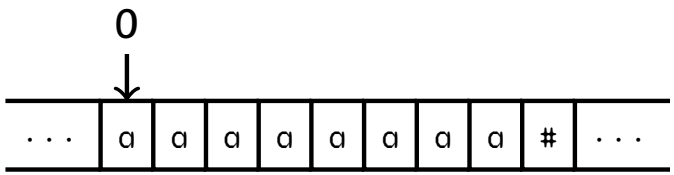
Machines de Turing
Configurations



définition (Configurations initiales/finales)

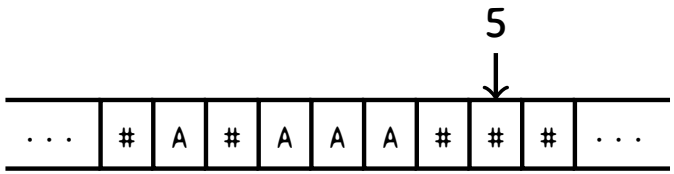
- Une configuration est initiale si elle est de la forme $q_0\, w$ avec $w \in \Sigma^\star$.
- Une configuration est finale si l'état courant est un état final. Autrement dit les configurations initiales s'écrivent $u\, q\, v$ avec $q \in F$ et $u, v \in \Gamma^\star$.

- Configuration initiale :



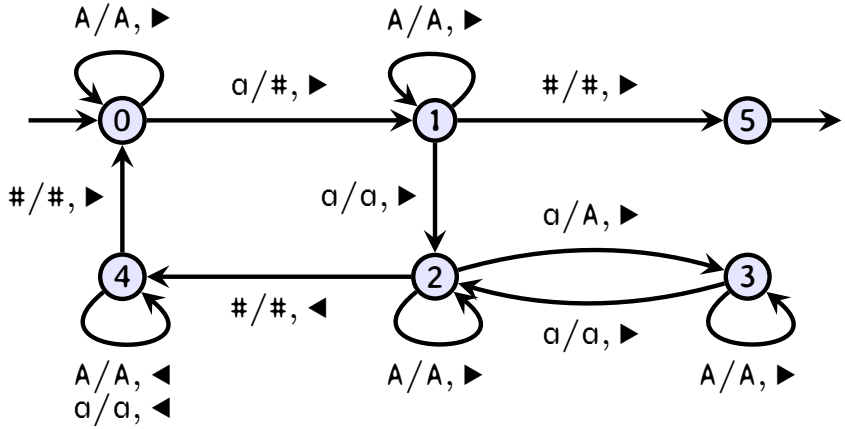
0 aaaaaaaa

- Configuration finale :



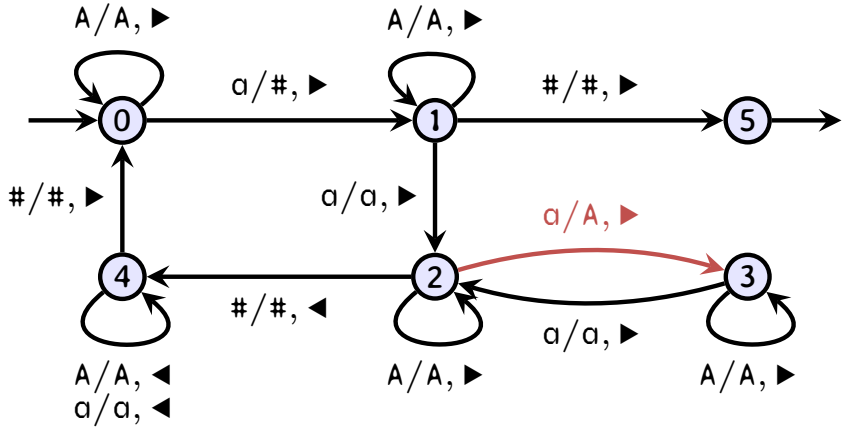
#A#AAA# 5 #

Machine de Turing
Étape de calcul



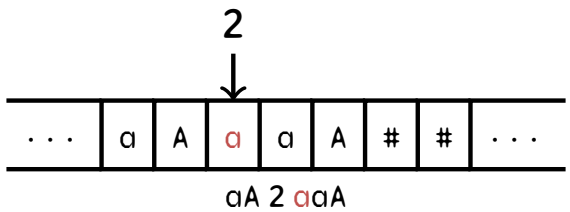
Une étape de calcul dans une machine de Turing correspond à l'exécution d'une transition depuis une configuration donnée, et mène à une nouvelle configuration.

Machine de Turing
Étape de calcul

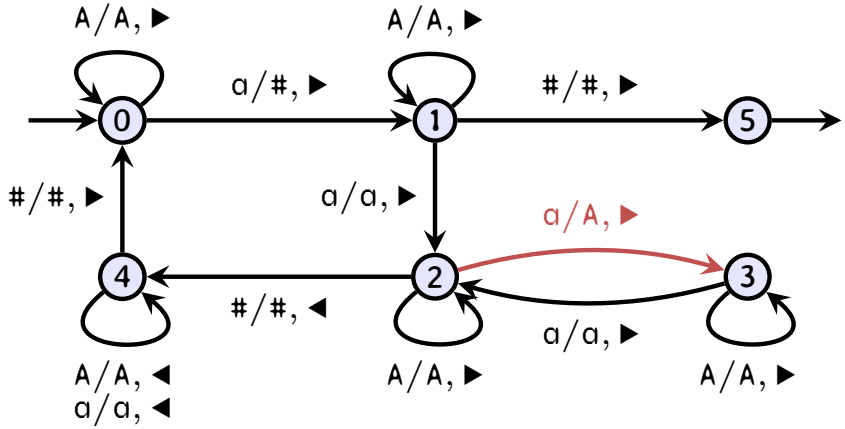


Une étape de calcul dans une machine de Turing correspond à l'exécution d'une transition depuis une configuration donnée, et mène à une nouvelle configuration.

☞ Si on a une transition $p \xrightarrow{a/b,\blacktriangleright} q \in \Delta$,

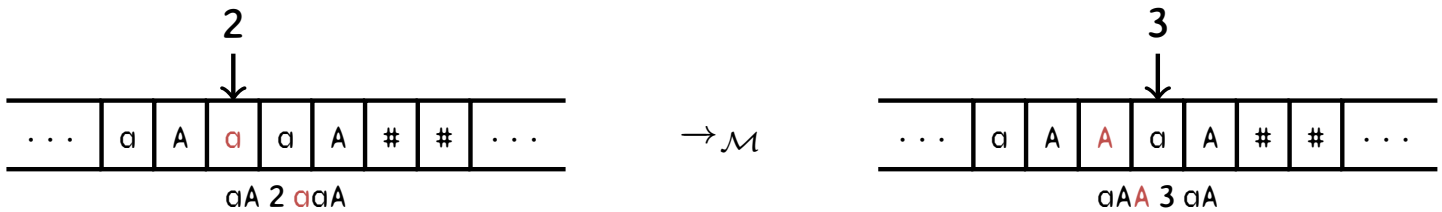


Machine de Turing
Étape de calcul

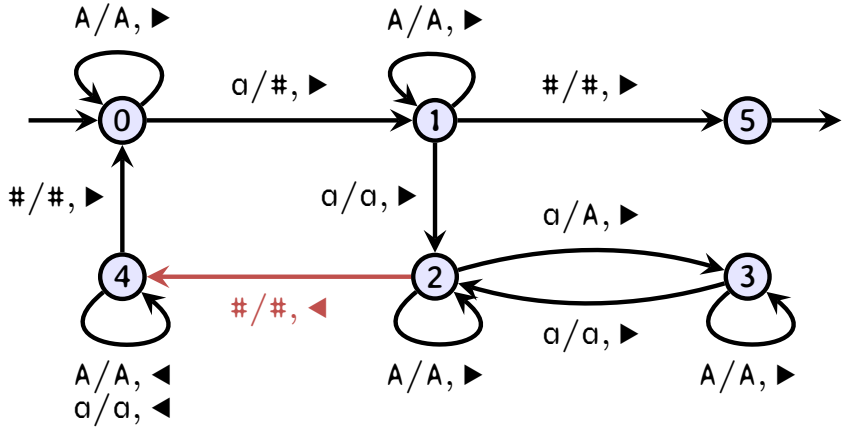


Une étape de calcul dans une machine de Turing correspond à l'exécution d'une transition depuis une configuration donnée, et mène à une nouvelle configuration.

☞ Si on a une transition $p \xrightarrow{a/b,\blacktriangleright} q \in \Delta$, alors $u\,p\,av \to_{\mathcal{M}} ub\,q\,v$.

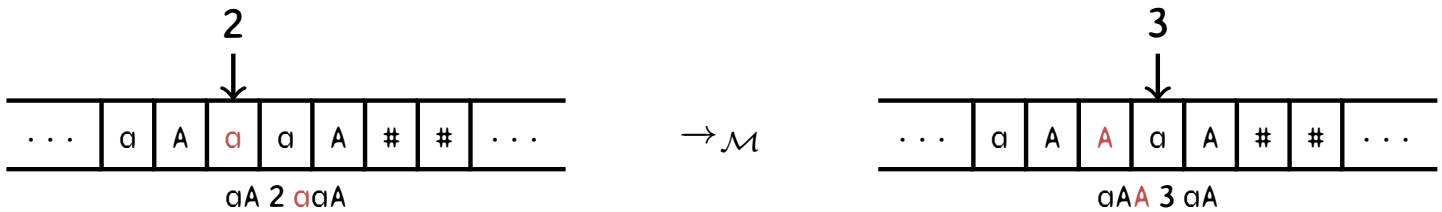


Machine de Turing
Étape de calcul

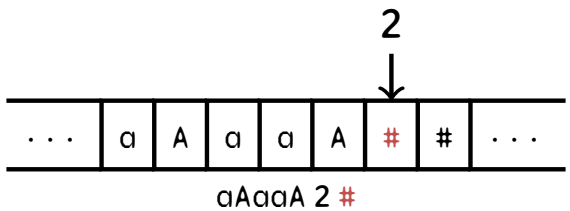


Une étape de calcul dans une machine de Turing correspond à l'exécution d'une transition depuis une configuration donnée, et mène à une nouvelle configuration.

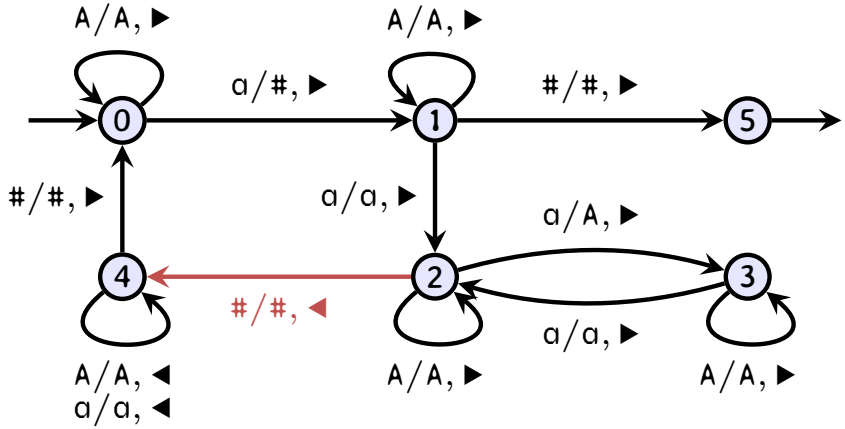
☞ Si on a une transition $p \xrightarrow{a/b,\blacktriangleright} q \in \Delta$, alors $u\,p\,av \to_{\mathcal{M}} ub\,q\,v$.



☞ Si on a une transition $p \xrightarrow{a/b,\blacktriangleleft} q \in \Delta$,

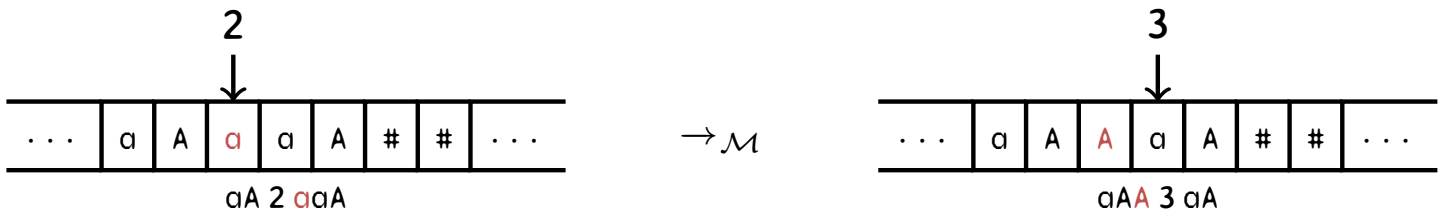


Machine de Turing
Étape de calcul

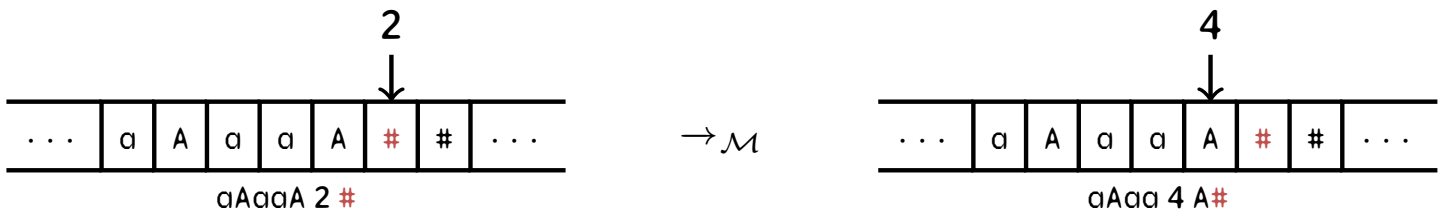


Une étape de calcul dans une machine de Turing correspond à l'exécution d'une transition depuis une configuration donnée, et mène à une nouvelle configuration.

☞ Si on a une transition $p \xrightarrow{a/b,\blacktriangleright} q \in \Delta$, alors $u\,p\,av \to_{\mathcal{M}} ub\,q\,v$.



☞ Si on a une transition $p \xrightarrow{a/b,\blacktriangleleft} q \in \Delta$, alors $uc\,p\,av \to_{\mathcal{M}} u\,q\,cbv$.



définition (Exécution)

- Une exécution d'une machine de Turing $\mathcal{M}$ est une séquence de configurations

$$C_0 \to_{\mathcal{M}} C_1 \to_{\mathcal{M}} \cdots \to_{\mathcal{M}} C_n.$$

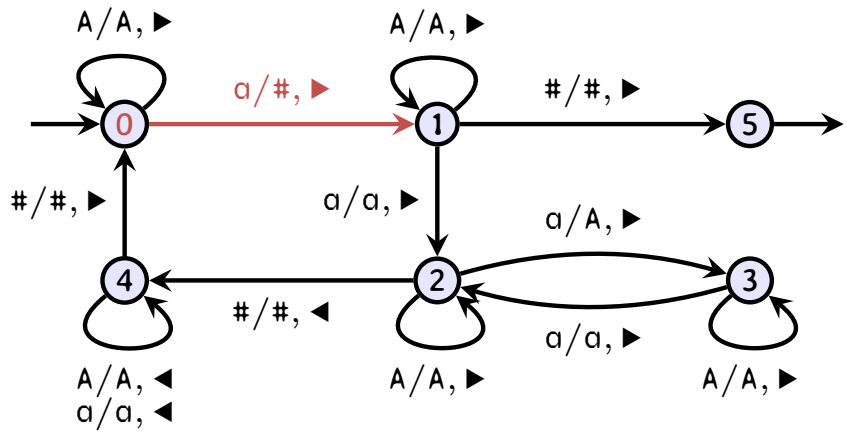
- On note $C_0 \to^*_{\mathcal{M}} C_n$ lorsequ'il existe une exécution de $\mathcal{M}$ allant de C_0 à C_n.
- Une exécution est acceptante si C_0 est une configuration initiale, et C_n est une configuration finale.

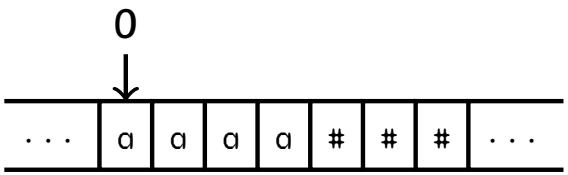
définition (Langage reconnu par $\mathcal{M}$)

Le langage reconnu par une machine de Turing est l'ensemble des mots $w \in \Sigma^\star$ tels qu'il existe une exécution $q_0\, w \to^*_{\mathcal{M}} C$ où C est finale.

$$\mathcal{L}(\mathcal{M}) := \{w \in \Sigma^\star \mid q_0\, w \to^*_{\mathcal{M}} u\, q\, v \text{ et } q \in F\}\,.$$

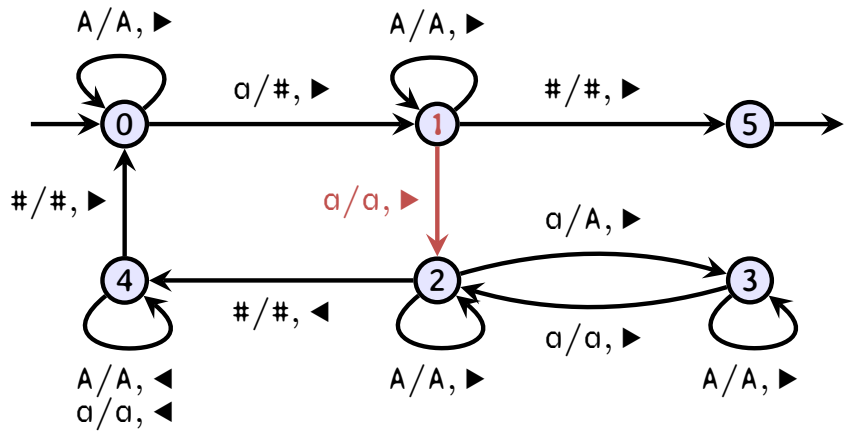
Calcul dans une machine de Turing
Exemple

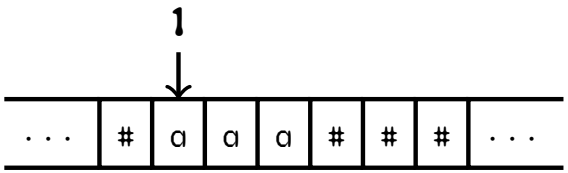




1) 0 aaaa

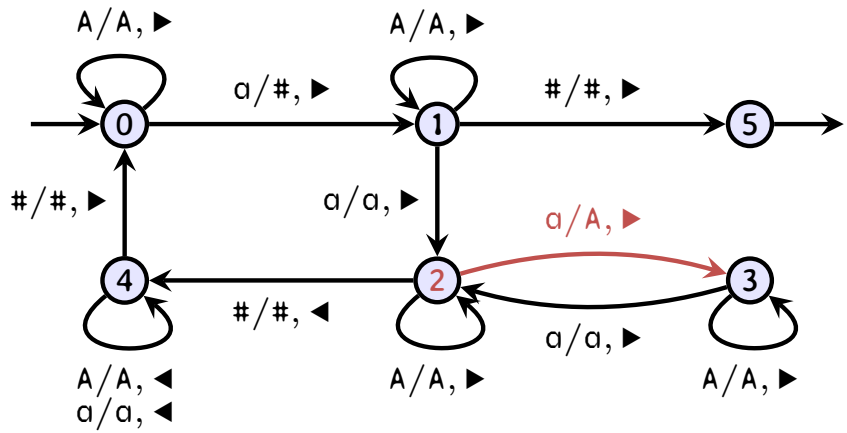
Calcul dans une machine de Turing
Exemple

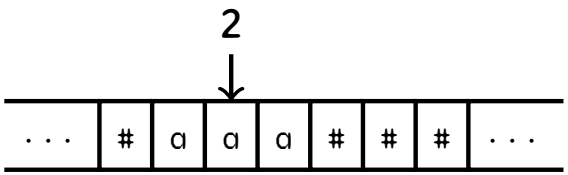




1) 0 aaaa
2) # 1 aaa

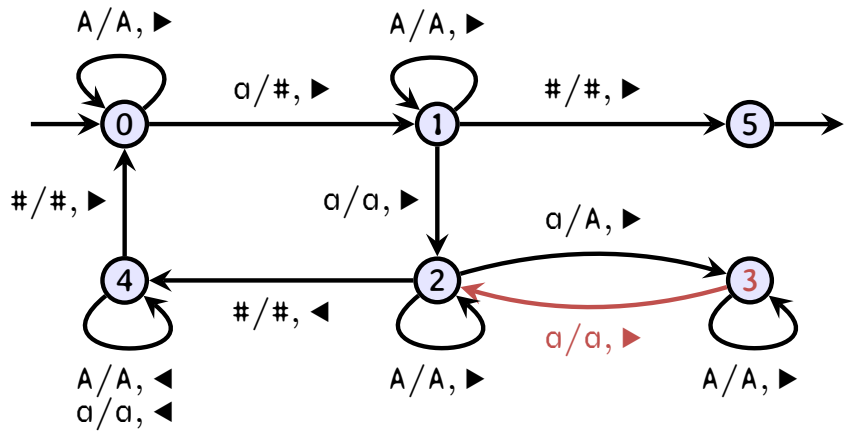
Calcul dans une machine de Turing
Exemple

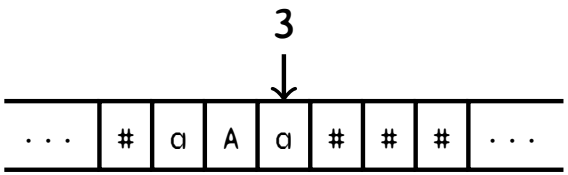




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa

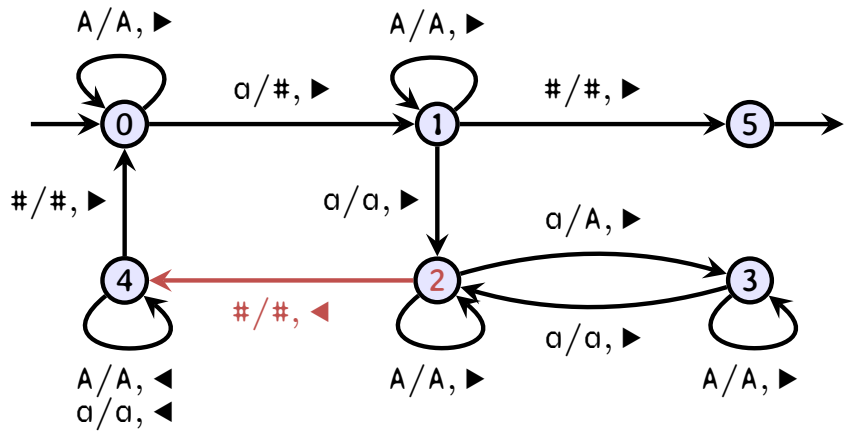
Calcul dans une machine de Turing
Exemple

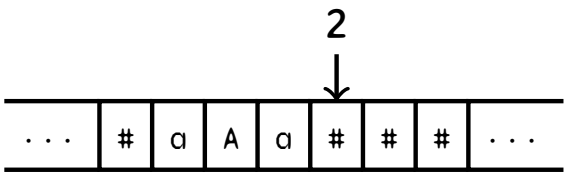




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a

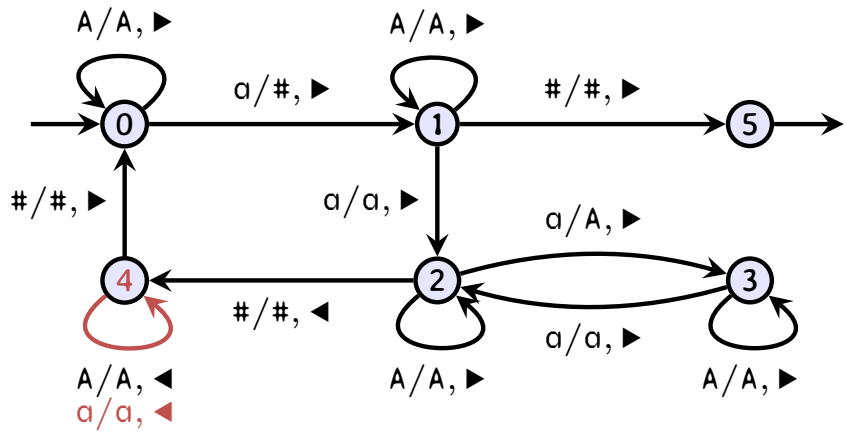
Calcul dans une machine de Turing
Exemple

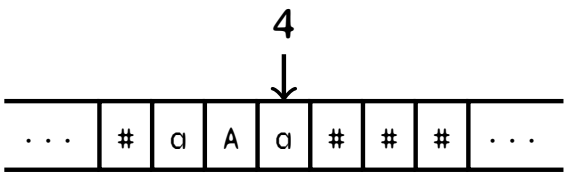




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #

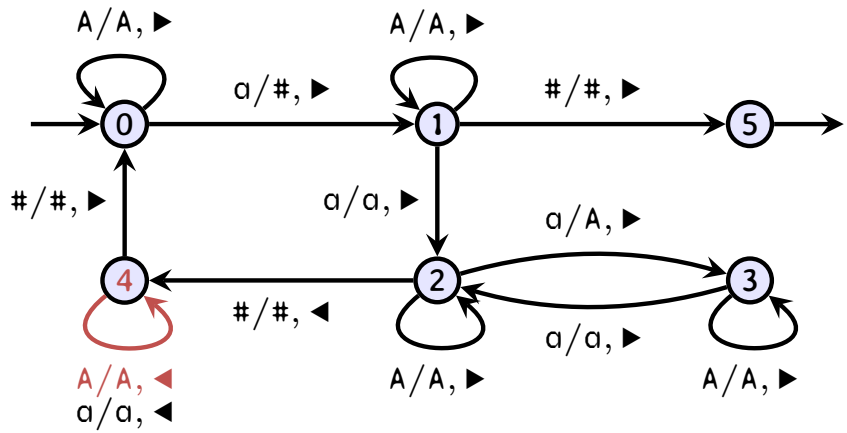
Calcul dans une machine de Turing
Exemple

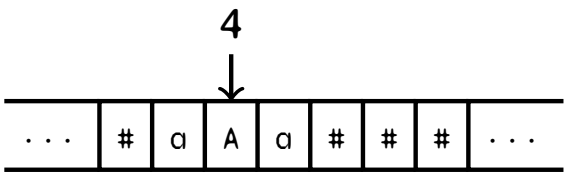




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#

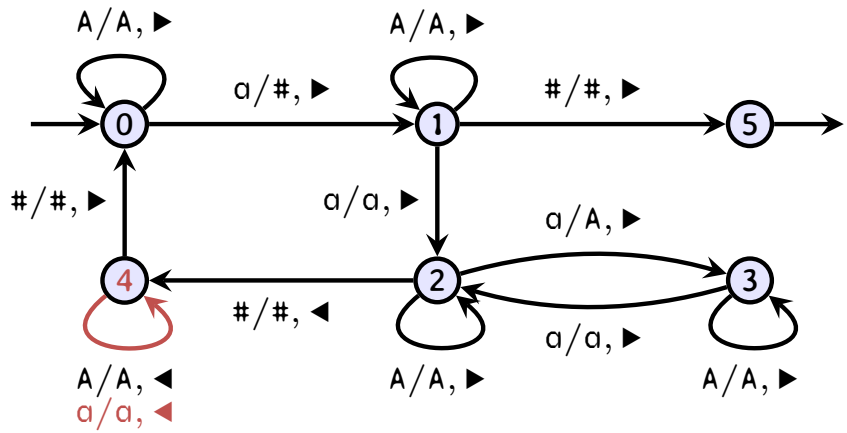
Calcul dans une machine de Turing
Exemple

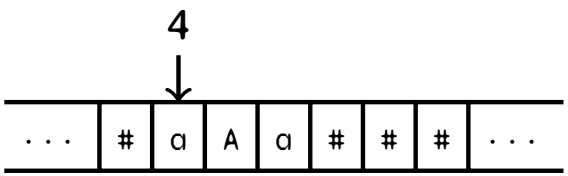




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#

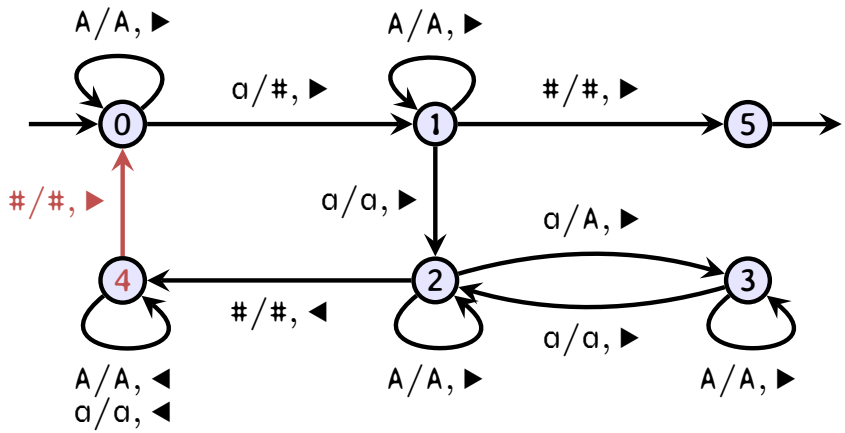
Calcul dans une machine de Turing
Exemple

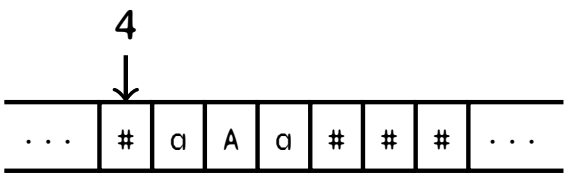




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#

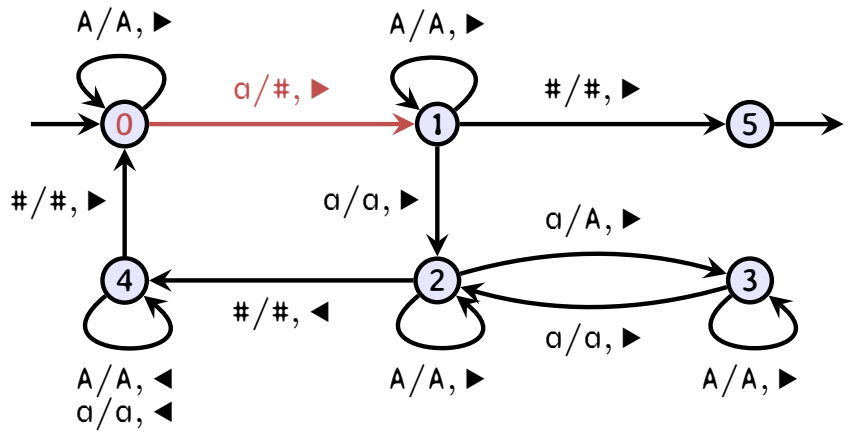
Calcul dans une machine de Turing
Exemple

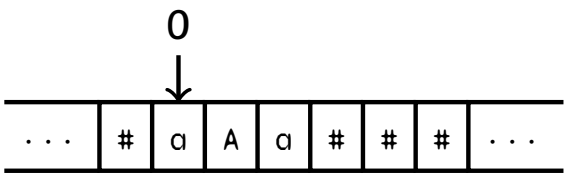




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#

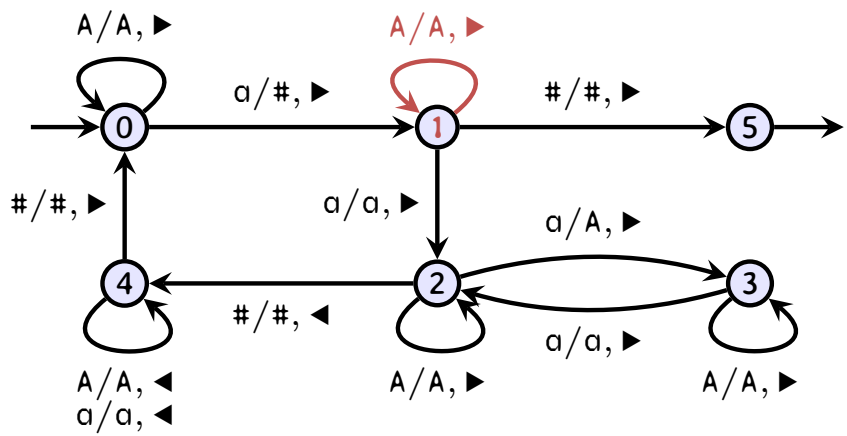
Calcul dans une machine de Turing
Exemple

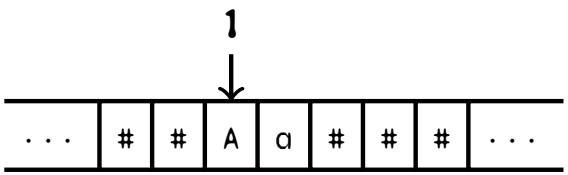




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#

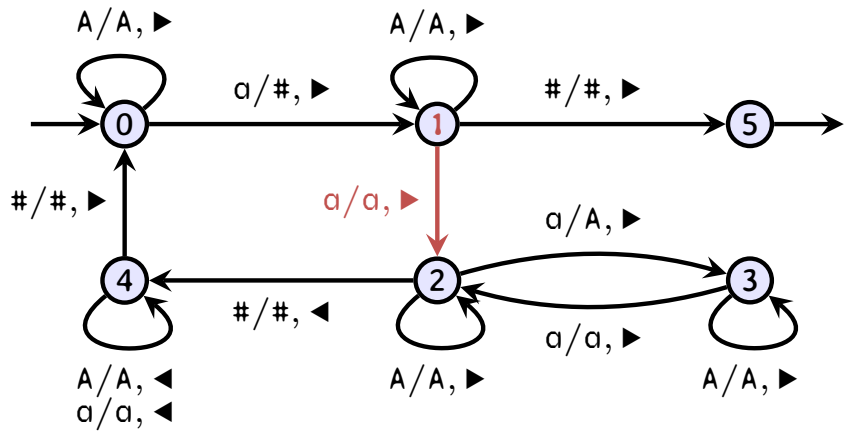
Calcul dans une machine de Turing
Exemple

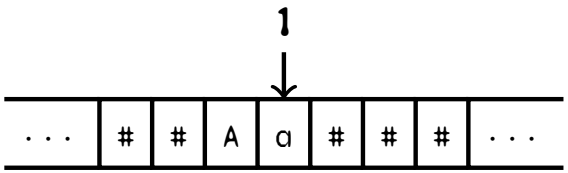




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#

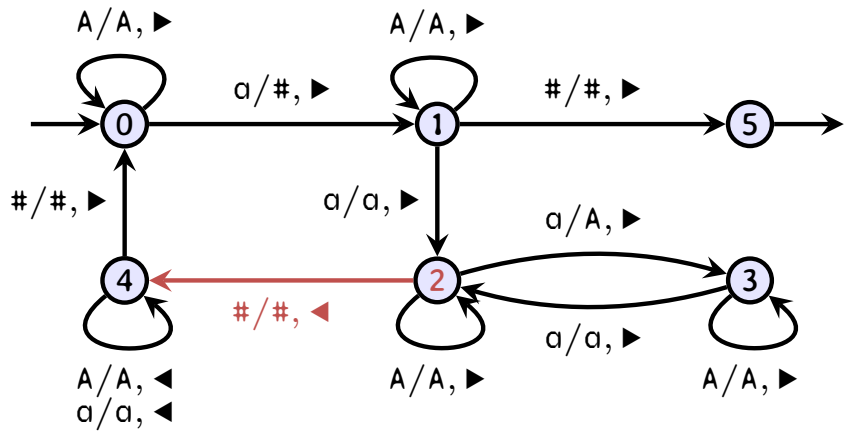
Calcul dans une machine de Turing
Exemple

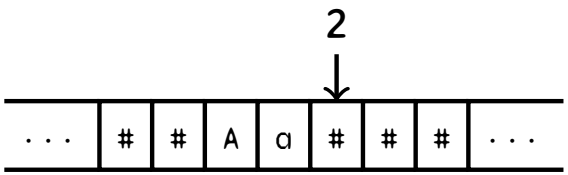




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#

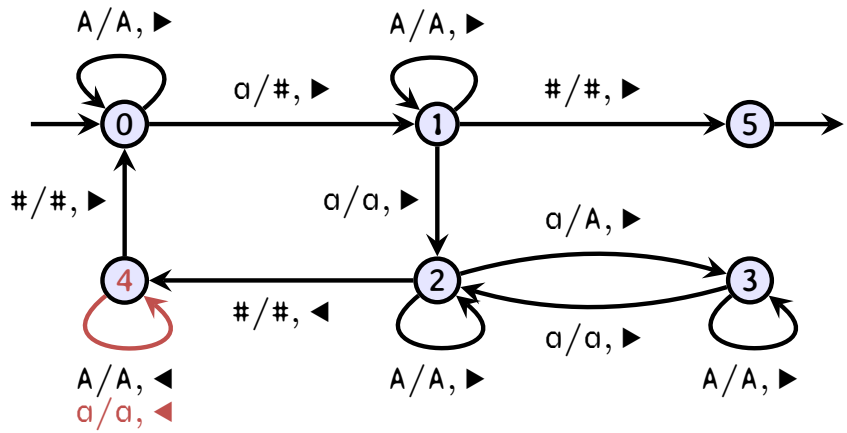
Calcul dans une machine de Turing
Exemple

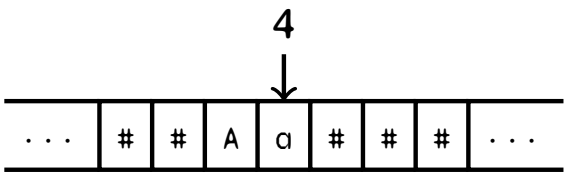




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #

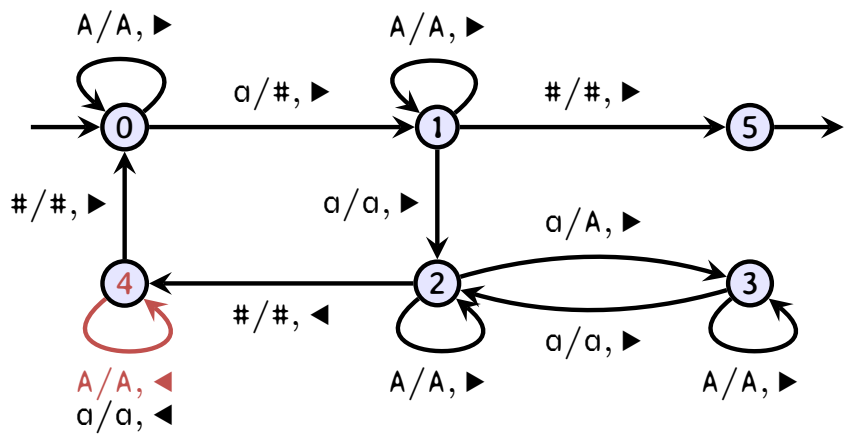
Calcul dans une machine de Turing
Exemple

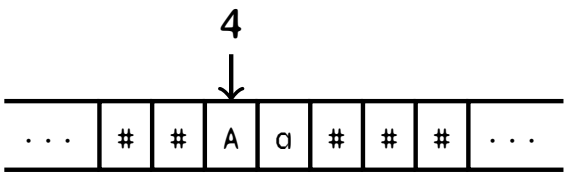




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#

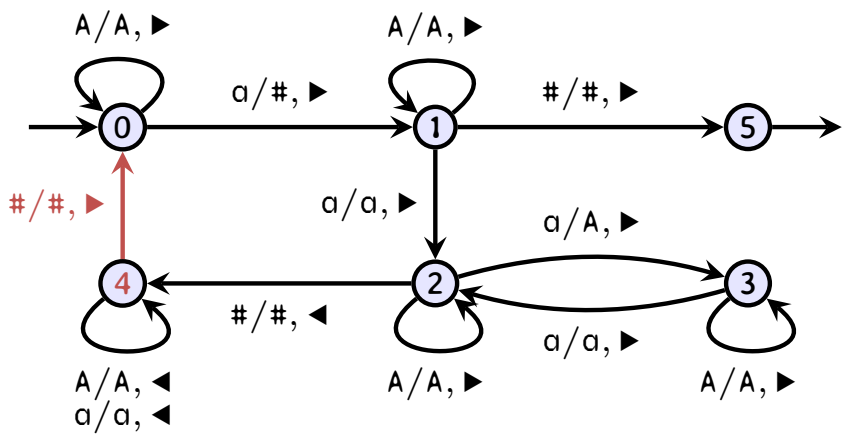
Calcul dans une machine de Turing
Exemple

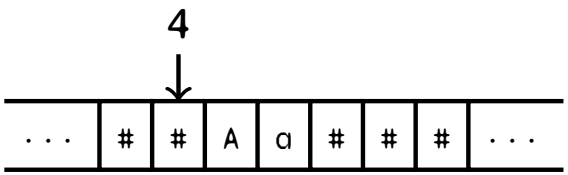




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#

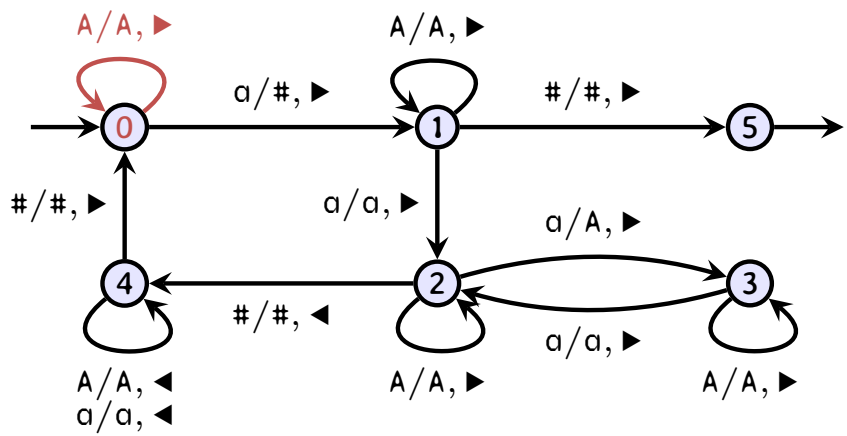
Calcul dans une machine de Turing
Exemple

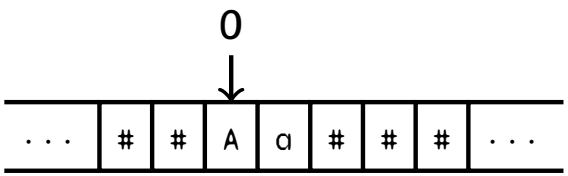




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#
16) # 4 #Aa#

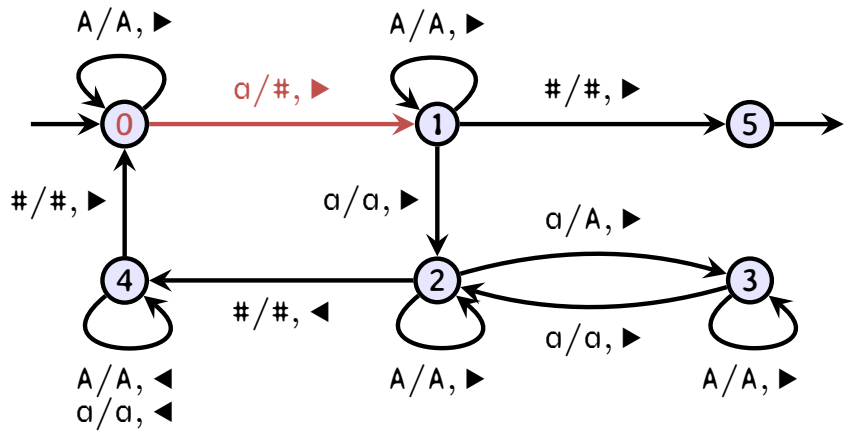
Calcul dans une machine de Turing
Exemple

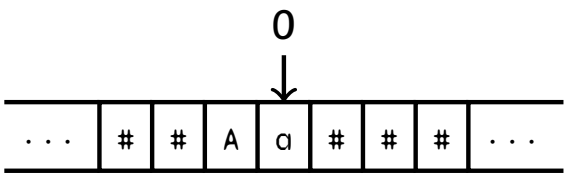




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#
16) # 4 #Aa#
17) ## 0 Aa#

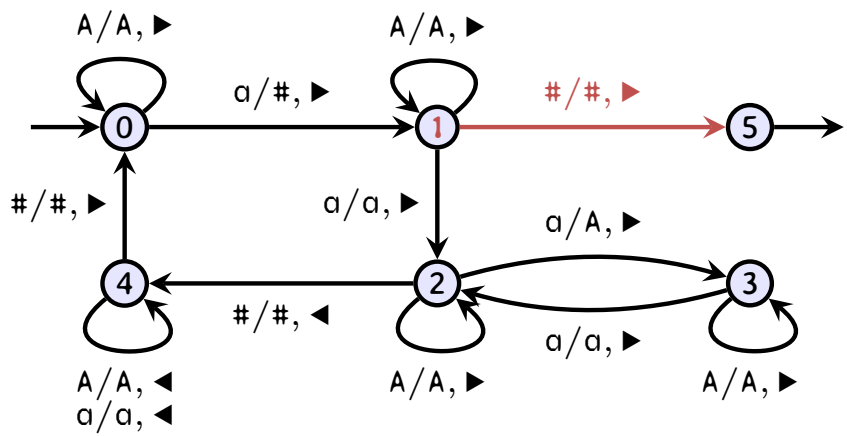
Calcul dans une machine de Turing
Exemple

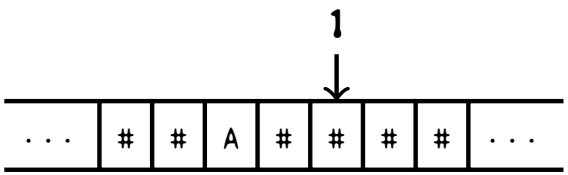




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#
16) # 4 #Aa#
17) ## 0 Aa#
18) ##A 0 a#

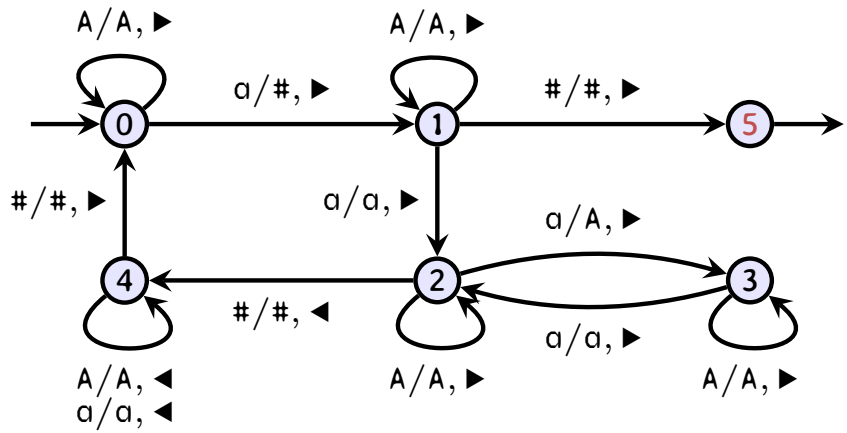
Calcul dans une machine de Turing
Exemple

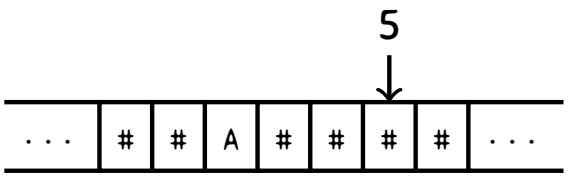




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#
16) # 4 #Aa#
17) ## 0 Aa#
18) ##A 0 a#
19) ##A# 1 #

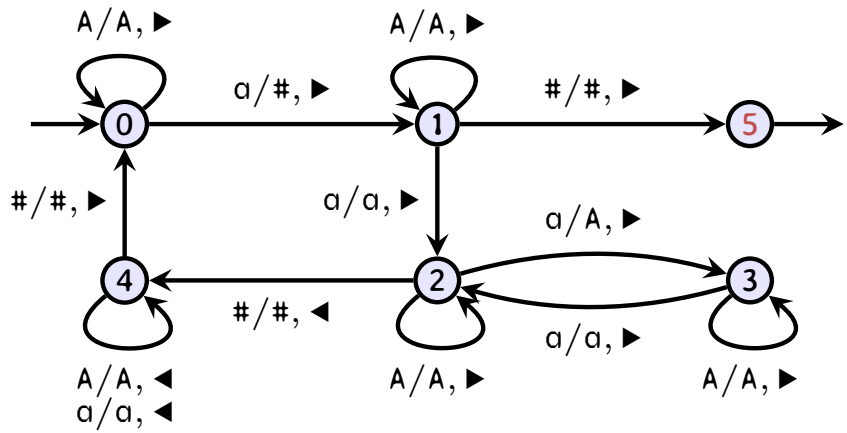
Calcul dans une machine de Turing
Exemple

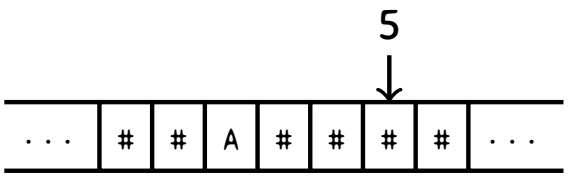




1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#
16) # 4 #Aa#
17) ## 0 Aa#
18) ##A 0 a#
19) ##A# 1 #
20) ##A## 5 #

Calcul dans une machine de Turing
Exemple

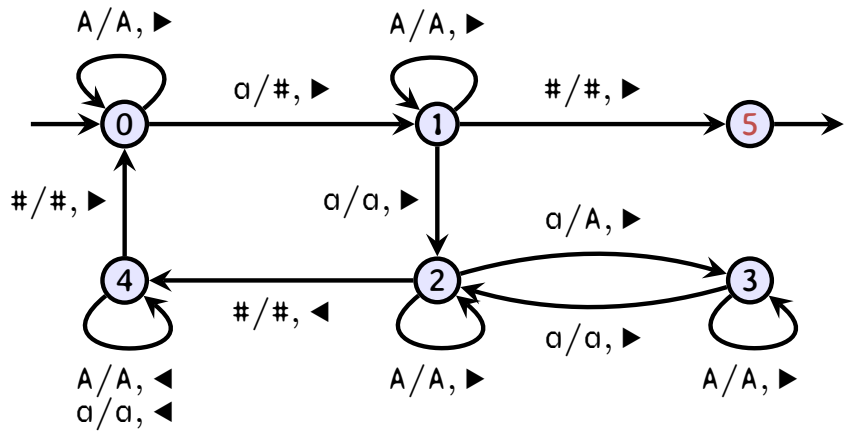


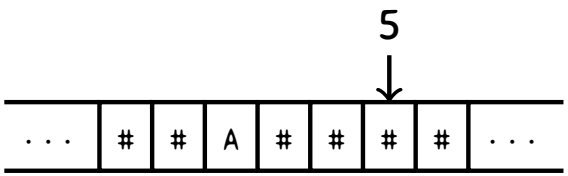


1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#
16) # 4 #Aa#
17) ## 0 Aa#
18) ##A 0 a#
19) ##A# 1 #
20) ##A## 5 #

Le mot aaaa est accepté.

Calcul dans une machine de Turing
Exemple





1) 0 aaaa
2) # 1 aaa
3) #a 2 aa
4) #aA 3 a
5) #aAa 2 #
6) #aA 4 a#
7) #a 4 Aa#
8) # 4 aAa#
9) 4 #aAa#
10) # 0 aAa#
11) ## 1 Aa#
12) ##A 1 a#
13) ##Aa 2 #
14) ##A 4 a#
15) ## 4 Aa#
16) # 4 #Aa#
17) ## 0 Aa#
18) ##A 0 a#
19) ##A# 1 #
20) ##A## 5 #

Le mot aaaa est accepté.

Exercice

Quel est le langage reconnu par cette machine ?

Langages reconnaissables

définition

Un langage $L \subseteq \Sigma^\star$ est dit **reconnaissable** si il existe une machine $\mathcal{M}$ telle que $L = \mathcal{L}(\mathcal{M})$.

Les langages reconnaissables constituent la classe des langages expressibles par le modèle de calcul $\langle \mathbb{MT}, \models \rangle$, où $\mathbb{MT}$ est l'ensemble des machines de Turing avec Σ comme alphabet d'entrée, et $w \models \mathcal{M}$ si et seulement si $w \in \mathcal{L}(\mathcal{M})$.

Machines de Turing
Plan

Équivalence de modèles

☞ On va maintenant considérer des variations de la définition de machine de Turing.

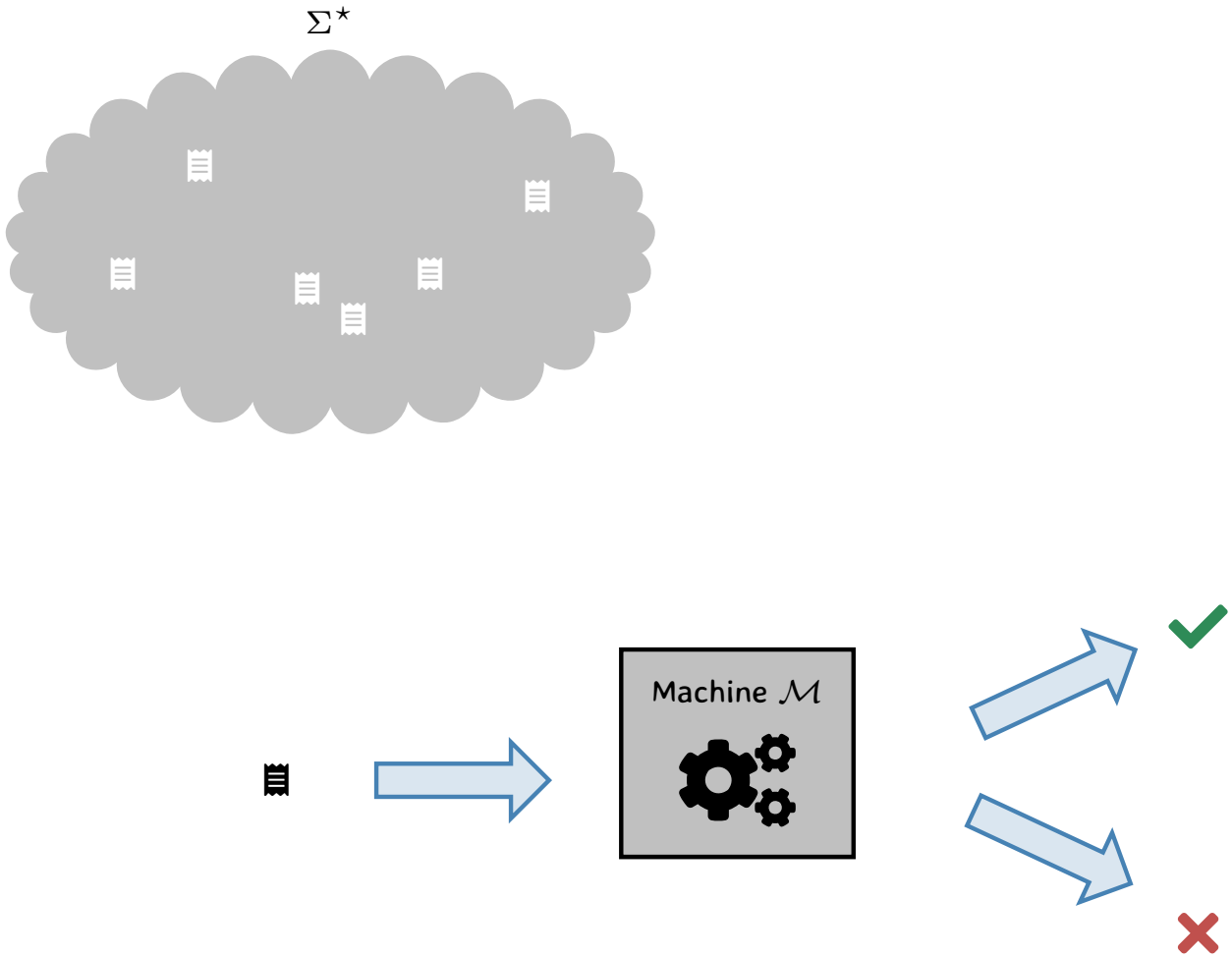
Équivalence de modèles

- ☞ On va maintenant considérer des variations de la définition de machine de Turing.
- ☞ On voudra vérifier que toutes ces variations sont **équivalentes**.

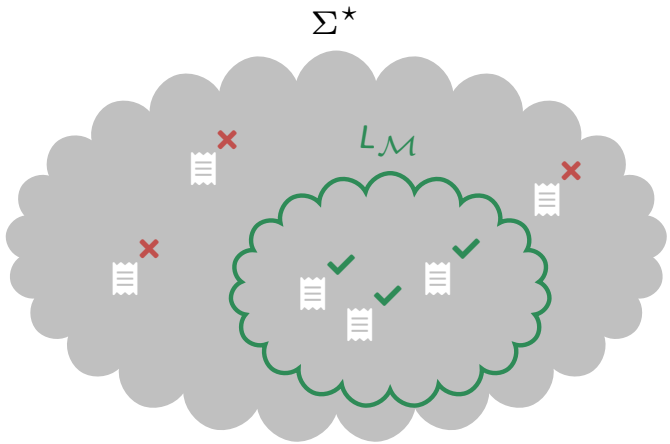
Équivalence de modèles

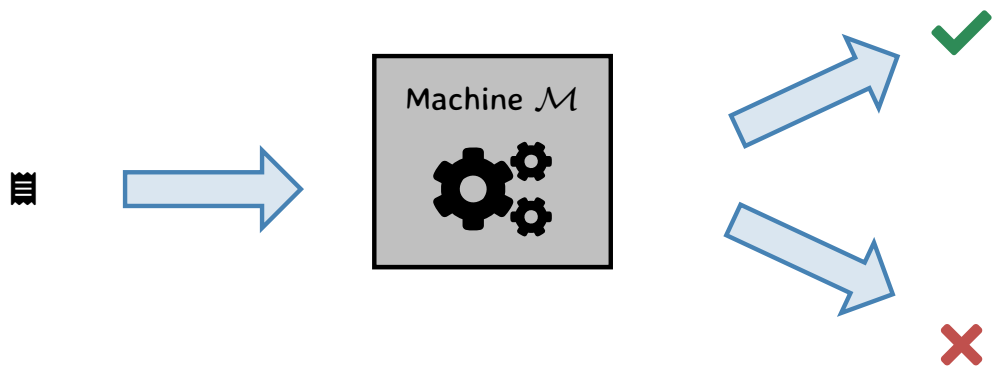
- On va maintenant considérer des variations de la définition de machine de Turing.
- On voudra vérifier que toutes ces variations sont **équivalentes**.
- Pour cela, on rappelle la définition d'un **modèle de calcul**, puis celle d'**équivalence de modèles**.

Modèle de calcul



Modèle de calcul





Expressivité d'un modèle de calcul

définition (Modèle de calcul)

Un **modèle de calcul** sur l'alphabet Σ est une paire $\mathscr{M} = \langle \mathbb{M}, \models \rangle$ où $\mathbb{M}$ est un ensemble de **machines**, et $\models \subseteq \Sigma^\star \times \mathbb{M}$ est une relation binaire d'**acceptation**.

Expressivité d'un modèle de calcul

définition (Modèle de calcul)

Un **modèle de calcul** sur l'alphabet Σ est une paire $\mathscr{M} = \langle \mathbb{M}, \models \rangle$ où $\mathbb{M}$ est un ensemble de **machines**, et $\models \subseteq \Sigma^\star \times \mathbb{M}$ est une relation binaire d'**acceptation**.

Une **machine** définit un **langage** :

$$L_{\mathcal{M}} := \{w \in \Sigma^\star \mid w \models \mathcal{M}\}\,.$$

Expressivité d'un modèle de calcul

définition (Modèle de calcul)

Un **modèle de calcul** sur l'alphabet Σ est une paire $\mathscr{M} = \langle \mathbb{M}, \models \rangle$ où $\mathbb{M}$ est un ensemble de **machines**, et $\models \subseteq \Sigma^\star \times \mathbb{M}$ est une relation binaire d'**acceptation**.

Une **machine** définit un **langage** :

$$L_{\mathcal{M}} := \{w \in \Sigma^\star \mid w \models \mathcal{M}\}\,.$$

Un **modèle** définit une **classe de langages** :

$$\mathsf{Lang}\,(\mathscr{M}) := \{L_{\mathcal{M}} \subseteq \Sigma^\star \mid \mathcal{M} \in \mathbb{M}\}\,.$$

Comparaison de modèles

définition (Comparaison de modèles)

Soient $\mathscr{M}_1$ et $\mathscr{M}_2$ deux modèles de calcul.

☞ $\mathscr{M}_1$ se réduit à $\mathscr{M}_2$, noté $\mathscr{M}_1 \preceq \mathscr{M}_2$, si pour toute machine $\mathcal{M} \in \mathbb{M}_1$ il existe une machine $\mathcal{M}' \in \mathbb{M}_2$ acceptant le même langage, autrement dit :

$$\forall \mathcal{M} \in \mathbb{M}_1,\ \exists \mathcal{M}' \in \mathbb{M}_2 :\ \forall w \in \Sigma^\star,\ w \models_1 \mathcal{M} \Leftrightarrow w \models_2 \mathcal{M}'.$$

☞ $\mathscr{M}_1$ est équivalent à $\mathscr{M}_2$, noté $\mathscr{M}_1 \approx \mathscr{M}_2$, si $\mathscr{M}_1 \preceq \mathscr{M}_2$ et $\mathscr{M}_2 \preceq \mathscr{M}_1$.

Comparaison de modèles

définition (Comparaison de modèles)

Soient $\mathscr{M}_1$ et $\mathscr{M}_2$ deux modèles de calcul.

☞ $\mathscr{M}_1$ se réduit à $\mathscr{M}_2$, noté $\mathscr{M}_1 \preceq \mathscr{M}_2$, si pour toute machine $\mathcal{M} \in \mathbb{M}_1$ il existe une machine $\mathcal{M}' \in \mathbb{M}_2$ acceptant le même langage, autrement dit :

$$\forall \mathcal{M} \in \mathbb{M}_1,\ \exists \mathcal{M}' \in \mathbb{M}_2 :\ \forall w \in \Sigma^\star,\ w \models_1 \mathcal{M} \Leftrightarrow w \models_2 \mathcal{M}'.$$

☞ $\mathscr{M}_1$ est équivalent à $\mathscr{M}_2$, noté $\mathscr{M}_1 \approx \mathscr{M}_2$, si $\mathscr{M}_1 \preceq \mathscr{M}_2$ et $\mathscr{M}_2 \preceq \mathscr{M}_1$.

remarque

Pour toute paire de modèles $\mathscr{M}_1$, $\mathscr{M}_2$ on a :

$$\mathscr{M}_1 \preceq \mathscr{M}_2 \Leftrightarrow \text{Lang}\,(\mathscr{M}_1) \subseteq \text{Lang}\,(\mathscr{M}_2)$$

Preuve de réduction $\approx$ compilateur

$\mathscr{M}_1$ se réduit à $\mathscr{M}_2$, noté $\mathscr{M}_1 \preceq \mathscr{M}_2$, si pour toute machine $\mathcal{M} \in \mathbb{M}_1$ il existe une machine $\mathcal{M}' \in \mathbb{M}_2$ acceptant le même langage, autrement dit :

$$\forall \mathcal{M} \in \mathbb{M}_1,\ \exists \mathcal{M}' \in \mathbb{M}_2 :\ \forall w \in \Sigma^\star,\ w \models_1 \mathcal{M} \Leftrightarrow w \models_2 \mathcal{M}'.$$

- Prouver que $\mathscr{M}_1 \preceq \mathscr{M}_2$ revient à trouver une fonction $\varphi : \mathbb{M}_1 \to \mathbb{M}_2$ telle que pour toute machine $\mathcal{M} \in \mathbb{M}_1$, on aie $\mathcal{L}_1(\mathcal{M}) = \mathcal{L}_2(\varphi(\mathcal{M}))$.
- Généralement, un modèle de calcul $\mathscr{M}$ est associé à une syntaxe, permettant de spécifier une machine par un texte structuré sur un alphabet $\Sigma_{\mathscr{M}}$.

$$\forall \mathcal{M} \in \mathbb{M},\ \exists \texttt{prog} \in (\Sigma_{\mathscr{M}})^\star :\ \mathcal{M} = \texttt{parse}_{\mathscr{M}}(\texttt{prog}).$$

définition

- Un parseur pour $\mathscr{M}$ est une fonction surjective $\texttt{parse}_{\mathscr{M}} : (\Sigma_{\mathscr{M}})^\star \to \mathbb{M}$.
- Un compilateur de $\mathscr{M}_1$ à $\mathscr{M}_2$ est une fonction $\texttt{compile} : (\Sigma_{\mathscr{M}_1})^\star \to (\Sigma_{\mathscr{M}_2})^\star$ telle que :

$$\mathcal{L}_1(\texttt{parse}_{\mathscr{M}_1}(\texttt{prog})) = \mathcal{L}_2(\texttt{parse}_{\mathscr{M}_2}(\texttt{compile}(\texttt{prog})))$$

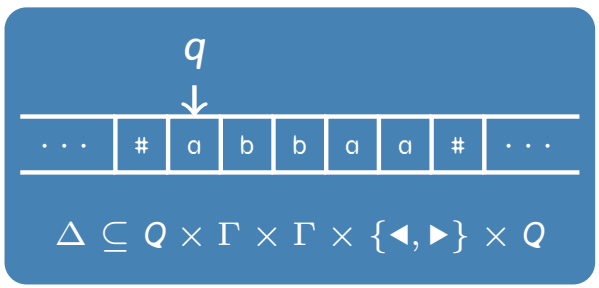
Machines de Turing
Plan

1. Premières définitions

2. Équivalence de modèles

3. Modèles équivalents
   - 3.1. Machine paresseuse
   - 3.2. Position initiale de la tête de lecture
   - 3.3. Ruban semi-infini
   - 3.4. Rubans multiples

Modèles équivalents



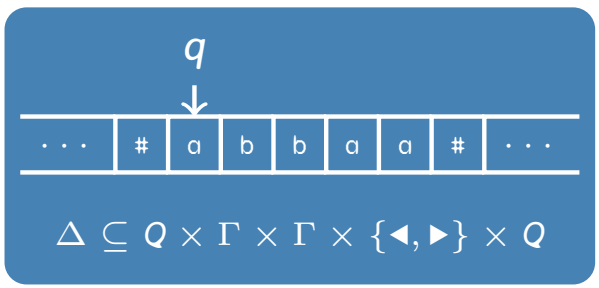
Machines de Turing

Modèles équivalents

$\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\} \times Q$

Jeux d'instructions différents

$\Delta \subseteq Q \times \Gamma \times (\Gamma + \{\blacktriangleleft, \blacktriangleright\}) \times Q$



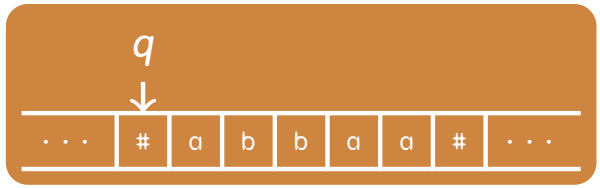
Machines de Turing

Modèles équivalents

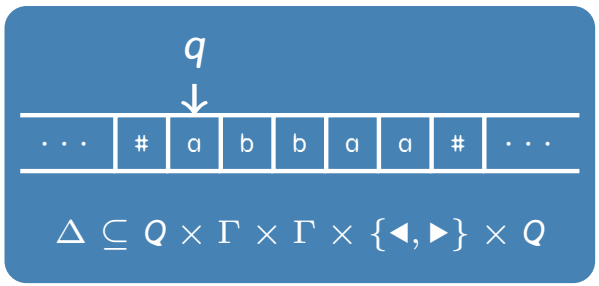
$\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\} \times Q$

Jeux d'instructions différents

$\Delta \subseteq Q \times \Gamma \times (\Gamma + \{\blacktriangleleft, \blacktriangleright\}) \times Q$



Initialisation différente



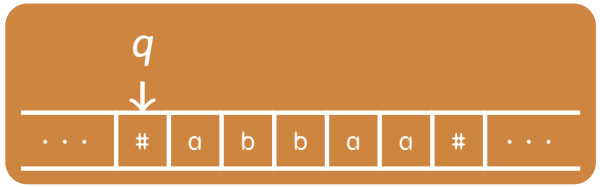
Machines de Turing

Modèles équivalents

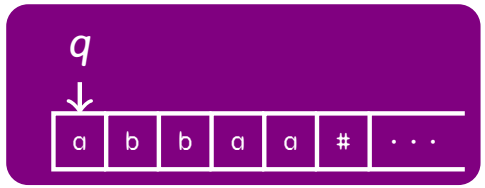
$\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\} \times Q$

Jeux d'instructions différents

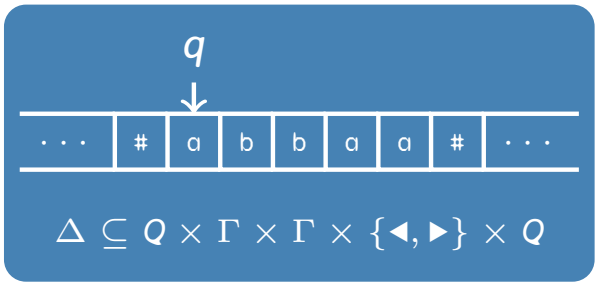
$\Delta \subseteq Q \times \Gamma \times (\Gamma + \{\blacktriangleleft, \blacktriangleright\}) \times Q$



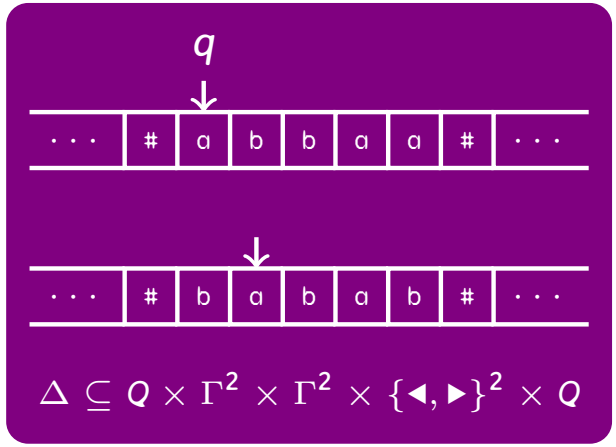
Initialisation différente



Rubans différents



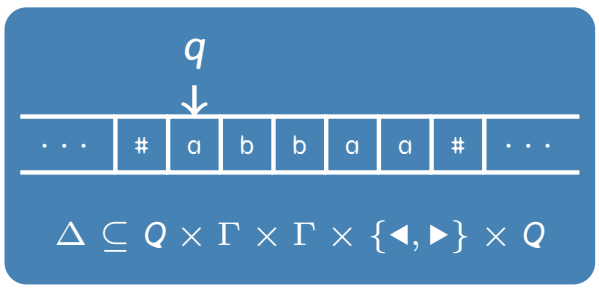
Machines de Turing



Modèles équivalents

$\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\} \times Q$

Jeux d'instructions différents



Machines de Turing

Machine paresseuse
Jeux d'instructions différents

définition (Machine paresseuse)

Une machine de Turing paresseuse est une structure $\langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$, où Q est un ensemble fini d'états, Σ et Γ sont deux alphabet (respectivement d'entrée et de bande), $q_0 \in Q$ est l'état initial, $F \subseteq Q$ est l'ensemble des états acceptants, et :

$$\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangleright, \blacktriangledown\} \times Q.$$

Les configurations et transitions d'une telle machine sont similaires à celles d'une machine de Turing classique, avec en plus des transitions stationnaires :

$$p \xrightarrow{a/b,\blacktriangledown} q \in \Delta \Rightarrow u\,p\,av \to_{\mathcal{M}} u\,q\,bv.$$

théorème 1

Les machines de Turing sont équivalentes aux machines de Turing paresseuses.

Machine paresseuse
Jeux d'instructions différents

théorème 1

Les machines de Turing sont équivalentes aux machines de Turing paresseuses.

Preuve.

☞ Soit $\mathcal{M}$ une machine de Turing. Alors $\mathcal{M}$ est également une machine paresseuse (simplement une machine paresseuse n'ayant aucune transition stationnaire). Donc les machines classiques se réduisent aux machines paresseuses.

□

théorème 1

Les machines de Turing sont équivalentes aux machines de Turing paresseuses.

Preuve.

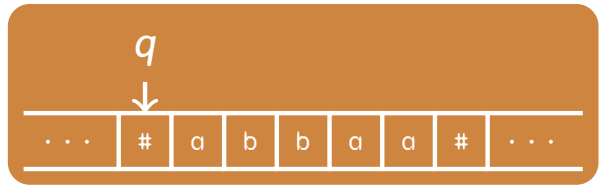
- Soit $\mathcal{M}$ une machine de Turing. Alors $\mathcal{M}$ est également une machine paresseuse (simplement une machine paresseuse n'ayant aucune transition stationnaire). Donc les machines classiques se réduisent aux machines paresseuses.
- Soit $\mathcal{M}$ une machine paresseuse. On va construire une machine classique reconnaissant le langage $\mathcal{L}(\mathcal{M})$, en gardant les transitions allant à gauche ou à droite, et en simulant les transitions stationnaires par une paire de transitions vers la droite puis vers la gauche. Plus précisément, si $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F\rangle$, on construit la machine classique $\mathcal{M}' = \langle Q \cup Q', \Sigma, \Gamma, \Delta', q_0, F\rangle$, avec

$$\begin{aligned} Q' &:= \{q' \mid q \in Q\} \text{ une copie de } Q; \\ \Delta' &:= \{p \xrightarrow{a/b,D} q \mid D \in \{\blacktriangleleft, \blacktriangleright\} \text{ et } p \xrightarrow{a/b,D} q \in \Delta\} \\ &\quad \cup \{p \xrightarrow{a/b,\blacktriangleright} q' \mid p \xrightarrow{a/b,\blacktriangledown} q \in \Delta\} \\ &\quad \cup \{q' \xrightarrow{a/a,\blacktriangleleft} q \mid q \in Q, a \in \Gamma\}. \end{aligned}$$

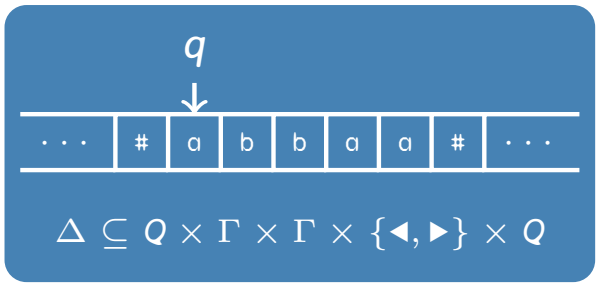
On voit sans peine que $\mathcal{L}(\mathcal{M}') = \mathcal{L}(\mathcal{M})$, et donc que les machines paresseuses se réduisent aux les machines classiques.

□

Modèles équivalents



Initialisation différente



Machines de Turing

Langage préfixé du symbole # (1/4)
Initialisation différente

Dans ce cours, on a défini le langage d'une machine de Turing comme suit :

$$\mathcal{L}(\mathcal{M}) := \{w \in \Sigma^\star \mid q_0\, w \to^*_{\mathcal{M}} C \text{ et } C \text{ acceptante}\}\,.$$

En TD, on utilise une définition différente du langage d'une machine de Turing :

$$\mathcal{L}'(\mathcal{M}) := \{w \in \Sigma^\star \mid q_0\, \#w \to^*_{\mathcal{M}} C \text{ et } C \text{ acceptante}\}\,.$$

Ces deux définitions correspondent à deux modèles de calcul différents :

☞ $\mathscr{M} = \langle \mathbb{MT}, \models \rangle$, avec $w \models \mathcal{M} \Leftrightarrow q_0\, w \to^*_{\mathcal{M}} C$ et C acceptante.

☞ $\mathscr{M}' = \langle \mathbb{MT}, \models' \rangle$, avec $w \models' \mathcal{M} \Leftrightarrow q_0\, \#w \to^*_{\mathcal{M}} C$ et C acceptante.

théorème 2

$\mathscr{M}$ et $\mathscr{M}'$ sont équivalents.

Langage préfixé du symbole # (2/4)
Initialisation différente

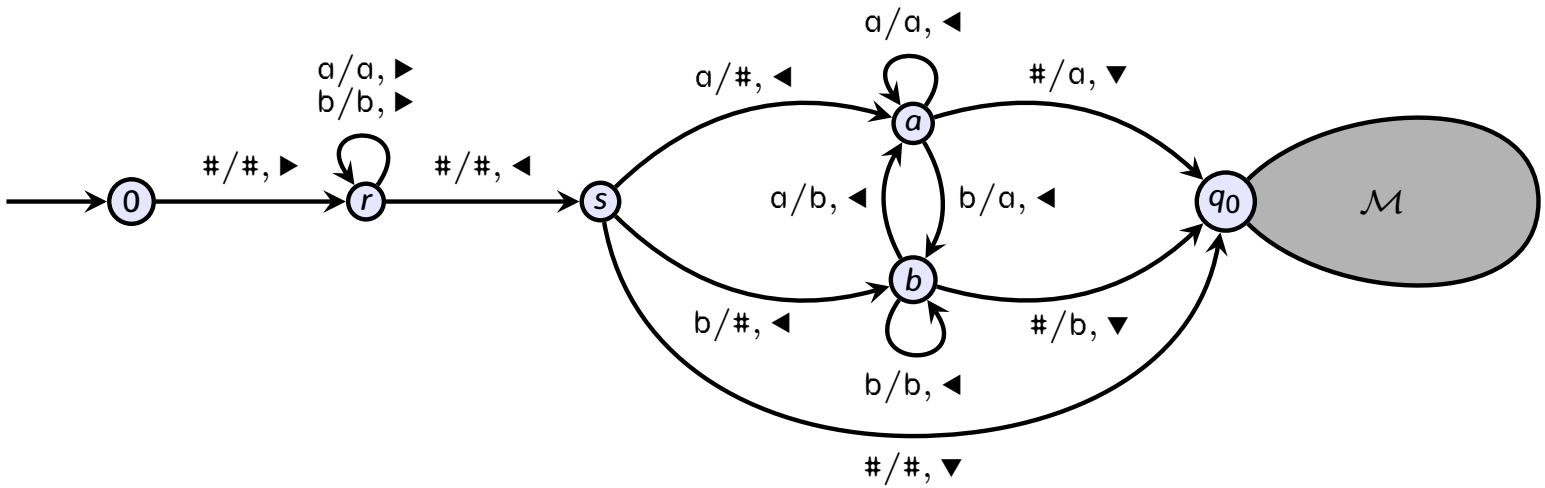
Preuve.

☞ $\mathcal{M} \preceq \mathcal{M}'$.

Soit $\mathcal{M}$ une machine de Turing, on va construire une machine de Turing (paresseuse) $\mathcal{M}'$ avec état initial 0 telle que :

$$q_0\, w \to^*_{\mathcal{M}} C \text{ acceptante} \Leftrightarrow 0\,\#w \to^*_{\mathcal{M}'} C \text{ acceptante}\ .$$

On propose la machine suivante :



On peut vérifier sans peine que :

$$\forall w \in \Sigma^\star,\ 0\,\#w \to^*_{\mathcal{M}'} \#w\,r\,\# \to^*_{\mathcal{M}'} q_0\, w.$$

Langage préfixé du symbole # (3/4)
Initialisation différente

Et que, comme $\mathcal{M}$ est incluse dans $\mathcal{M}'$:

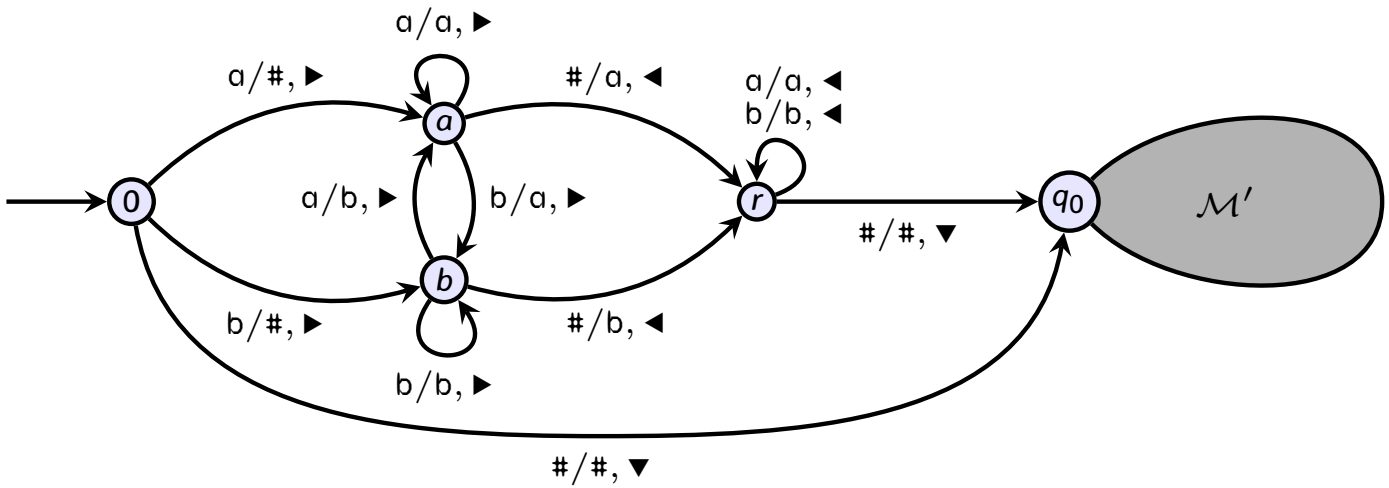
$$q_0\, w \to^*_{\mathcal{M}} C \Leftrightarrow q_0\, w \to^*_{\mathcal{M}'} C \Leftrightarrow 0\, \#w \to^*_{\mathcal{M}'} q_0\, w \to^*_{\mathcal{M}'} C.$$

☞ $\mathcal{M}' \preceq \mathcal{M}$.

Soit $\mathcal{M}'$ une machine de Turing, on va construire une machine de Turing (paresseuse) $\mathcal{M}$ avec état initial 0 telle que :

$$q_0\, \#w \to^*_{\mathcal{M}'} C \text{ acceptante} \Leftrightarrow 0\, w \to^*_{\mathcal{M}} C \text{ acceptante}\ .$$

On propose la machine suivante :



Langage préfixé du symbole # (4/4)
Initialisation différente

On peut vérifier sans peine que :

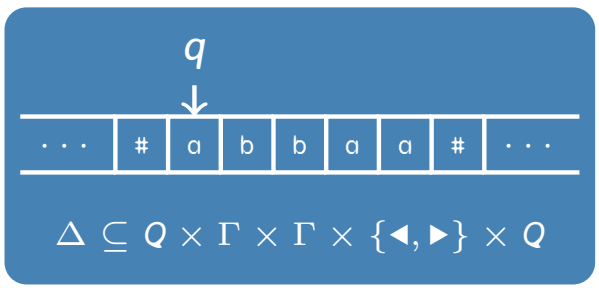
$$\forall w \in \Sigma^\star,\ 0\, w \to^*_{\mathcal{M}} q_0\, \#w.$$

Et que, comme $\mathcal{M}'$ est incluse dans $\mathcal{M}$:

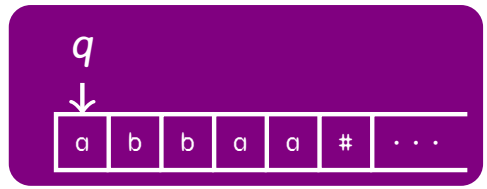
$$q_0\, \#w \to^*_{\mathcal{M}'} C \Leftrightarrow q_0\, \#w \to^*_{\mathcal{M}} C \Leftrightarrow 0\, w \to^*_{\mathcal{M}} q_0\, \#w \to^*_{\mathcal{M}} C.$$

□

Modèles équivalents



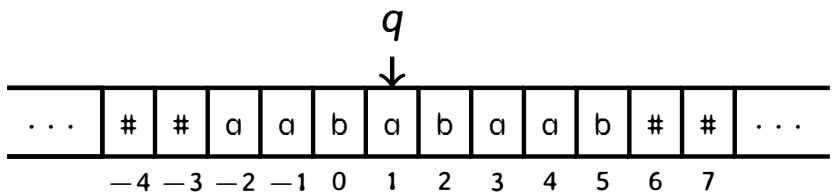
Machines de Turing



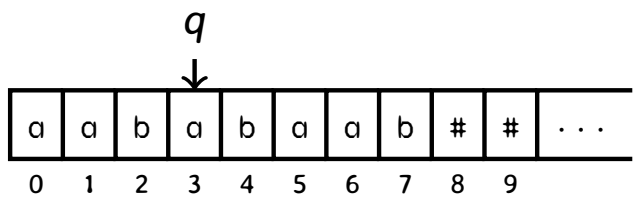
Rubans différents

Ruban semi-infini
Rubans différents

Nous avons défini le ruban d'une machine de Turing comme étant infini des deux côtés ; c'est un tableau indexé par les entiers relatifs :



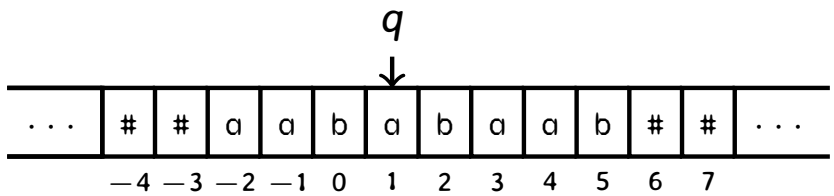
Alternativement, on peut considérer un ruban semi-infini. Un tel ruban indexé par les entiers naturels, est infini vers la droite **mais** arrêté à gauche :



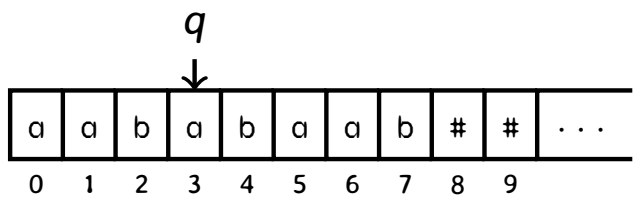
Cela ne change pas la définition de la machine en elle-même, mais change en revanche la définition de configuration. Jusqu'ici, on considérait que $u\,q\,v \approx u\,q\,v\# \approx \#u\,q\,v$. Pour les machines à ruban semi-infini, on a seulement $u\,q\,v \approx u\,q\,v\#$ mais $u\,q\,v \not\approx \#u\,q\,v$.

Ruban semi-infini
Rubans différents

Nous avons défini le ruban d'une machine de Turing comme étant infini des deux côtés ; c'est un tableau indexé par les entiers relatifs :



Alternativement, on peut considérer un ruban semi-infini. Un tel ruban indexé par les entiers naturels, est infini vers la droite **mais** arrêté à gauche :



Cela ne change pas la définition de la machine en elle-même, mais change en revanche la définition de configuration. Jusqu'ici, on considérait que $u\,q\,v \approx u\,q\,v\# \approx \#u\,q\,v$. Pour les machines à ruban semi-infini, on a seulement $u\,q\,v \approx u\,q\,v\#$ mais $u\,q\,v \not\approx \#u\,q\,v$.

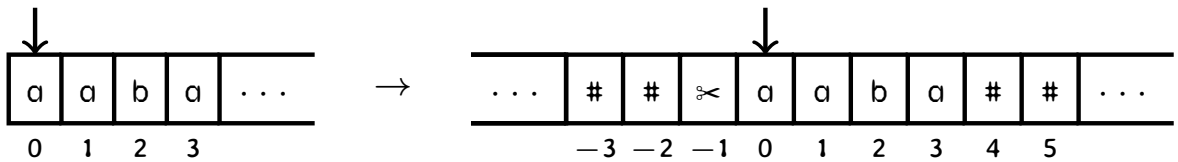
théorème 4

Les machines à ruban semi-infini sont équivalentes aux machines de Turing.

Encoder un ruban semi-infini sur un ruban normal

Rubans différents

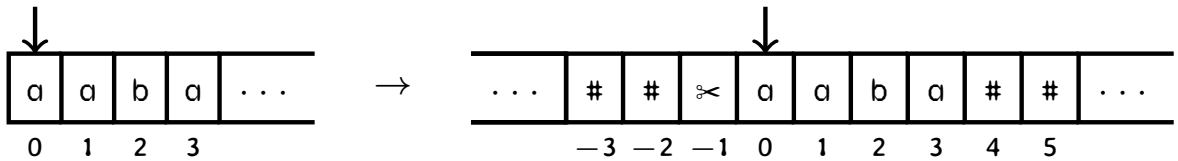
Idée : on va ajouter avant l'entrée un symbole spécial ✂.
Ce symbole restera pendant toute l'exécution à la même place, et toute exécution le visitant sera rejetée.
Ainsi, on s'assure que l'exécution reste toujours sur les cellules d'index positif.



Encoder un ruban semi-infini sur un ruban normal

Rubans différents

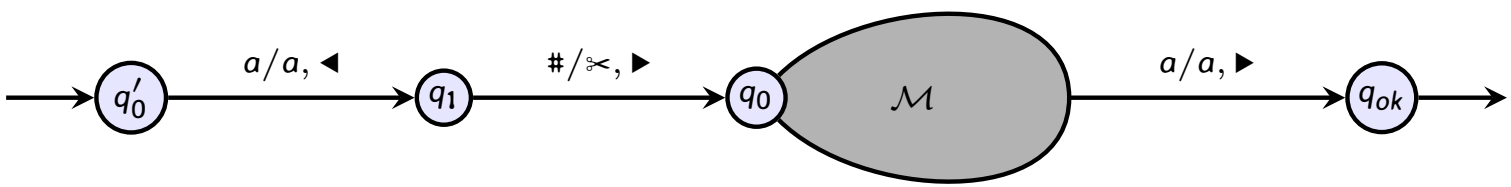
Idée : on va ajouter avant l'entrée un symbole spécial ✂.
Ce symbole restera pendant toute l'exécution à la même place, et toute exécution le visitant sera rejetée.
Ainsi, on s'assure que l'exécution reste toujours sur les cellules d'index positif.



En partant de la machine $\mathcal{M} = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$, on construit une machine

$$\mathcal{M}' = \left\langle Q \cup \{q_0', q_1, q_{ok}\}, \Sigma, \Gamma \cup \{✂\}, \Delta', q_0', \{q_{ok}\} \right\rangle, \text{ avec :}$$

$$\Delta' := \Delta \cup \{q_0' \xrightarrow{a/a,◀} q_1 \mid a \in \Sigma \cup \{\#\}\} \cup \{q_1 \xrightarrow{\#/✂,▶} q_0\}$$
$$\cup \{q \xrightarrow{a/a,▶} q_{ok} \mid q \in F,\ a \neq ✂\}$$

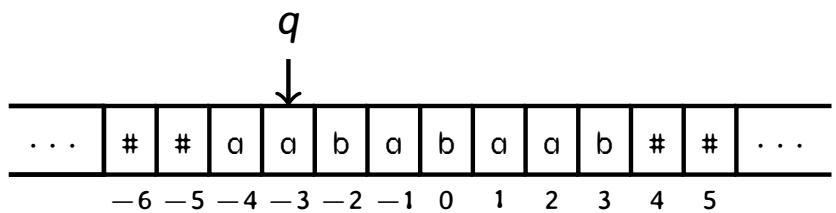


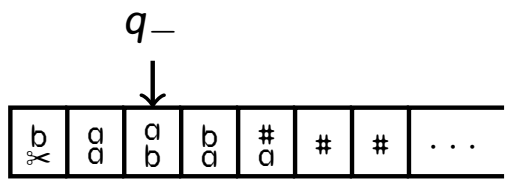
Encoder un ruban normal sur un ruban semi-infini

Rubans différents

Idée : on va représenter dans chaque case $i > 0$ du ruban normal le contenu des cases i et $-i$. Dans la case 0, on stocke le contenu de la case 0 du ruban bi-infini avec un symbole spécial $\between$.
Pour cela, on passe de l'alphabet Γ à l'alphabet

$$\Sigma \cup \left(\left\{ \begin{matrix} a \\ b \end{matrix} \;\middle|\; a, b \in \Gamma \right\} \setminus \left\{ \begin{matrix} \# \\ \# \end{matrix} \right\} \right) \cup \left\{ \begin{matrix} a \\ \between \end{matrix} \;\middle|\; a \in \Gamma \right\} \cup \{\#\}.$$



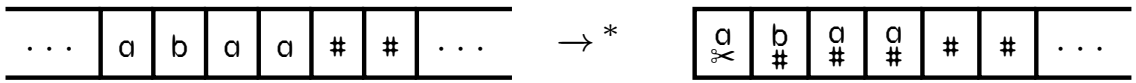


Encoder un ruban bi-infini sur un ruban normal

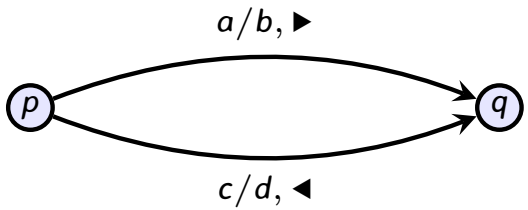
Rubans différents

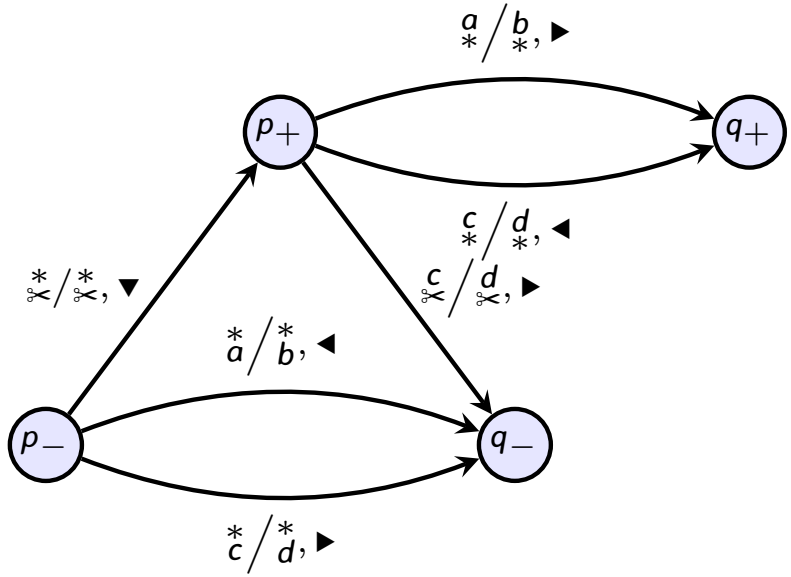
À partir d'une machine $\mathcal{M}$, on construit une machine $\mathcal{M}'$ qui procède comme suit sur une entrée $w \in \Sigma^\star$:

1) on lit le mot d'entrée, et on le reformate dans le nouvel alphabet :



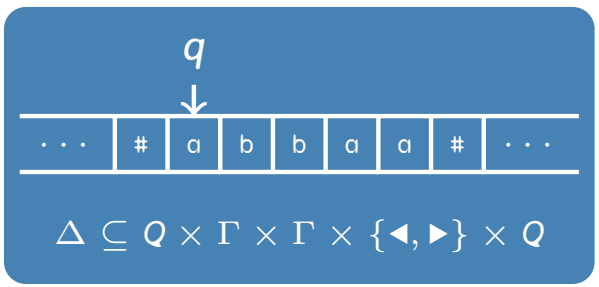
2) on simule la machine $\mathcal{M}$, avec pour chaque état $q \in Q$ deux états :
   - q_+ qui sera utilisé pour les configurations où la tête de lecture est sur une case positive ;
   - q_- qui sera utilisé pour les configurations où la tête de lecture est sur une case négative.





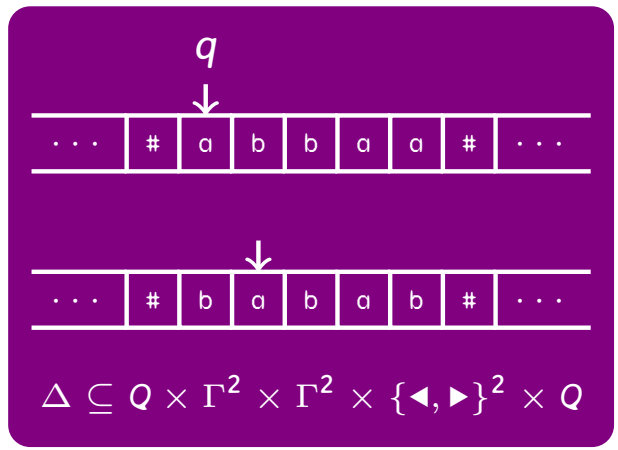
3) on accepte lorsequ'on se trouve sur un état q_+ ou q_- tel que $q \in F$.

Modèles équivalents



Machines de Turing

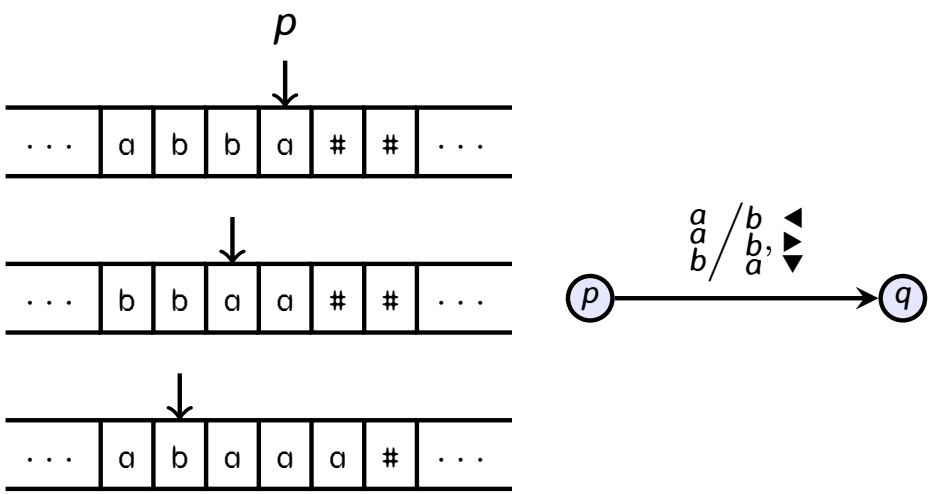
Rubans différents



Machine multi-ruban
Rubans différents

- On peut également considérer des machines qui opèrent sur plusieurs rubans.
- Elles possèdent une tête de lecture par ruban.
- Chaque transition consulte le contenu des cases pointées par chacune des têtes de lecture, peut écrire dans chacune de ses cases, et déplacer chaque tête.

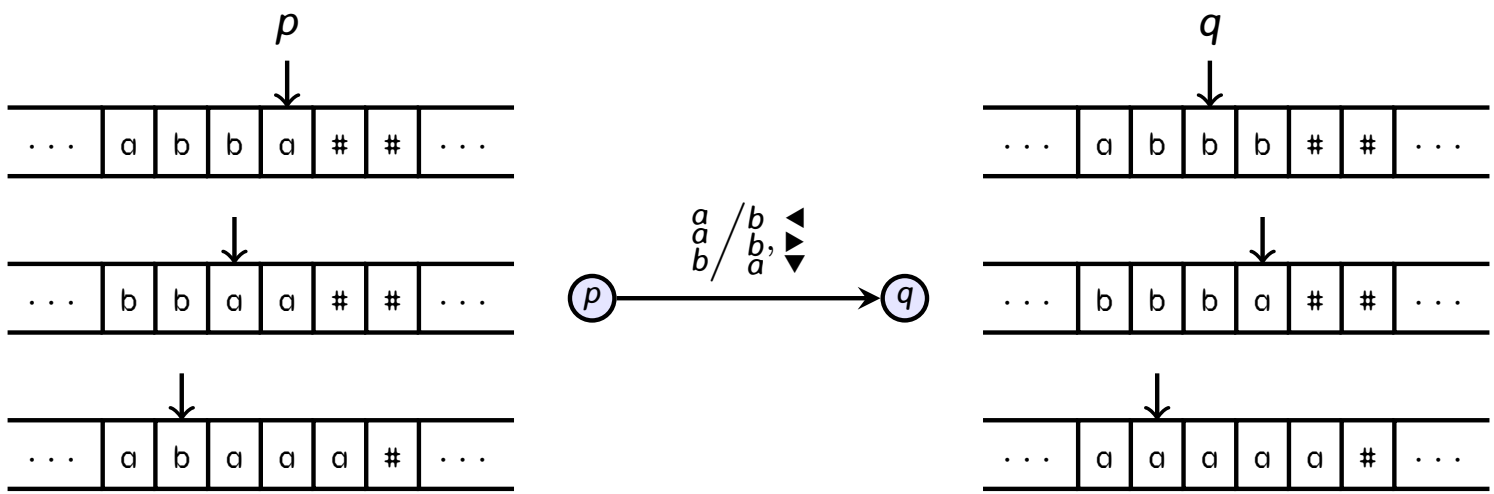
$$\Delta \subseteq Q \times \Gamma^3 \times \Gamma^3 \times \{\blacktriangleleft, \blacktriangleright, \blacktriangledown\}^3 \times Q$$



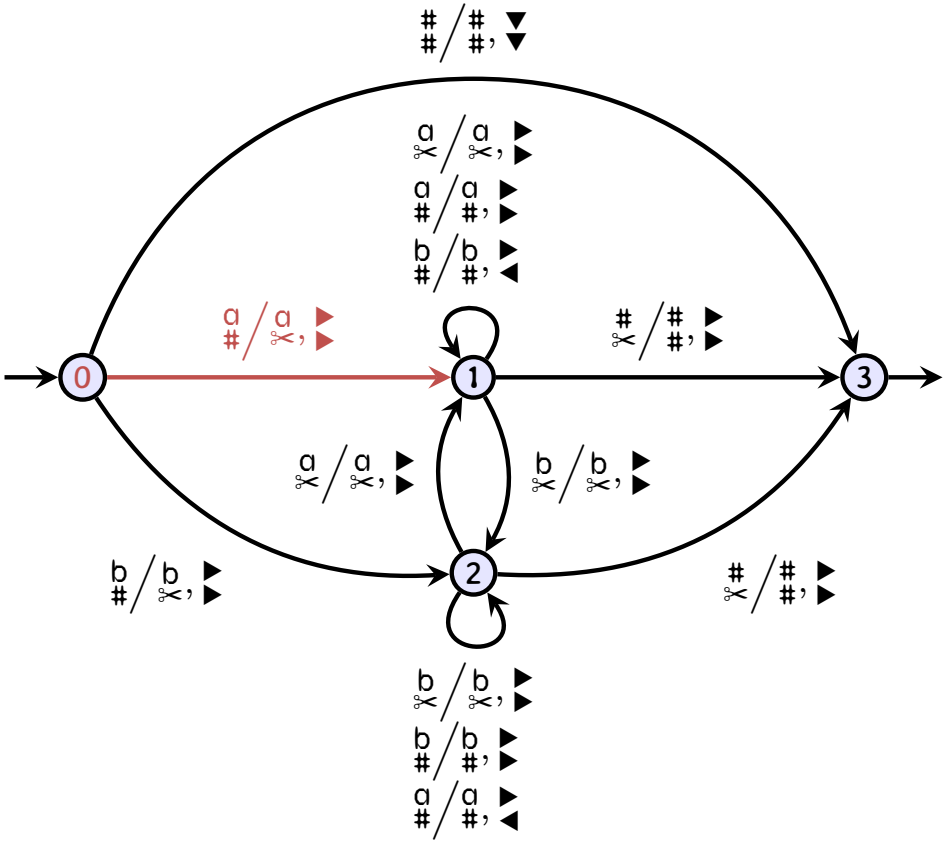
Machine multi-ruban
Rubans différents

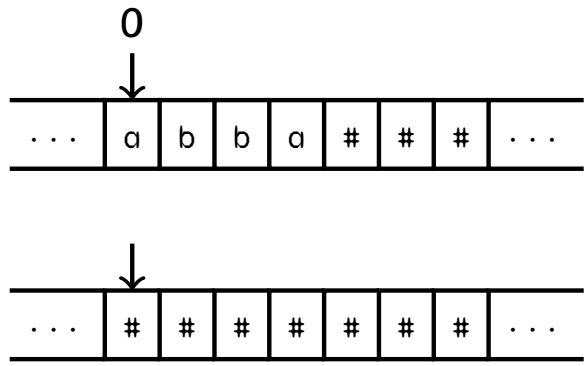
- On peut également considérer des machines qui opèrent sur plusieurs rubans.
- Elles possèdent une tête de lecture par ruban.
- Chaque transition consulte le contenu des cases pointées par chacune des têtes de lecture, peut écrire dans chacune de ses cases, et déplacer chaque tête.

$$\Delta \subseteq Q \times \Gamma^3 \times \Gamma^3 \times \{\blacktriangleleft, \blacktriangleright, \blacktriangledown\}^3 \times Q$$



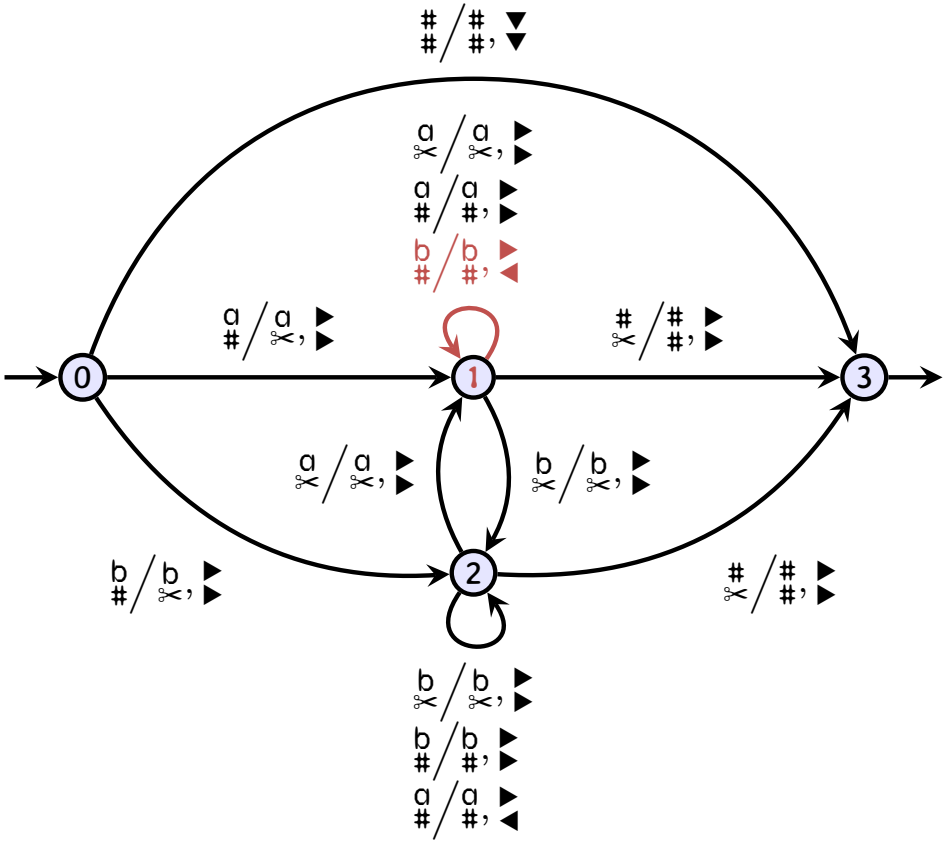
Exemple de machine multi-ruban
Rubans différents

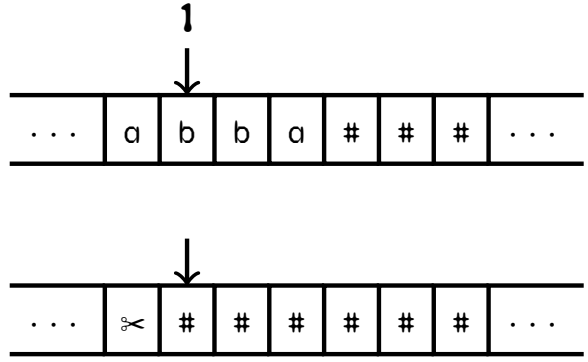




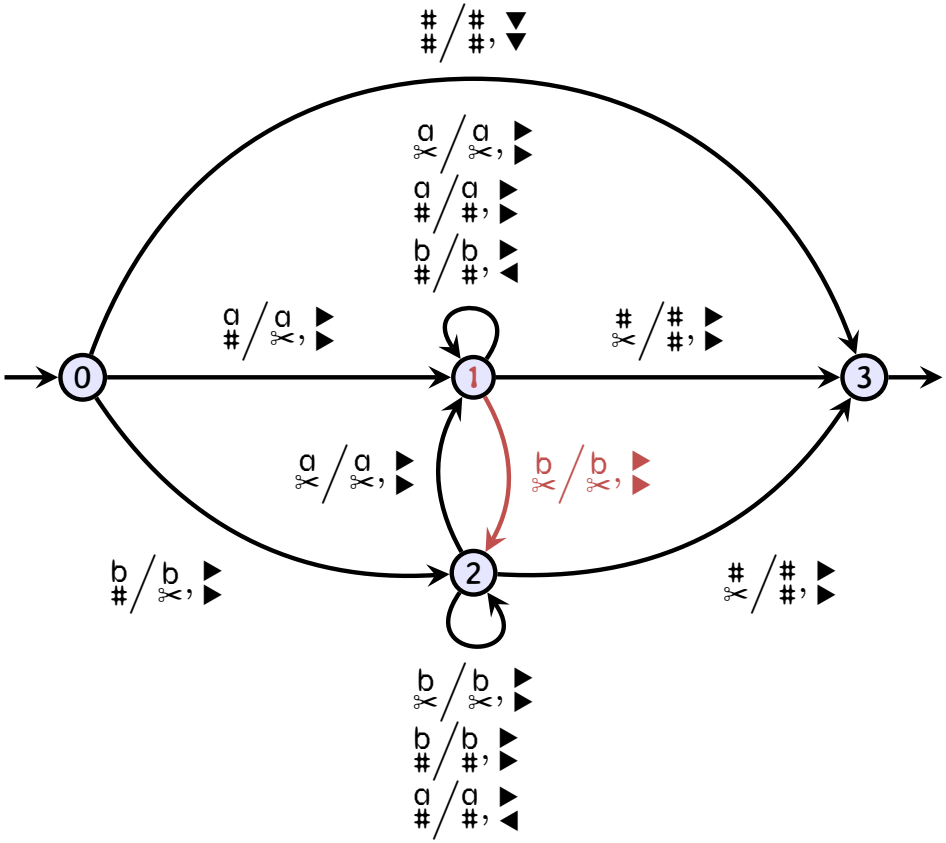
Exemple de machine multi-ruban

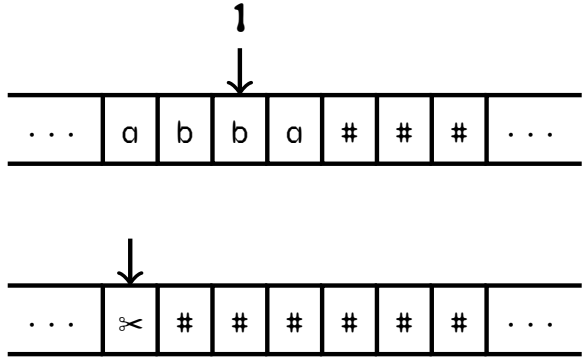
Rubans différents





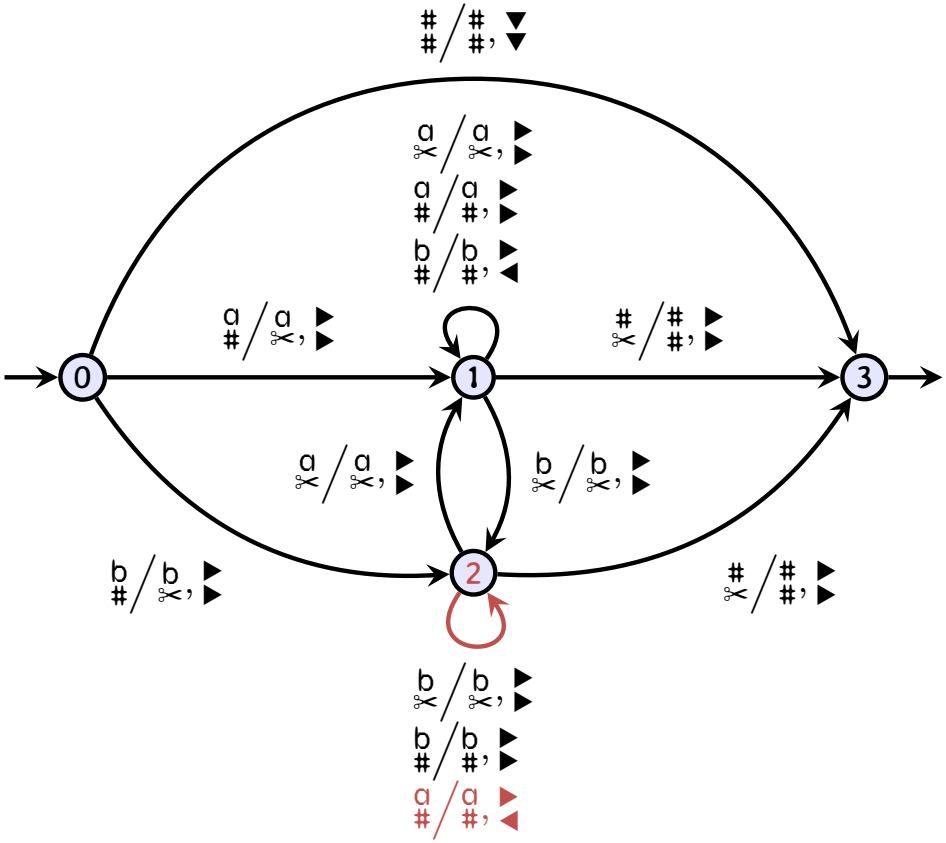
Exemple de machine multi-ruban
Rubans différents

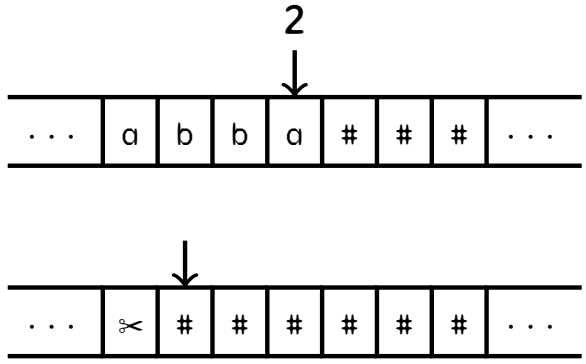




Exemple de machine multi-ruban

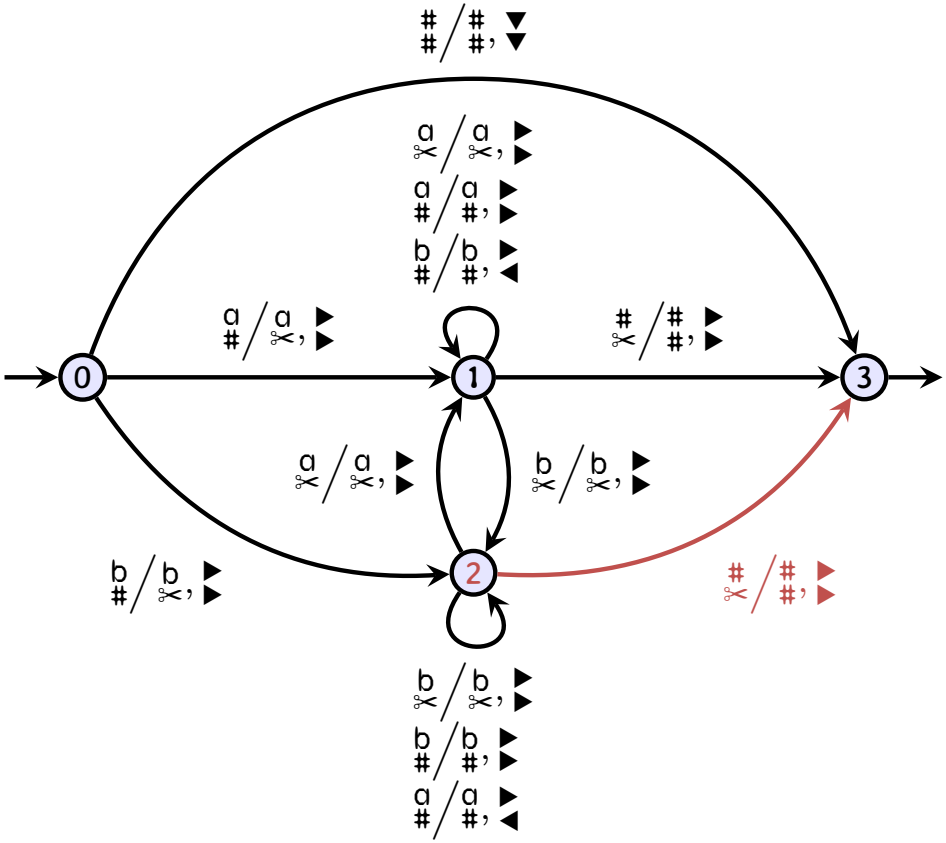
Rubans différents

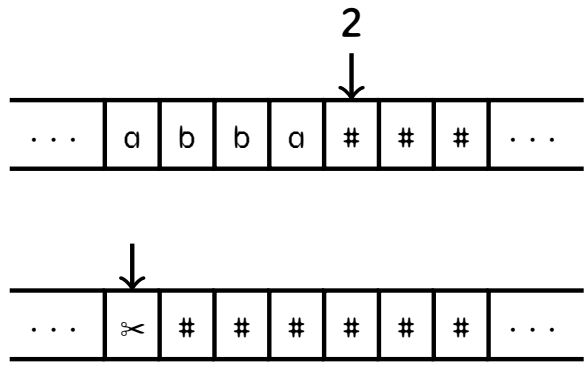




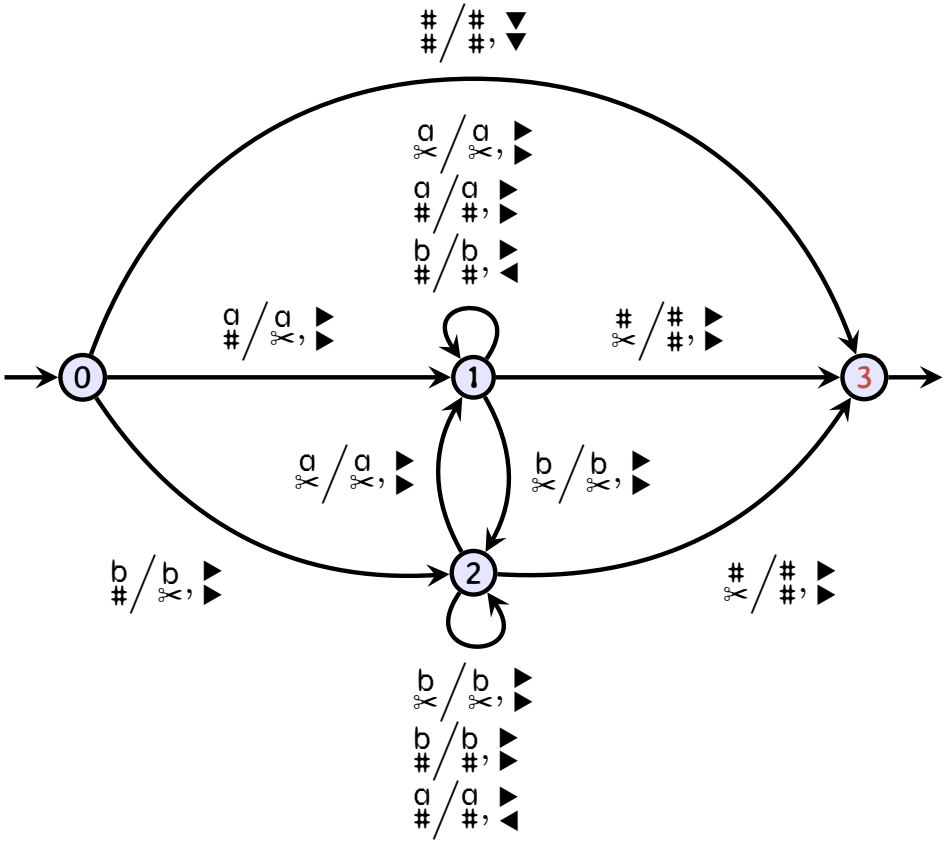
Exemple de machine multi-ruban

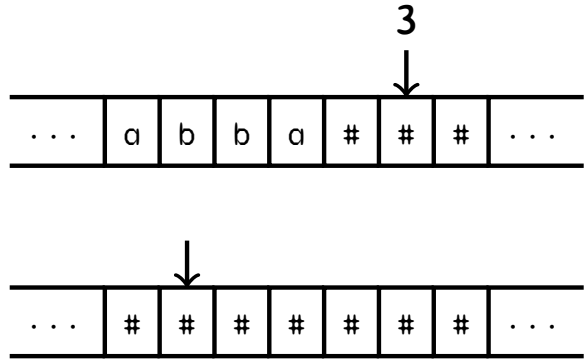
Rubans différents





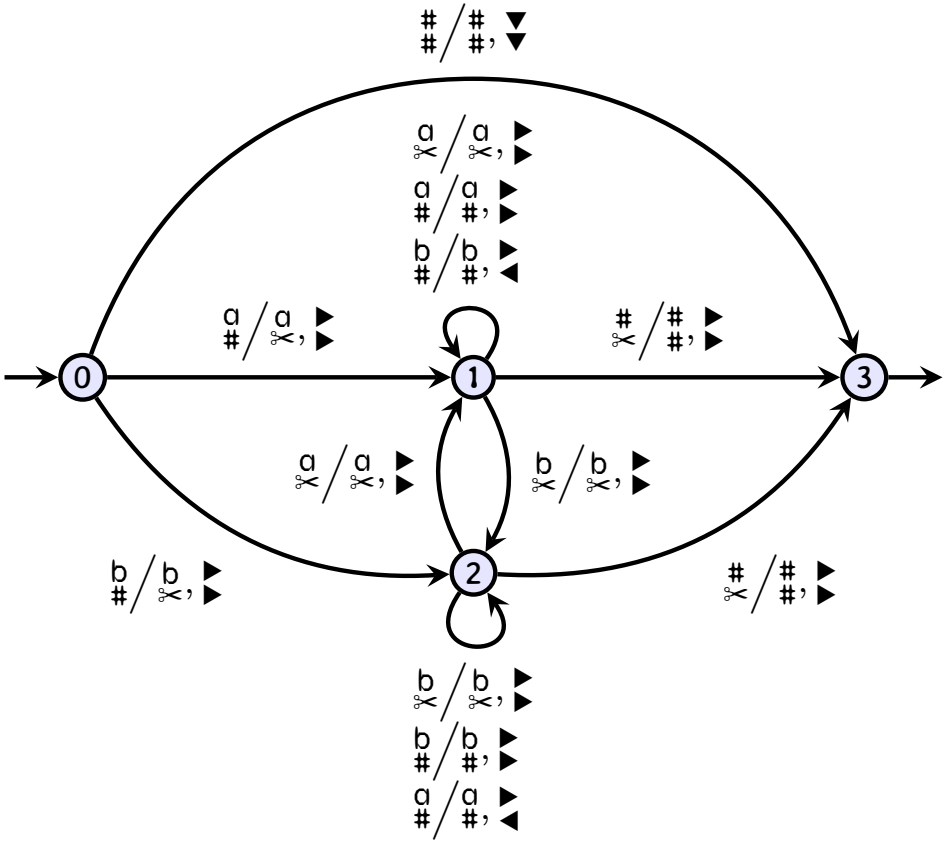
Exemple de machine multi-ruban
Rubans différents

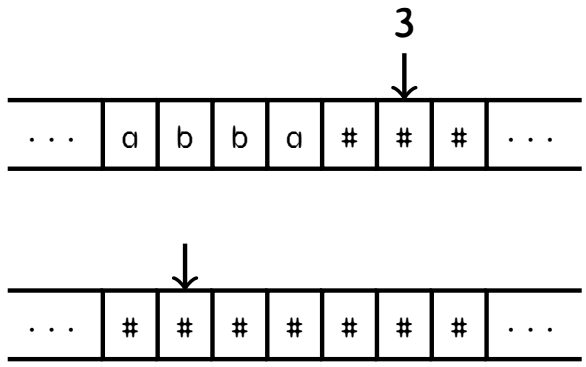




Exemple de machine multi-ruban

Rubans différents





Exercice

Quel est le langage reconnu par cette machine ?

Machine multi-ruban
Rubans différents

exercice

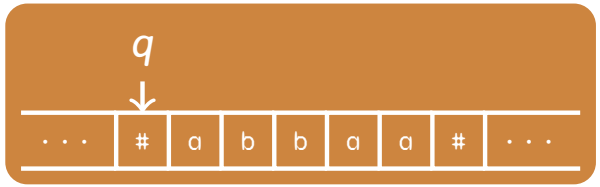
Montrer que les machines à plusieurs rubans sont équivalentes aux machines à un ruban.

Modèles équivalents

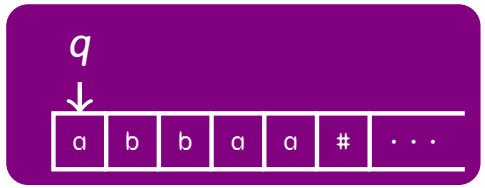
$\Delta \subseteq Q \times \Gamma \times \Gamma \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\} \times Q$

Jeux d'instructions différents

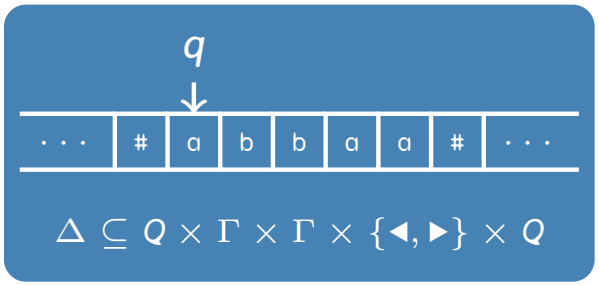
$\Delta \subseteq Q \times \Gamma \times (\Gamma + \{\blacktriangleleft, \blacktriangleright\}) \times Q$



Initialisation différente



Rubans différents



Machines de Turing