Calculabilité TD5

Antonio E. Porreca
aeporreca.org/calculabilite

Calculabilité TD5

Antonio E. Porreca
aeporreca.org/calculabilite

Examen session 1
2018–2019

Examen session 1
2018–2019

Exercice 1
Notions de base

Exercice 1
Notions de base

Exercice 1.1

Exercice 1.1

Exercice 1.1

Compléter la phrase suivante :

Un langage L est récursif si et seulement si…

Exercice 1.1

Compléter la phrase suivante :
Un langage L est récursif si et seulement si…

- …il existe une machine de Turing M telle que

Exercice 1.1

Compléter la phrase suivante :
Un langage L est récursif si et seulement si…

- …il existe une machine de Turing M telle que
    - M accepte tous les mots $w \in L$

Exercice 1.1

Compléter la phrase suivante :
Un langage L est récursif si et seulement si…

- …il existe une machine de Turing M telle que
  - M accepte tous les mots $w \in L$
  - M rejette tous les mots $w \notin L$

Exercice 1.1

Compléter la phrase suivante :
Un langage L est récursif si et seulement si…

- …il existe une machine de Turing M telle que
  - M accepte tous les mots $w \in L$
  - M rejette tous les mots $w \notin L$
  - M s'arrête toujours

Exercice 1.2

Exercice 1.2

Exercice 1.2

Compléter la phrase suivante :

Un langage L est récursivement énumérable si et seulement si…

Exercice 1.2

Compléter la phrase suivante :
Un langage L est récursivement énumérable si et seulement si…

- …il existe une machine de Turing M telle que

Exercice 1.2

Compléter la phrase suivante :
Un langage L est récursivement énumérable si et seulement si…

- …il existe une machine de Turing M telle que
  - M accepte tous les mots $w \in L$

Exercice 1.2

Compléter la phrase suivante :
Un langage L est récursivement énumérable si et seulement si…

- …il existe une machine de Turing M telle que
  - M accepte tous les mots $w \in L$
  - M rejette tous les mots $w \notin L$

Exercice 1.2

Compléter la phrase suivante :
Un langage L est récursivement énumérable si et seulement si…

- …il existe une machine de Turing M telle que
  - M accepte tous les mots $w \in L$
  - M rejette tous les mots $w \notin L$
- ⚠️ On ne demande pas que M s'arrête toujours : les mots sur lesquels M ne s'arrête pas comptent aussi comme rejetés

Exercice 1.3

Exercice 1.3

Exercice 1.3

Dans la définition des machines de Turing, pourquoi impose-t-on que $B \in \Gamma \setminus \Sigma$? (avec B le symbole blanc, Γ l'alphabet de ruban, et Σ l'alphabet d'entrée)

Exercice 1.3

Dans la définition des machines de Turing, pourquoi impose-t-on que $B \in \Gamma \setminus \Sigma$? (avec B le symbole blanc, Γ l'alphabet de ruban, et Σ l'alphabet d'entrée)

- On utilise le symbole B pour trouver la fin du mot d'entrée

Exercice 1.3

Dans la définition des machines de Turing, pourquoi impose-t-on que $B \in \Gamma \setminus \Sigma$? (avec B le symbole blanc, Γ l'alphabet de ruban, et Σ l'alphabet d'entrée)

- On utilise le symbole B pour trouver la fin du mot d'entrée
- Si B était un symbole d'entrée, on ne saurait pas où arrêter de lire le ruban d'entrée de la machine

Exercice 1.4

Exercice 1.4

Exercice 1.4

Soient L_1 et L_2 deux langages. Montrer que si $L_1 \leq_m^T L_2$ et L_2 est récursif, alors L_1 est récursif.

Exercice 1.4

Soient L_1 et L_2 deux langages. Montrer que si $L_1 \leq_m^T L_2$ et L_2 est récursif, alors L_1 est récursif.

- Si L_2 est récursif, alors il existe une machine M_2 qui reconnaît L_2 et qui s'arrête toujours

Exercice 1.4

Soient L_1 et L_2 deux langages. Montrer que si $L_1 \leq_m^T L_2$ et L_2 est récursif, alors L_1 est récursif.

- Si L_2 est récursif, alors il existe une machine M_2 qui reconnaît L_2 et qui s'arrête toujours
- Si $L_1 \leq_m^T L_2$ alors il existe une machine M_f qui calcule la réduction f: $\Sigma_1^\star \rightarrow \Sigma_2^\star$ tel que $x \in L_1$ ssi $f(x) \in L_2$ (donc M_f s'arrête toujours)

Exercice 1.4

Exercice 1.4

- Voici une machine M_1 qui reconnaît L_1 et s'arrête toujours :

Exercice 1.4

- Voici une machine M_1 qui reconnaît L_1 et s'arrête toujours :

$M_1(x) =$

simuler $M_f(x)$ en obtenant $f(x)$;

simuler $M_2\big(f(x)\big)$ et renvoyer le même résultat

Exercice 1.4

- Voici une machine M_1 qui reconnaît L_1 et s'arrête toujours :

  $M_1(x) =$

  simuler $M_f(x)$ en obtenant $f(x)$;

  simuler $M_2\big(f(x)\big)$ et renvoyer le même résultat

- M_1 accepte x ssi M_2 accepte $f(x)$

Exercice 1.4

- Voici une machine M_1 qui reconnaît L_1 et s'arrête toujours :

$$M_1(x) =$$

simuler $M_f(x)$ en obtenant $f(x)$;
simuler $M_2\big(f(x)\big)$ et renvoyer le même résultat

- M_1 accepte x ssi M_2 accepte $f(x)$
- M_2 accepte $f(x)$ ssi $f(x) \in L_2$ ssi $x \in L_1$

Exercice 1.4

- Voici une machine M_1 qui reconnaît L_1 et s'arrête toujours :

  $M_1(x) =$

  simuler $M_f(x)$ en obtenant $f(x)$;

  simuler $M_2\big(f(x)\big)$ et renvoyer le même résultat

- M_1 accepte x ssi M_2 accepte $f(x)$
- M_2 accepte $f(x)$ ssi $f(x) \in L_2$ ssi $x \in L_1$
- Donc M_1 reconnaît L_1 en s'arrêtant toujours, donc L_1 est récursif

Exercice 1.5

Exercice 1.5

Exercice 1.5

Parmi les deux affirmations suivantes, laquelle est correcte ?

(a) Si L est récursif, alors L est récursivement énumérable.

(b) Si L est récursivement énumérable, alors L est récursif.

Exercice 1.5

Parmi les deux affirmations suivantes, laquelle est correcte ?
(a) Si L est récursif, alors L est récursivement énumérable.
(b) Si L est récursivement énumérable, alors L est récursif.

- Si L est récursif alors il existe une MT M qui accepte les mots $w \in L$, rejette les mots $w \notin L$ et s'arrête toujours

Exercice 1.5

Parmi les deux affirmations suivantes, laquelle est correcte ?
(a) Si L est récursif, alors L est récursivement énumérable.
(b) Si L est récursivement énumérable, alors L est récursif.

- Si L est récursif alors il existe une MT M qui accepte les mots $w \in L$, rejette les mots $w \notin L$ et s'arrête toujours

- Donc, en particulier, c'est vrai que M accepte les mots $w \in L$ et rejette les mots $w \notin L$, ce qui implique que L est récursivement énumérable

Exercice 1.5

Parmi les deux affirmations suivantes, laquelle est correcte ?
(a) Si L est récursif, alors L est récursivement énumérable.
(b) Si L est récursivement énumérable, alors L est récursif.

- Si L est récursif alors il existe une MT M qui accepte les mots $w \in L$, rejette les mots $w \notin L$ et s'arrête toujours

- Donc, en particulier, c'est vrai que M accepte les mots $w \in L$ et rejette les mots $w \notin L$, ce qui implique que L est récursivement énumérable

- Donc (a) est la bonne réponse ; (b) n'est pas vrai en général, parce qu'on ne demande pas que la machine s'arrête toujours

Exercice 1.6

Exercice 1.6

Exercice 1.6

Donner un exemple de langage non récursivement énumérable, différent de

$$L_{\bar{u}} = \{\langle M \rangle \# w \mid M \text{ n'accepte pas } w\}$$

Exercice 1.6

Donner un exemple de langage non récursivement énumérable, différent de

$$L_{\bar{u}} = \{\langle M \rangle \# w \mid M \text{ n'accepte pas } w\}$$

- $L_d = \{\langle M \rangle \mid M \text{ n'accepte pas } \langle M \rangle\}$ (voir notes du CM3)

Exercice 1.6

Donner un exemple de langage non récursivement énumérable, différent de

$$L_{\bar{u}} = \{\langle M \rangle \# w \mid M \text{ n'accepte pas } w\}$$

- $L_d = \{\langle M \rangle \mid M \text{ n'accepte pas } \langle M \rangle\}$ (voir notes du CM3)
- Sinon, rappel : $L \in \mathbf{R}$ ssi $L \in \mathbf{RE}$ et $\bar{L} \in \mathbf{RE}$;
  donc on peut prendre le complémentaire de n'importe quel langage L qui soit $\mathbf{RE}$ mais pas $\mathbf{R}$

Exercice 1.7

Exercice 1.7

Exercice 1.7

Donner si possible un exemple de langage non récursivement énumérable mais récursif

Exercice 1.7

Donner si possible un exemple de langage non récursivement énumérable mais récursif

- C’est impossible, on a vu dans l’exercice 1.5 que chaque langage $\mathbf{R}$ est aussi $\mathbf{RE}$

Exercice 1.8

Exercice 1.8

Exercice 1.8

Donner si possible un exemple de langage non récursif
mais récursivement énumérable

Exercice 1.8

Donner si possible un exemple de langage non récursif
mais récursivement énumérable

- $L_u = \{\langle M \rangle \# w \mid M \text{ accepte } w\}$, voici une machine qui le reconnaît :

Exercice 1.8

Donner si possible un exemple de langage non récursif
mais récursivement énumérable

- $L_u = \{\langle M \rangle \# w \mid M \text{ accepte } w\}$, voici une machine qui le reconnaît :

$$M_u\big(\langle M \rangle \# w\big) = \text{simuler } M \text{ sur } w \text{ et renvoyer le même résultat}$$

Exercice 1.8

Donner si possible un exemple de langage non récursif
mais récursivement énumérable

- $L_u = \{\langle M \rangle \# w \mid M$ accepte $w\}$, voici une machine qui le reconnaît :

$$M_u\big(\langle M \rangle \# w\big) = \text{simuler } M \text{ sur } w \text{ et renvoyer le même résultat}$$

- Si M accepte w alors M_u accepte $\langle M \rangle \# w$

Exercice 1.8

Donner si possible un exemple de langage non récursif
mais récursivement énumérable

- $L_u = \{\langle M \rangle \# w \mid M \text{ accepte } w\}$, voici une machine qui le reconnaît :

$$M_u\big(\langle M \rangle \# w\big) = \text{simuler } M \text{ sur } w \text{ et renvoyer le même résultat}$$

- Si M accepte w alors M_u accepte $\langle M \rangle \# w$

- Si M rejette w en s'arrêtant, alors M_u rejette $\langle M \rangle \# w$ en s'arrêtant

Exercice 1.8

Donner si possible un exemple de langage non récursif
mais récursivement énumérable

- $L_u = \{\langle M \rangle \# w \mid M \text{ accepte } w\}$, voici une machine qui le reconnaît :

  $M_u\big(\langle M \rangle \# w\big)$ = simuler M sur w et renvoyer le même résultat

- Si M accepte w alors M_u accepte $\langle M \rangle \# w$

- Si M rejette w en s'arrêtant, alors M_u rejette $\langle M \rangle \# w$ en s'arrêtant

- Si M ne s'arrête pas sur w, alors M_u non plus sur $\langle M \rangle \# w$

Exercice 1.8

Donner si possible un exemple de langage non récursif
mais récursivement énumérable

- $L_u = \{\langle M \rangle \# w \mid M \text{ accepte } w\}$, voici une machine qui le reconnaît :

$$M_u\big(\langle M \rangle \# w\big) = \text{simuler } M \text{ sur } w \text{ et renvoyer le même résultat}$$

- Si M accepte w alors M_u accepte $\langle M \rangle \# w$
- Si M rejette w en s'arrêtant, alors M_u rejette $\langle M \rangle \# w$ en s'arrêtant
- Si M ne s'arrête pas sur w, alors M_u non plus sur $\langle M \rangle \# w$
- M_u reconnaît L_u, qui est donc $\mathbf{RE}$, mais il n'est pas $\mathbf{R}$ (Corollaire 3, notes CM3)

Exercice 2
Machine de Turing

Exercice 2
Machine de Turing

Exercice 2.1

Exercice 2.1

Exercice 2.1

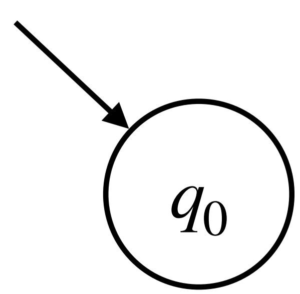
Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$

Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

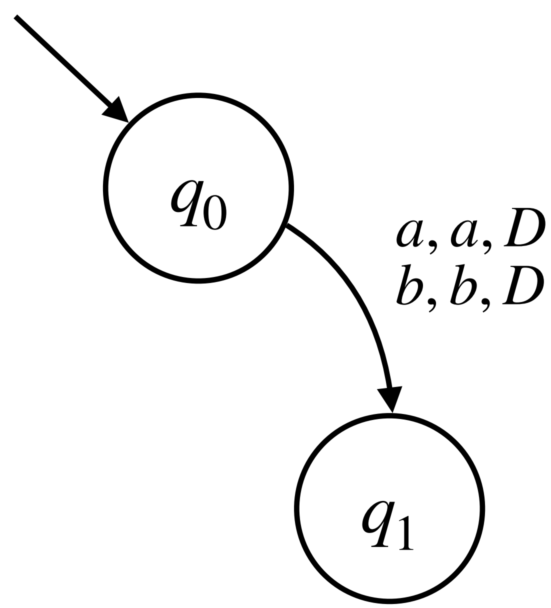
$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

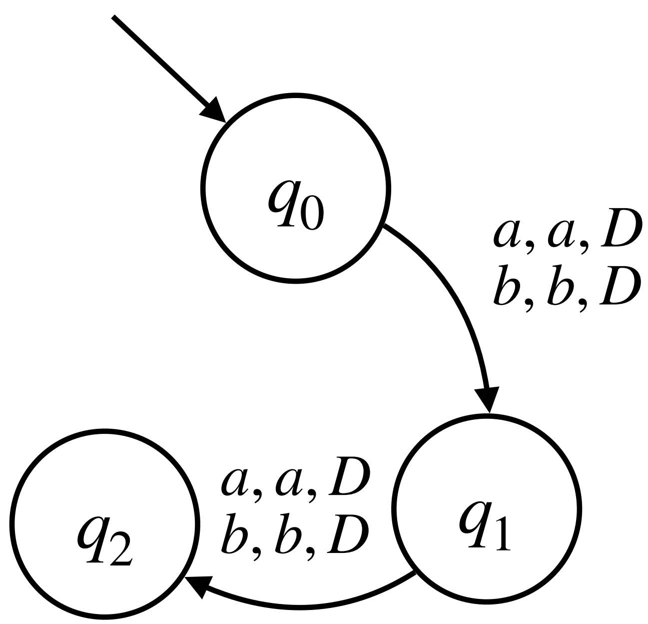
$L_1 = \left\{w_1w_2\cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a\right\}$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

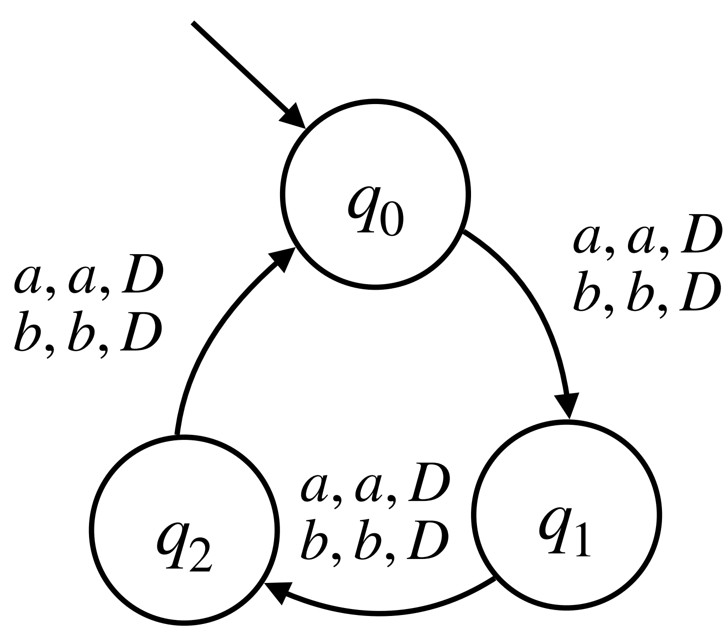
$$L_1 = \{w_1 w_2 \cdots w_n \in \{a,b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

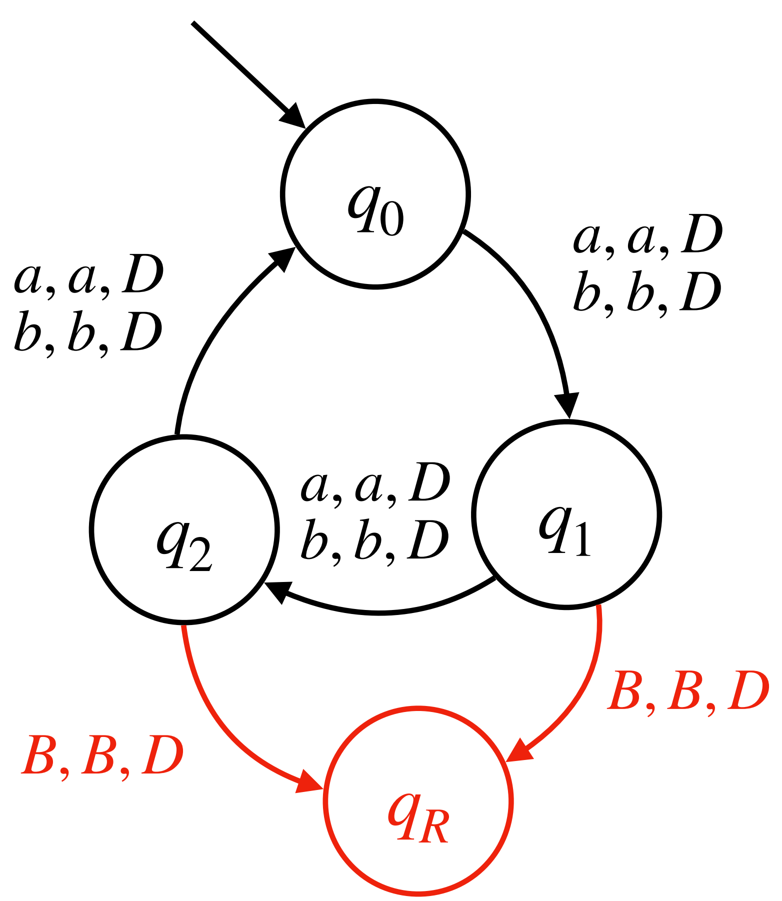
$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

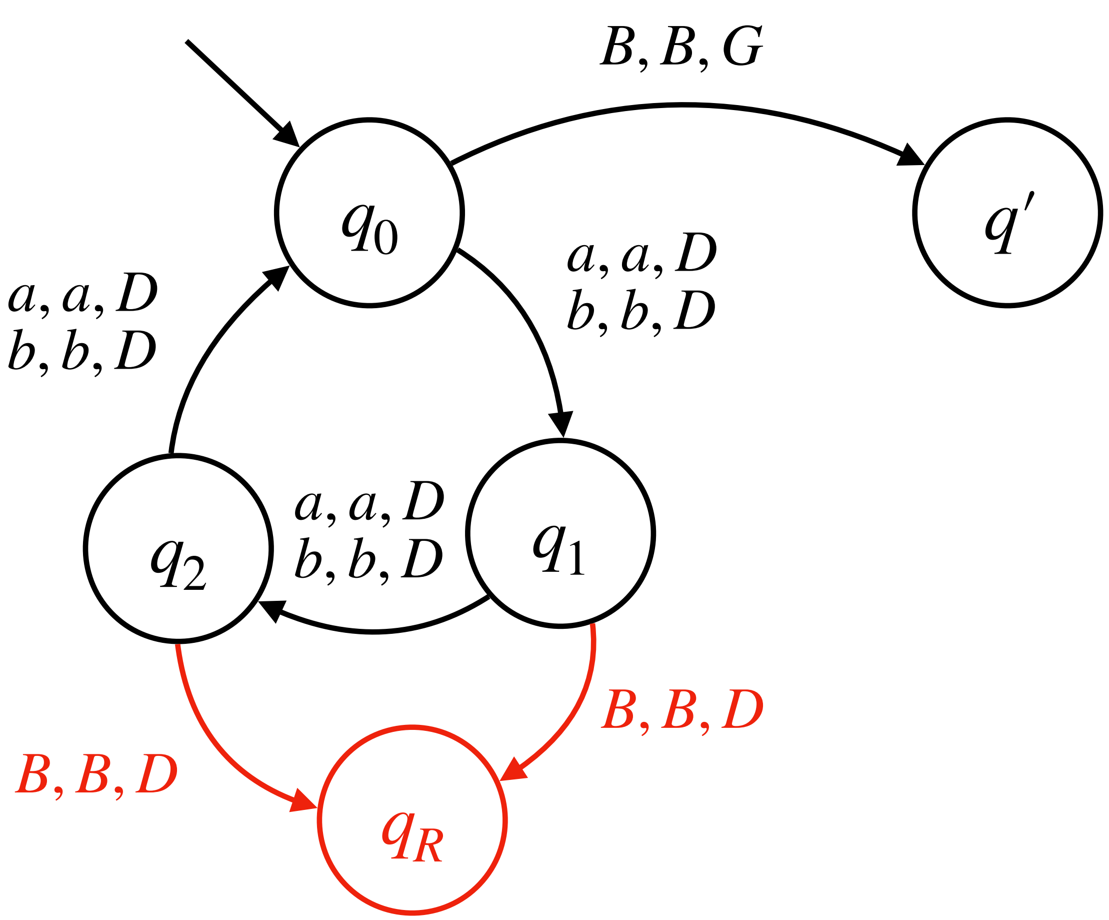
$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

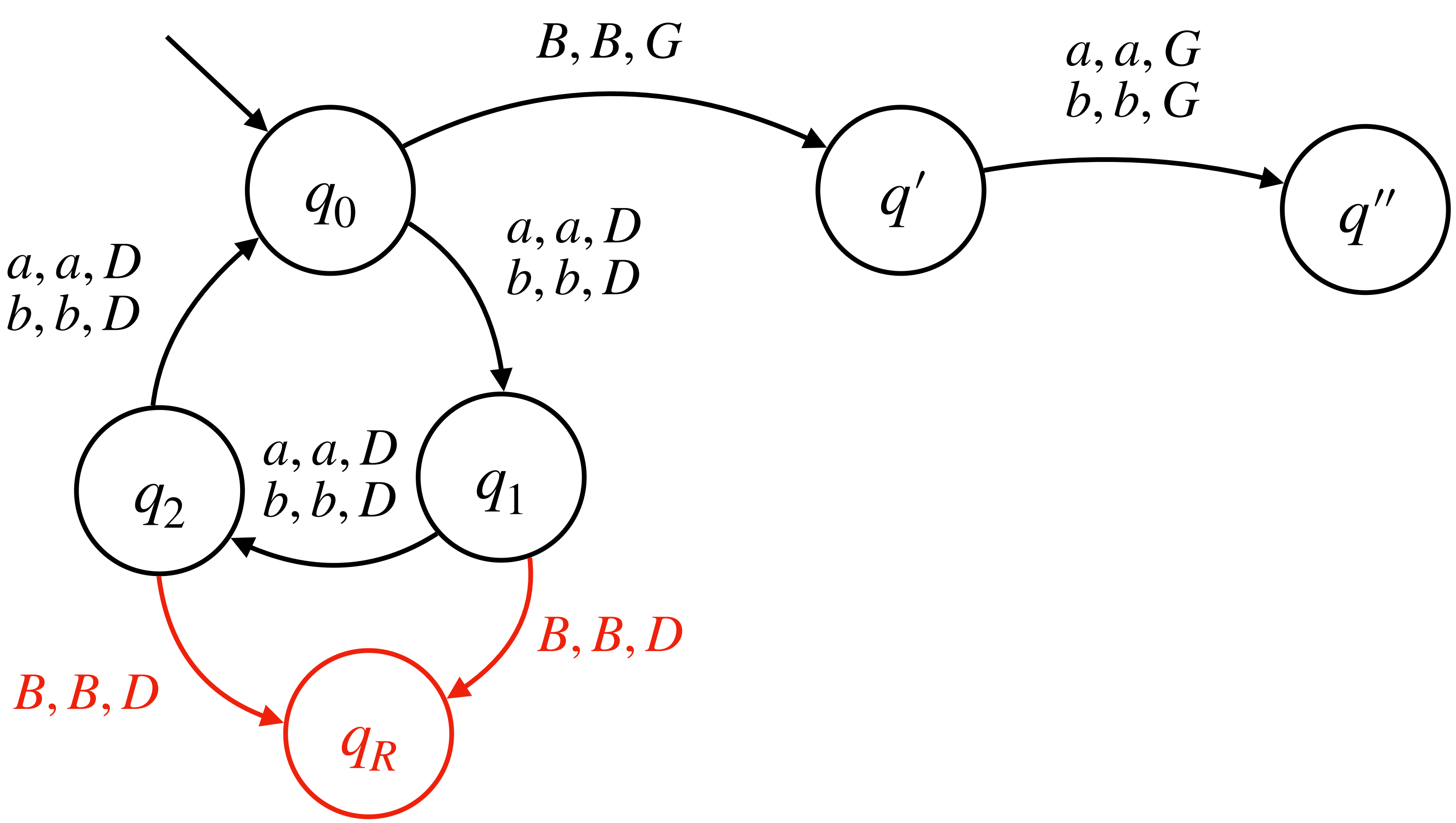
$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

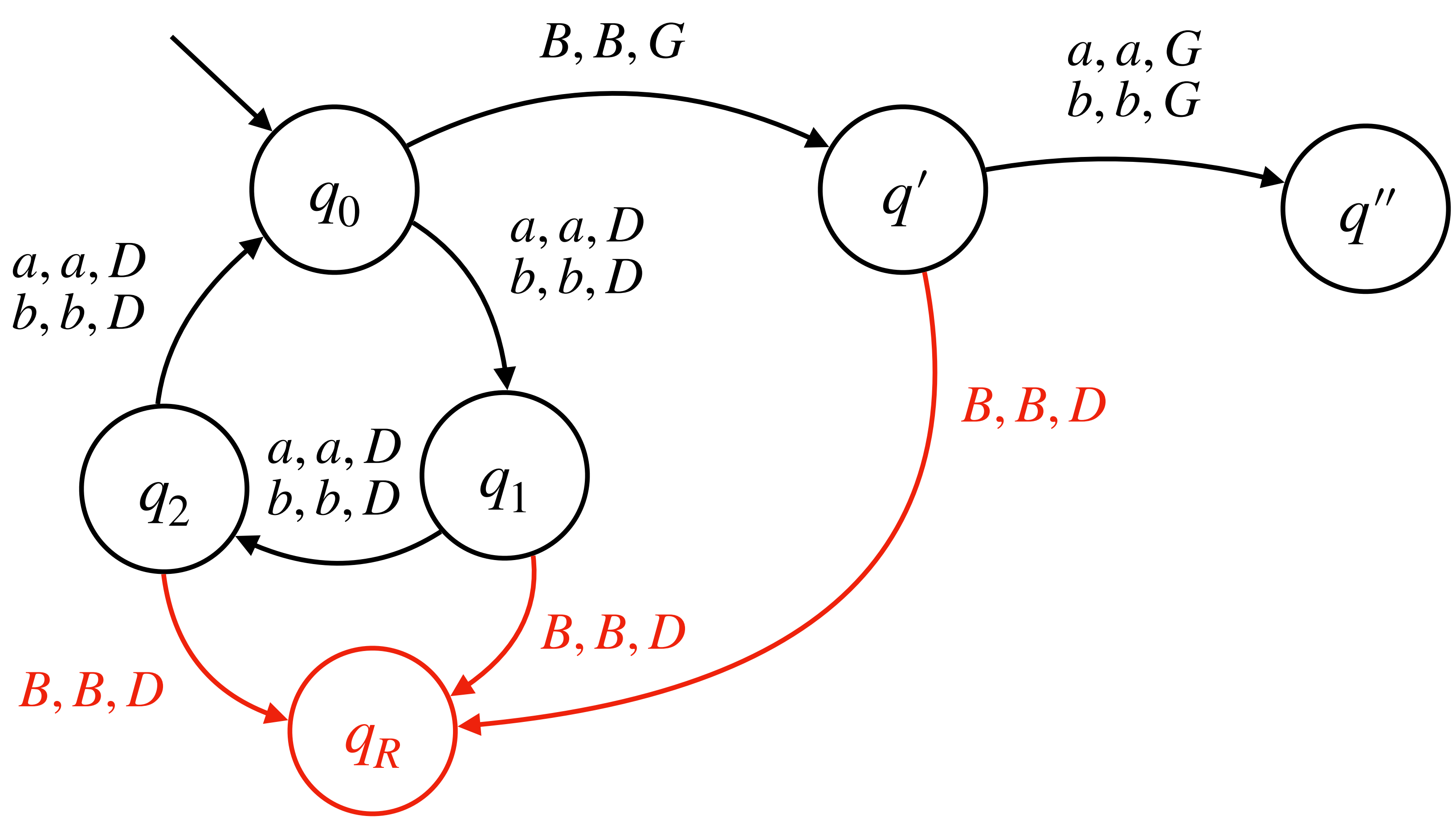
$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

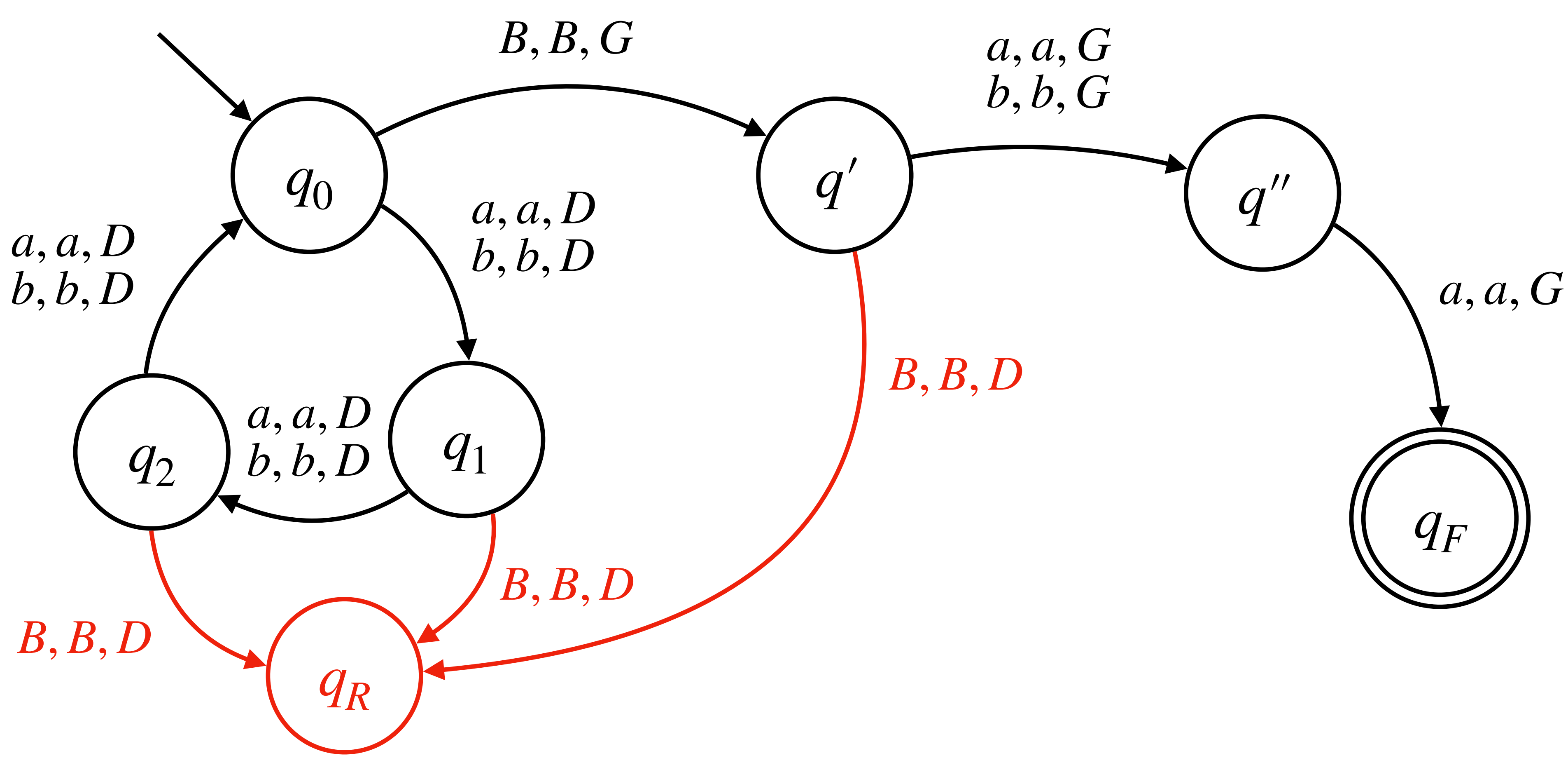
$$L_1 = \left\{w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a\right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

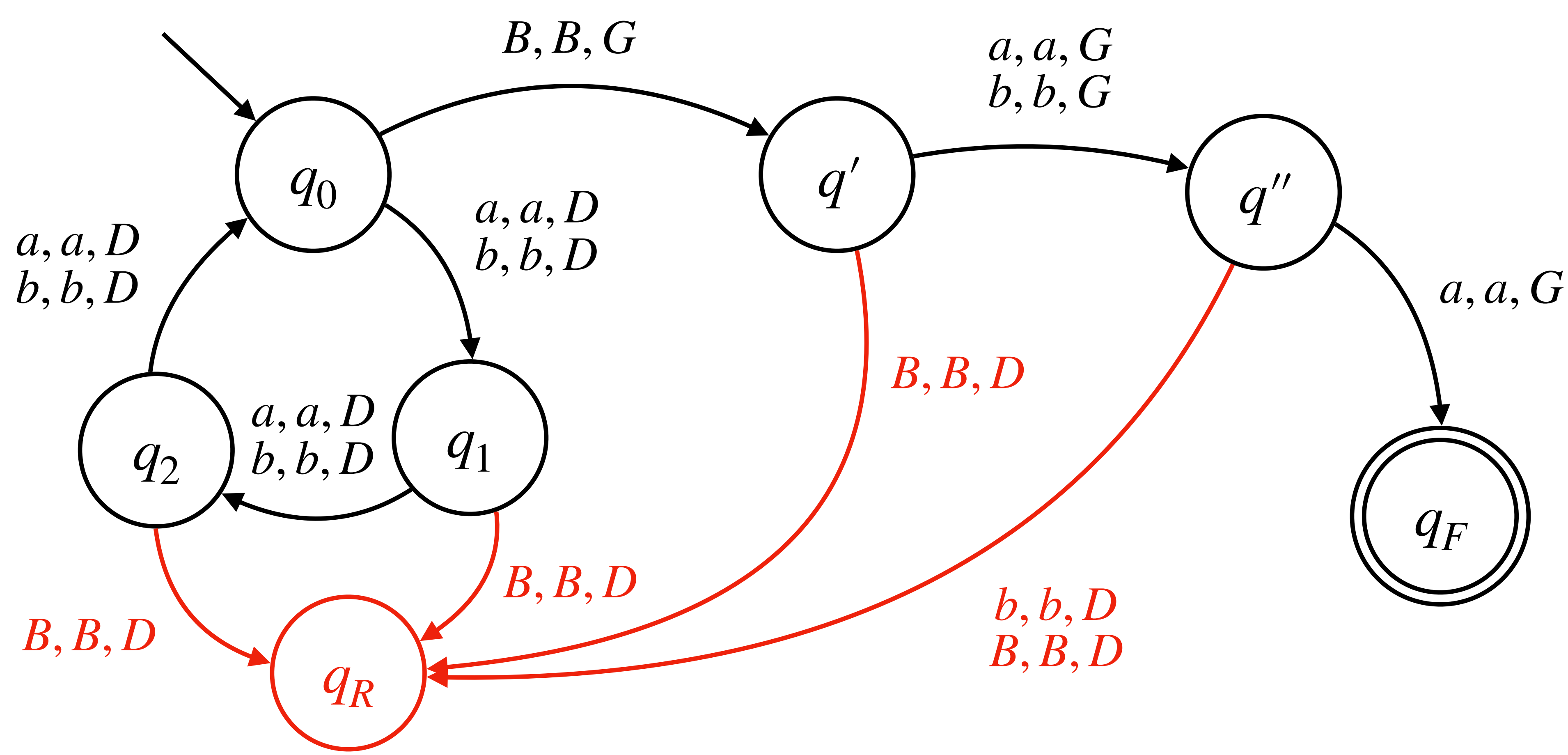
$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

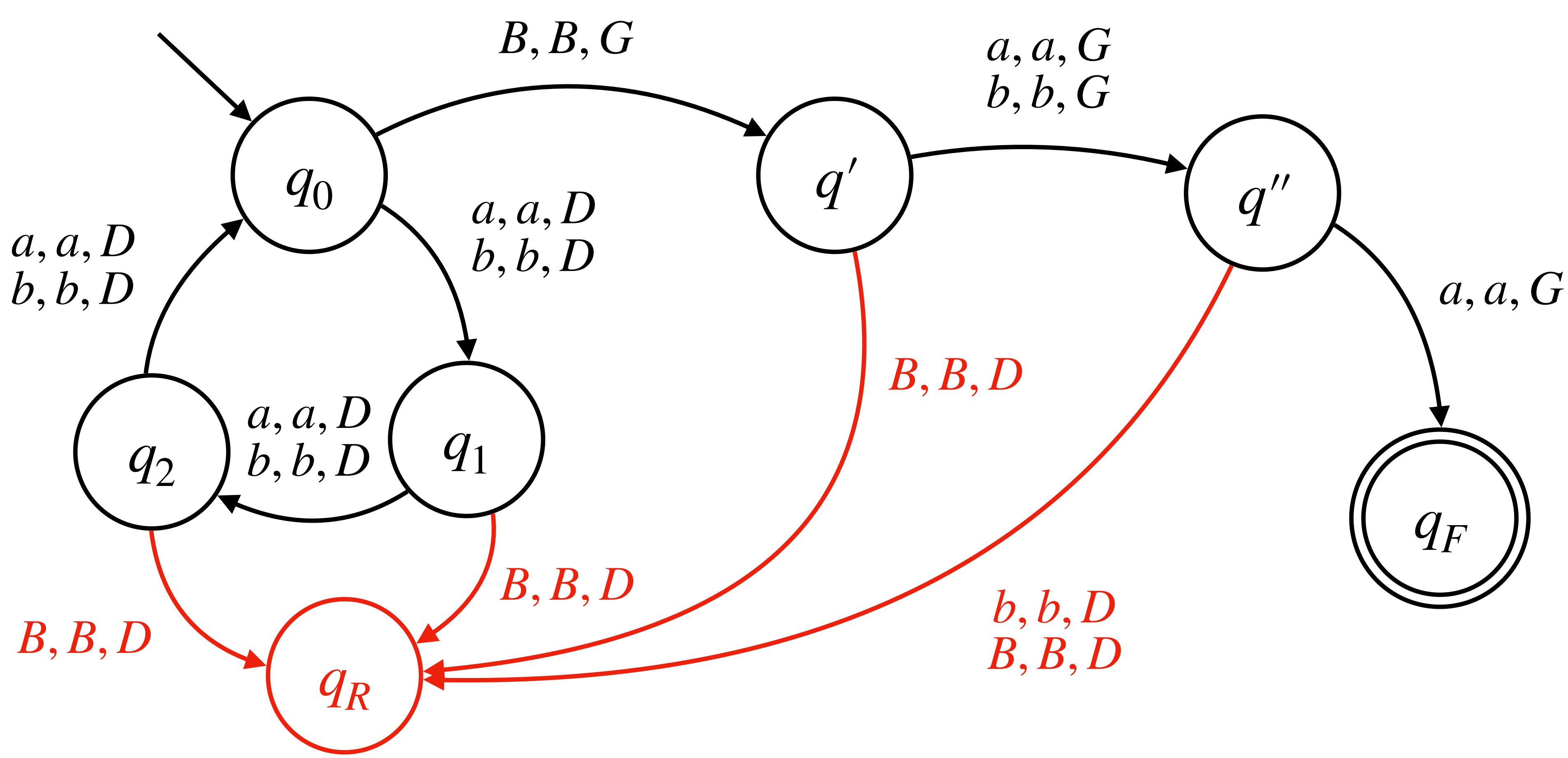
$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.1

Dessiner l'automate d'une machine de Turing qui reconnaît le langage suivant et qui s'arrête toujours :

$$L_1 = \left\{ w_1 w_2 \cdots w_n \in \{a, b\}^\star \mid n \geq 2 \text{ et } n \equiv 0 \bmod 3 \text{ et } w_{n-1} = a \right\}$$



Exercice 2.2

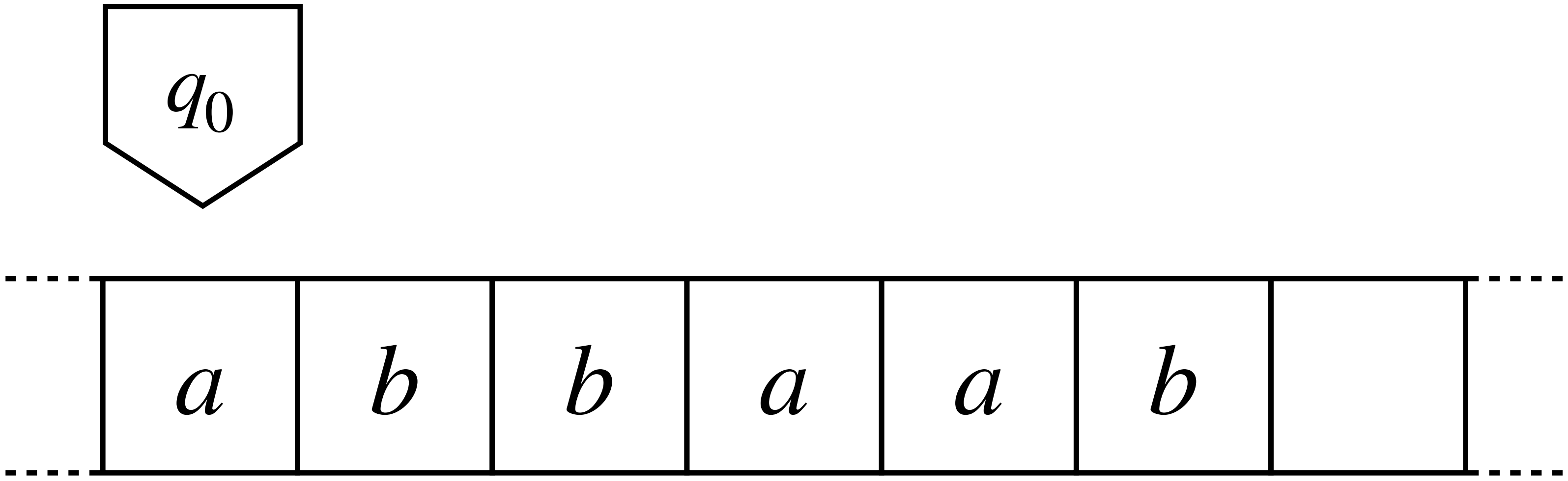
Exercice 2.2

Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$

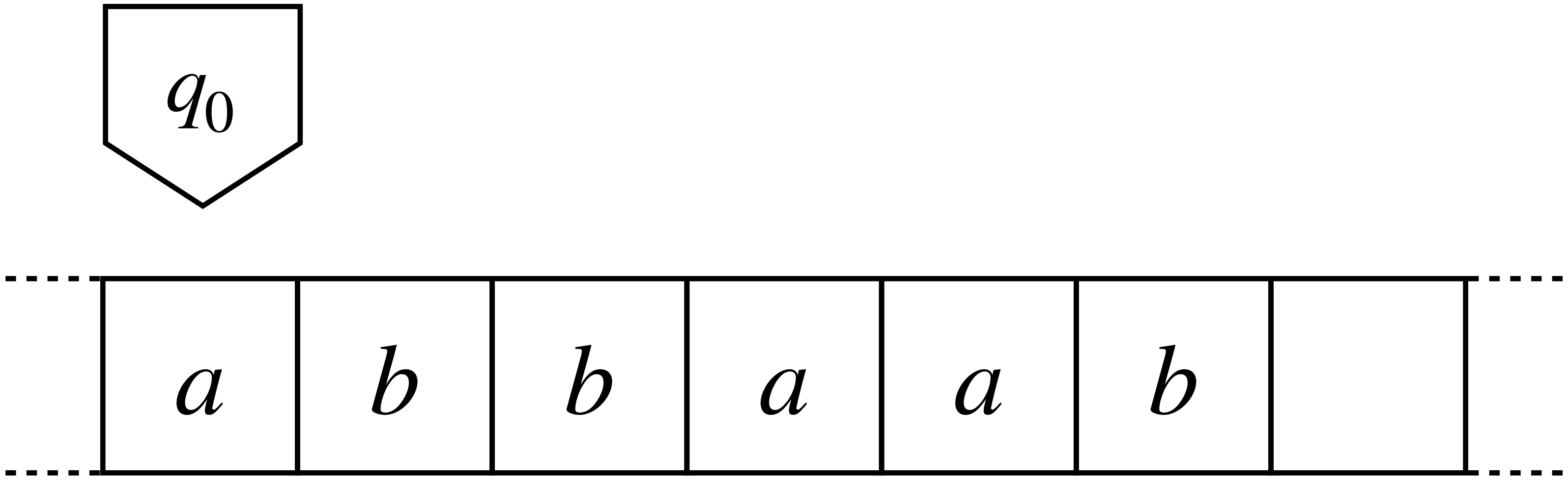
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



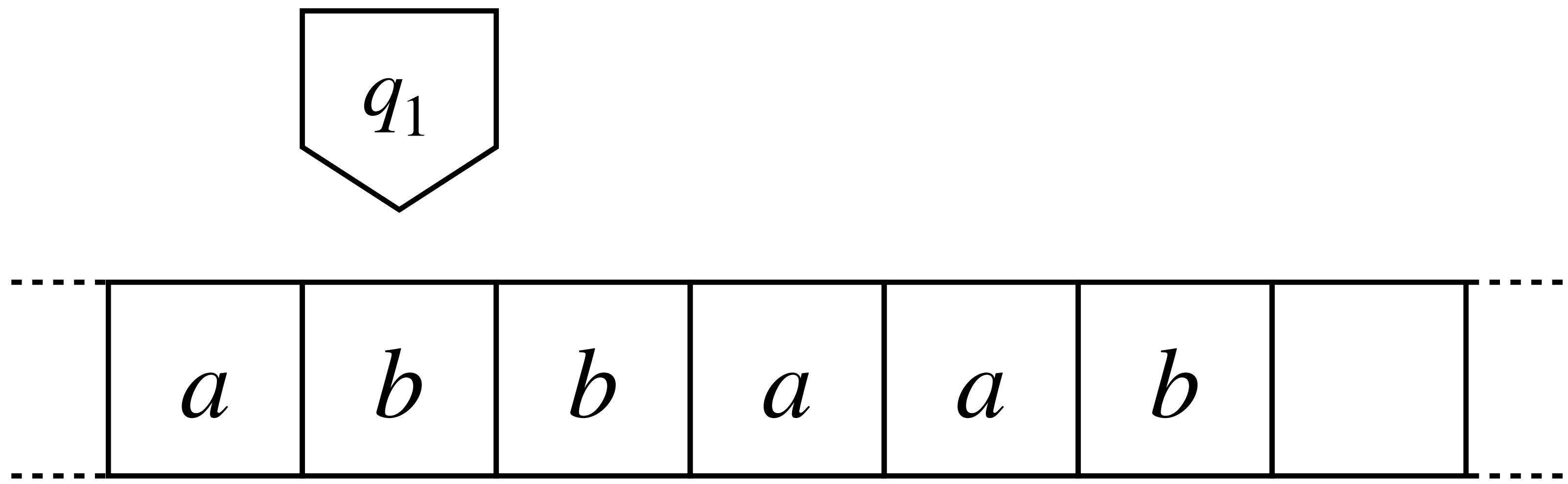
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



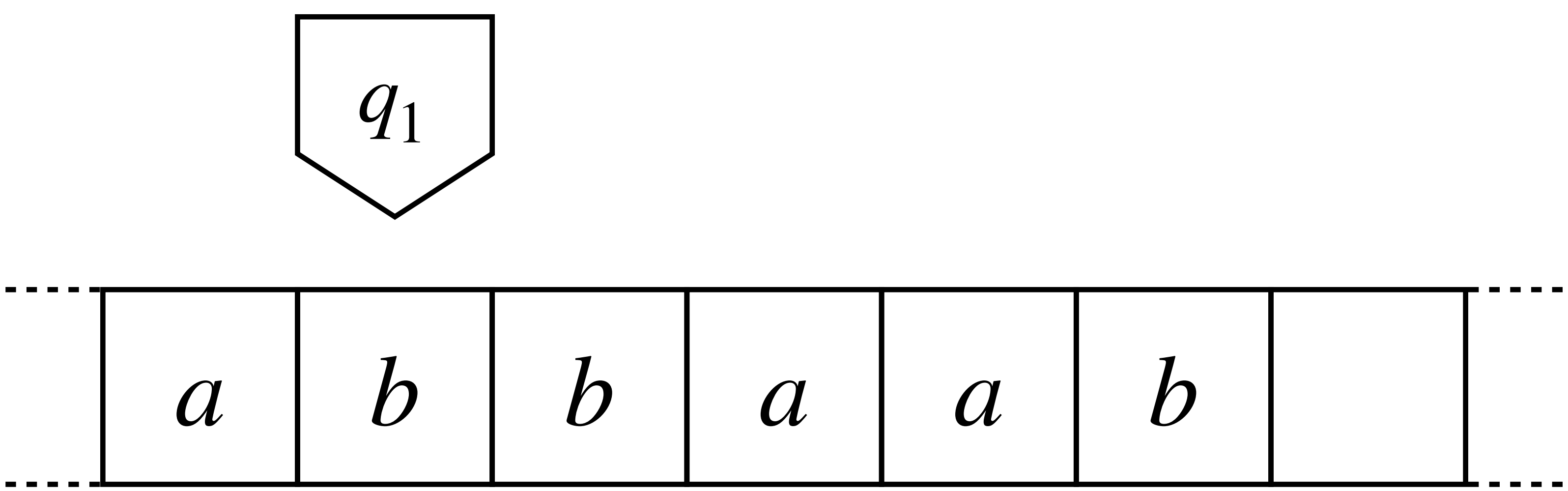
Exercice 2.2

Donner l’exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l’entrée $abbaab$



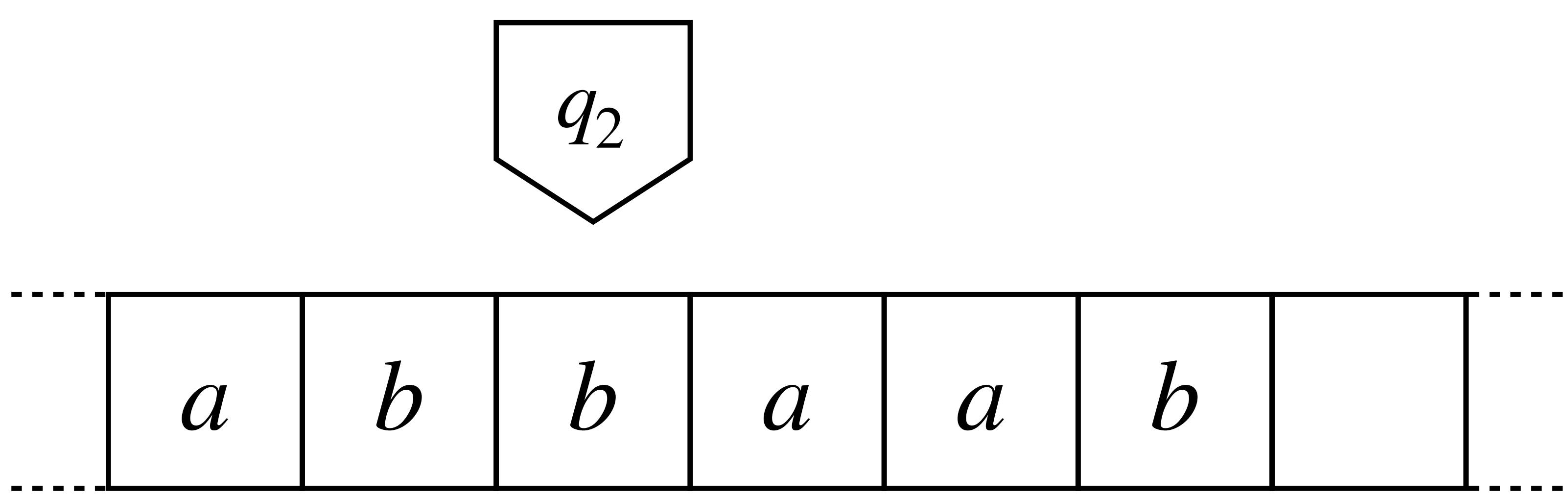
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



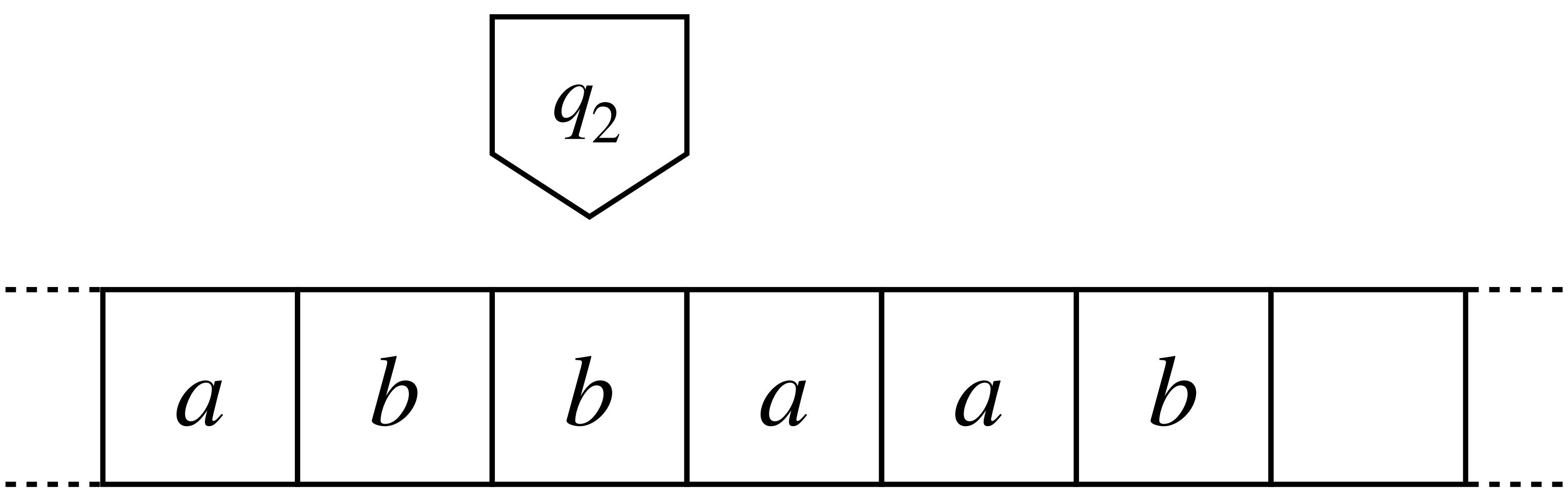
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



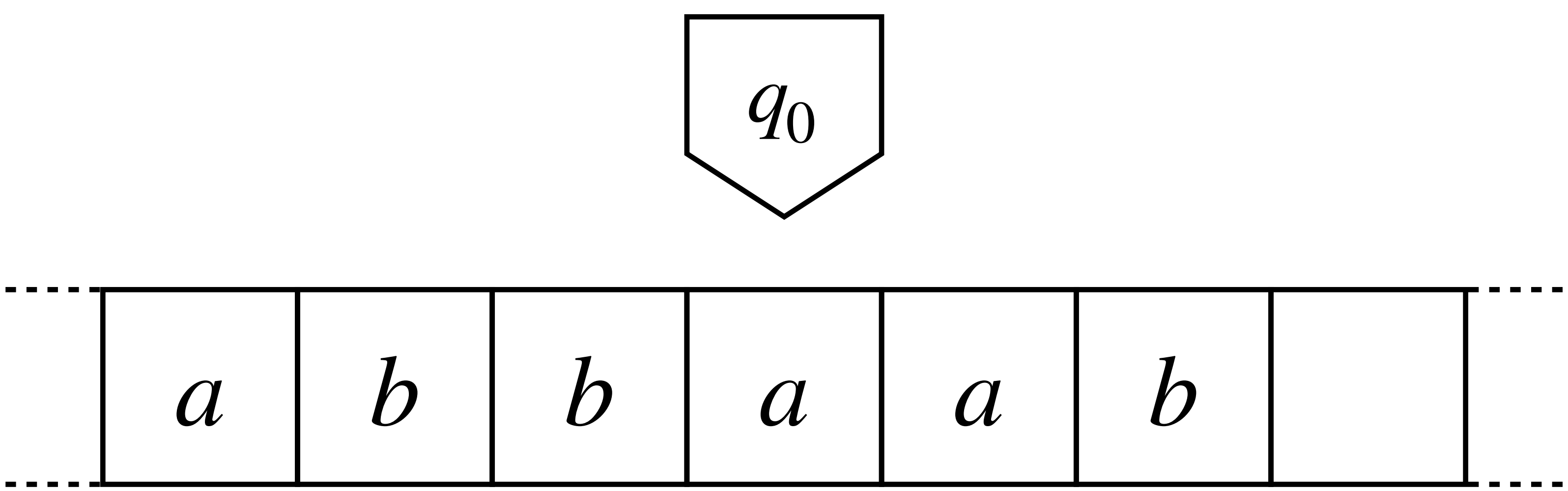
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



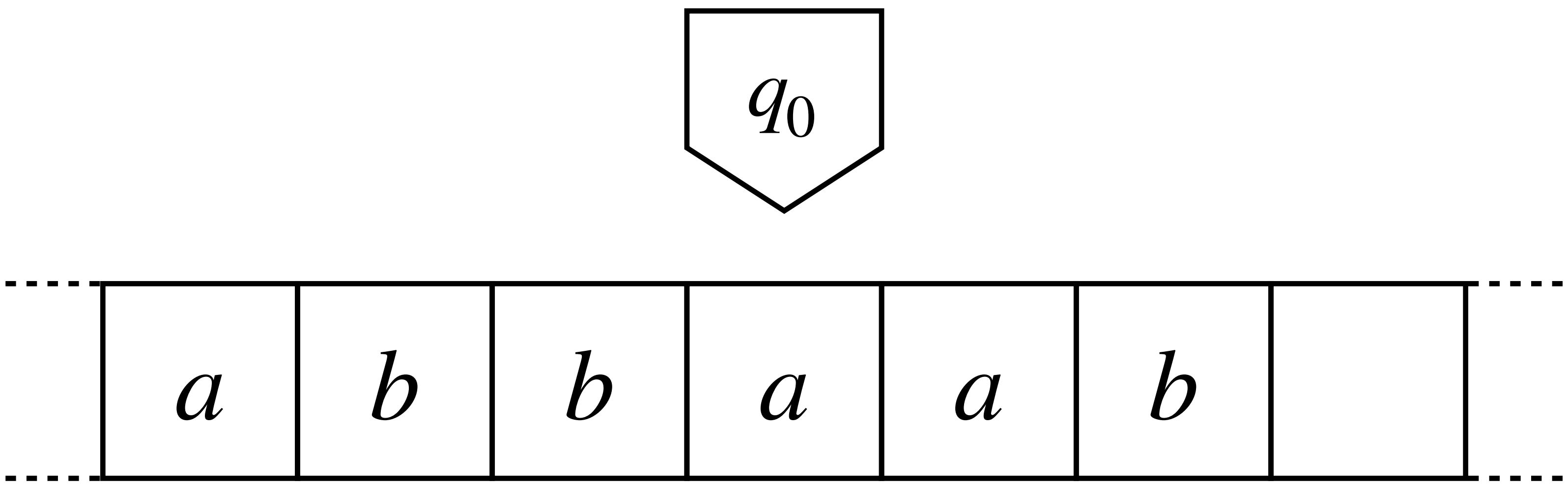
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



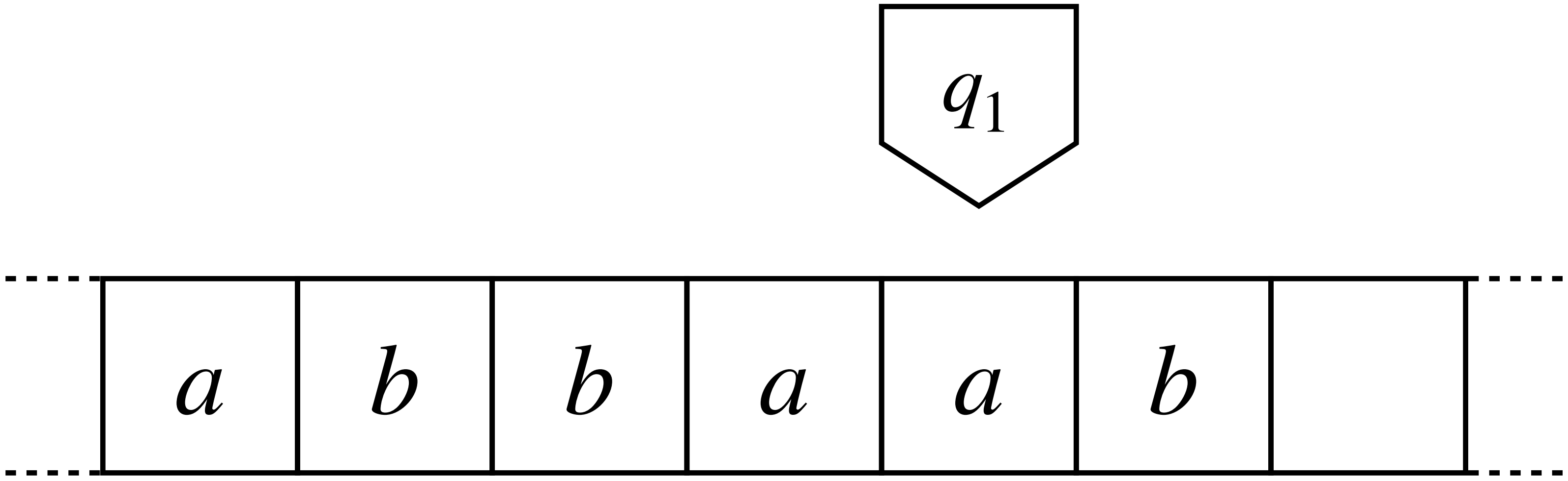
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



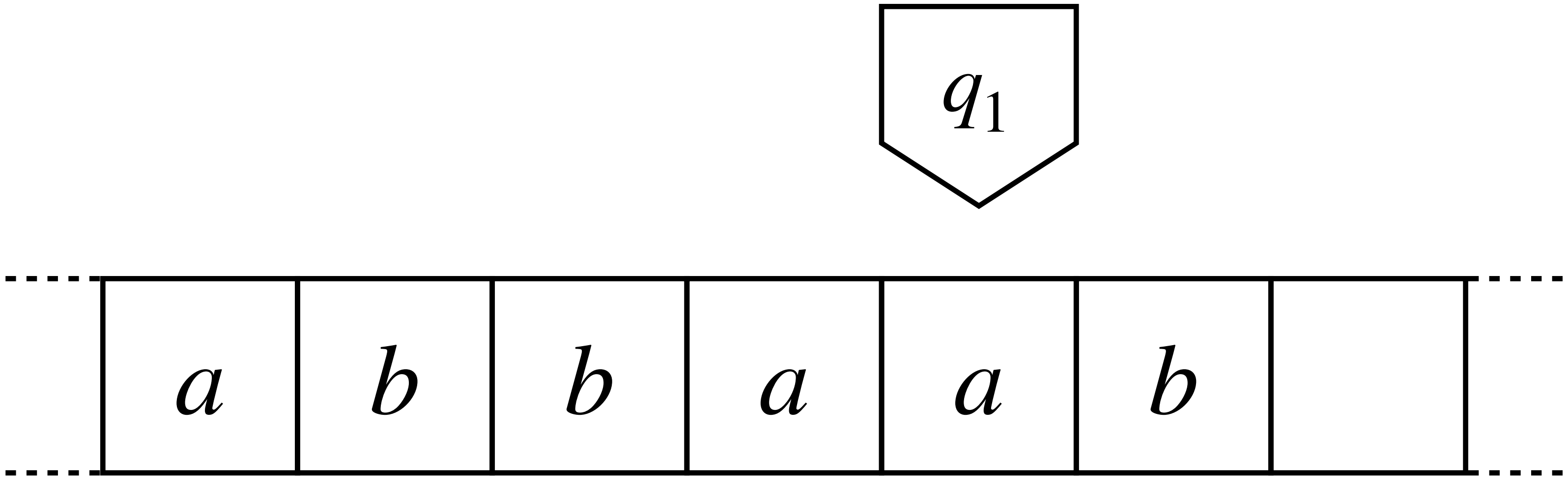
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



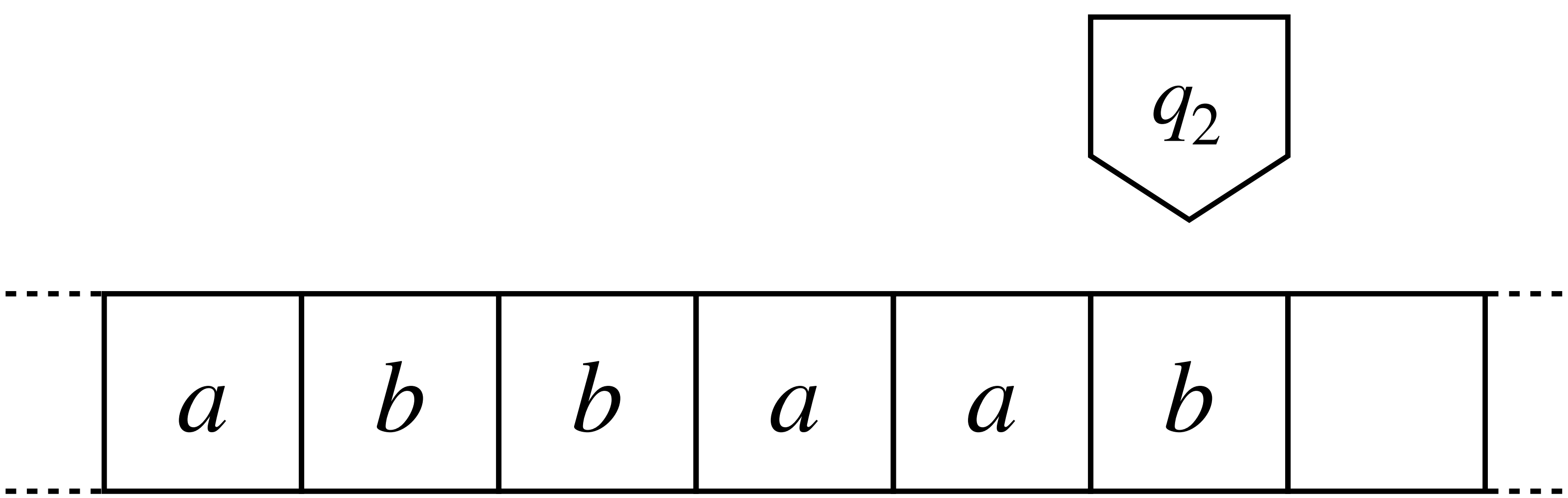
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



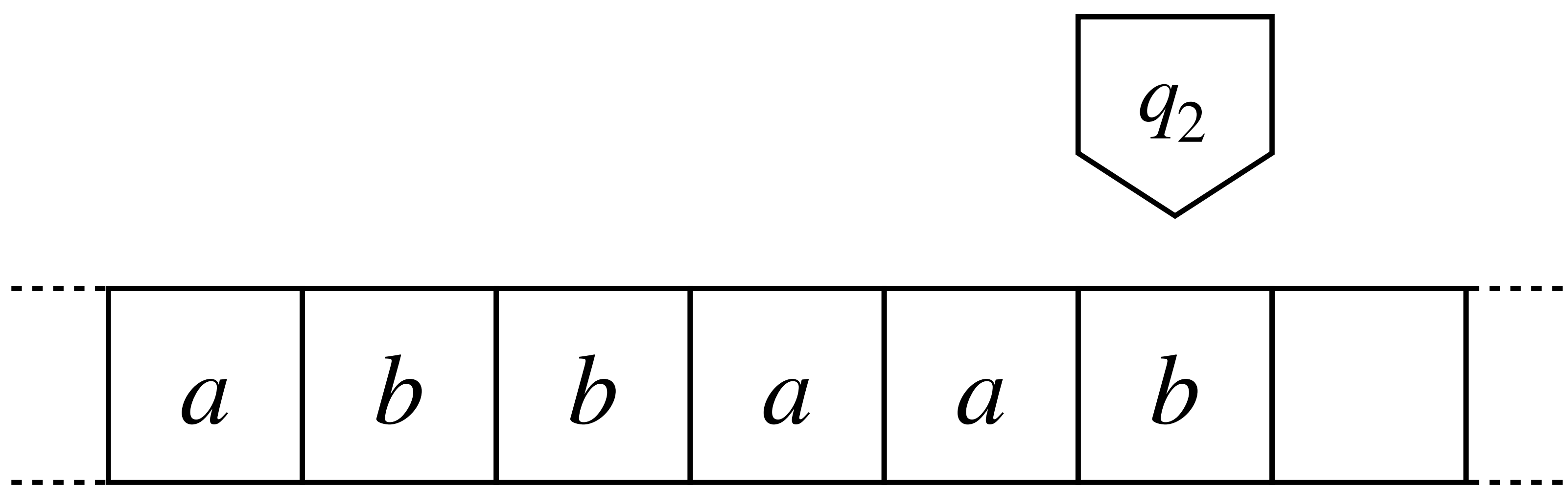
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



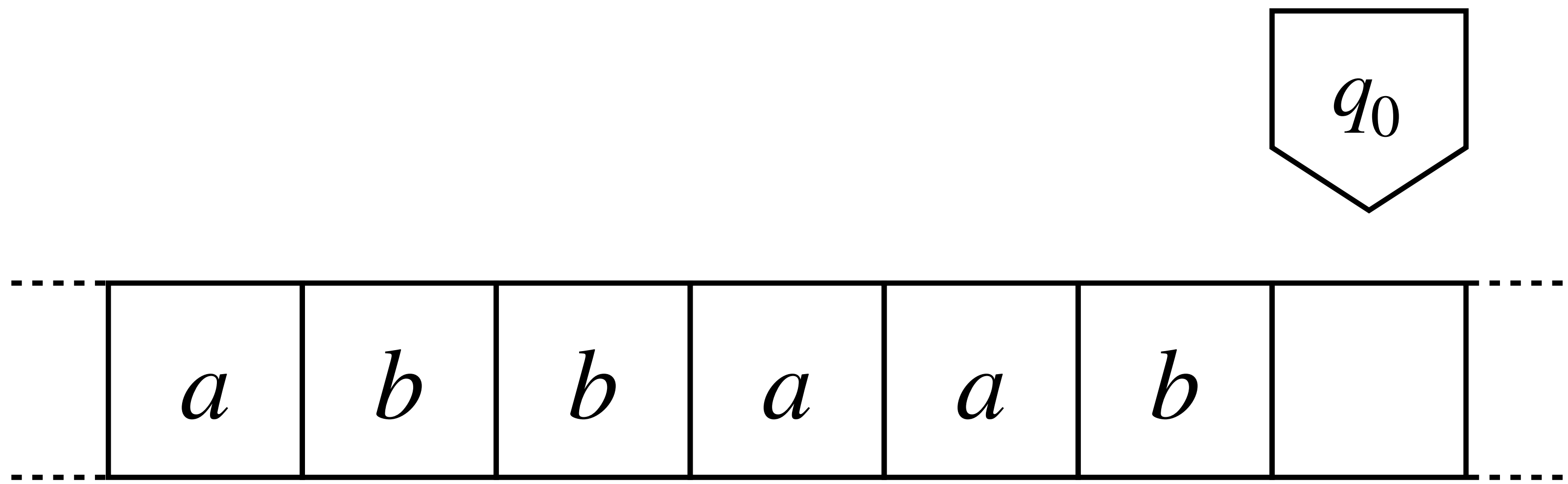
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



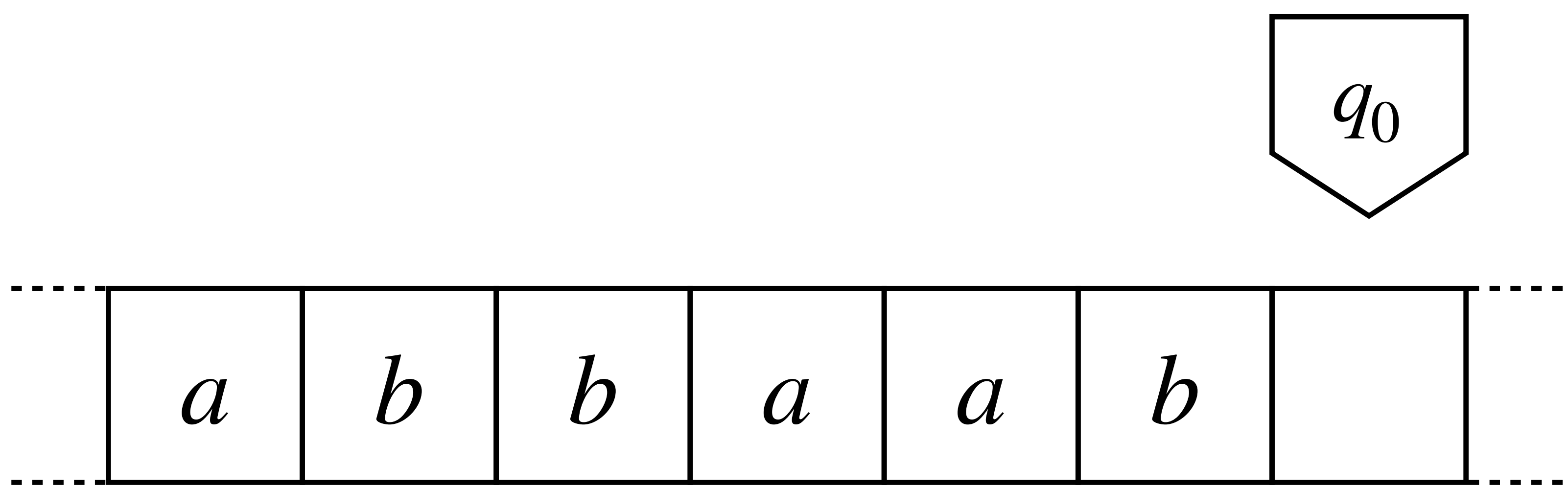
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



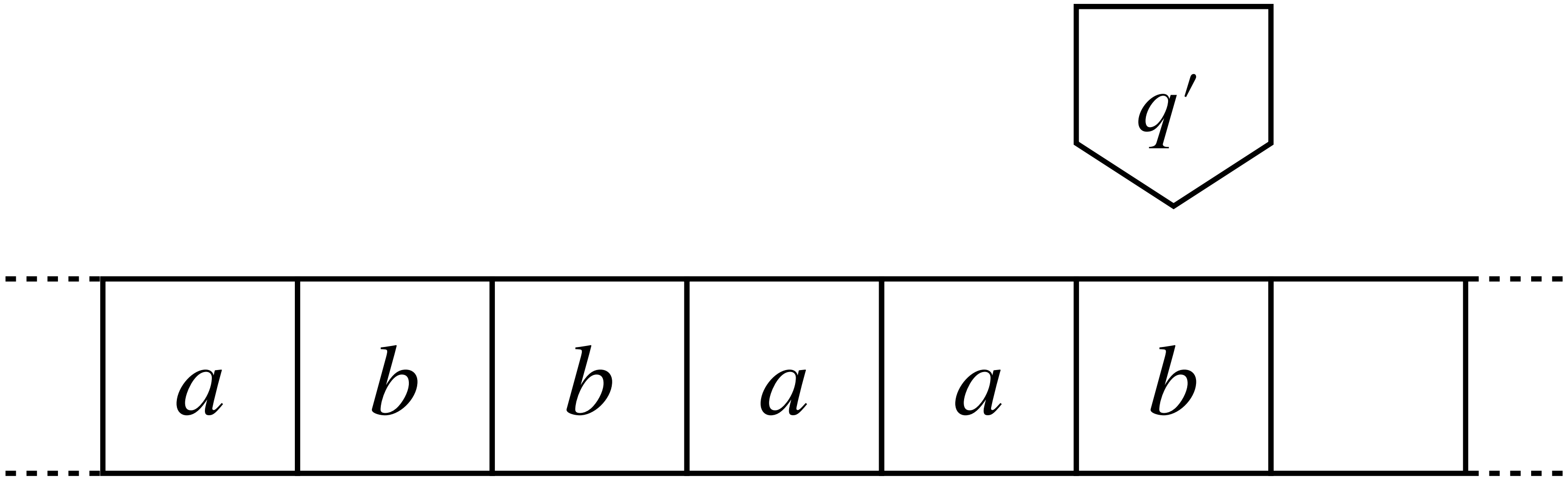
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



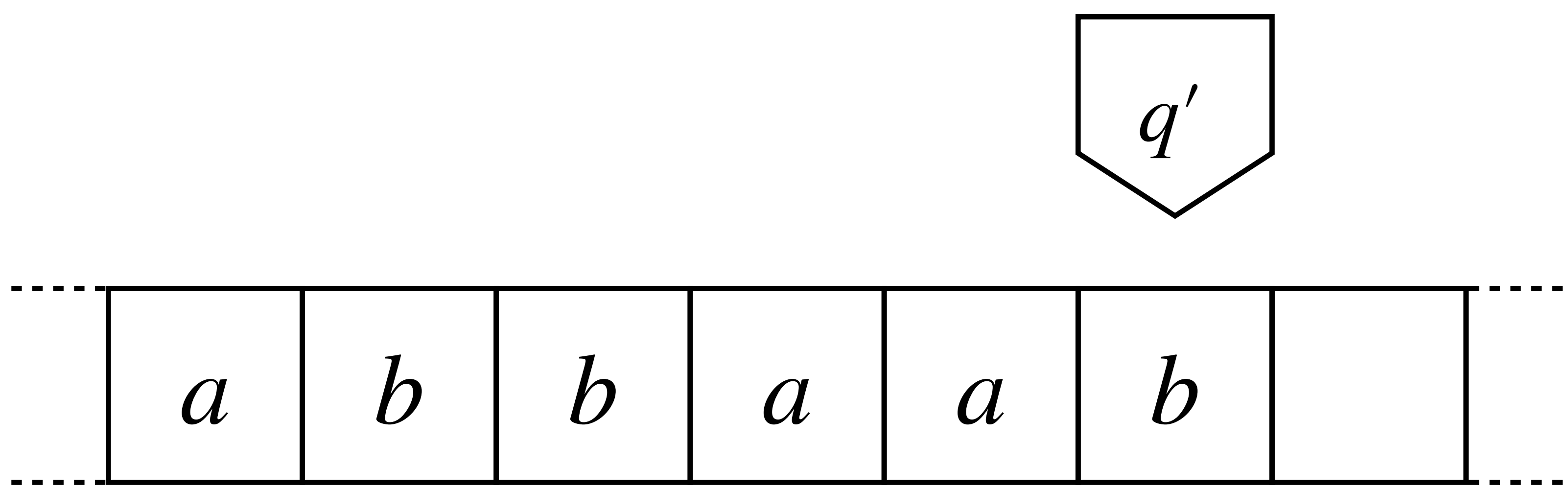
Exercice 2.2

Donner l’exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l’entrée $abbaab$



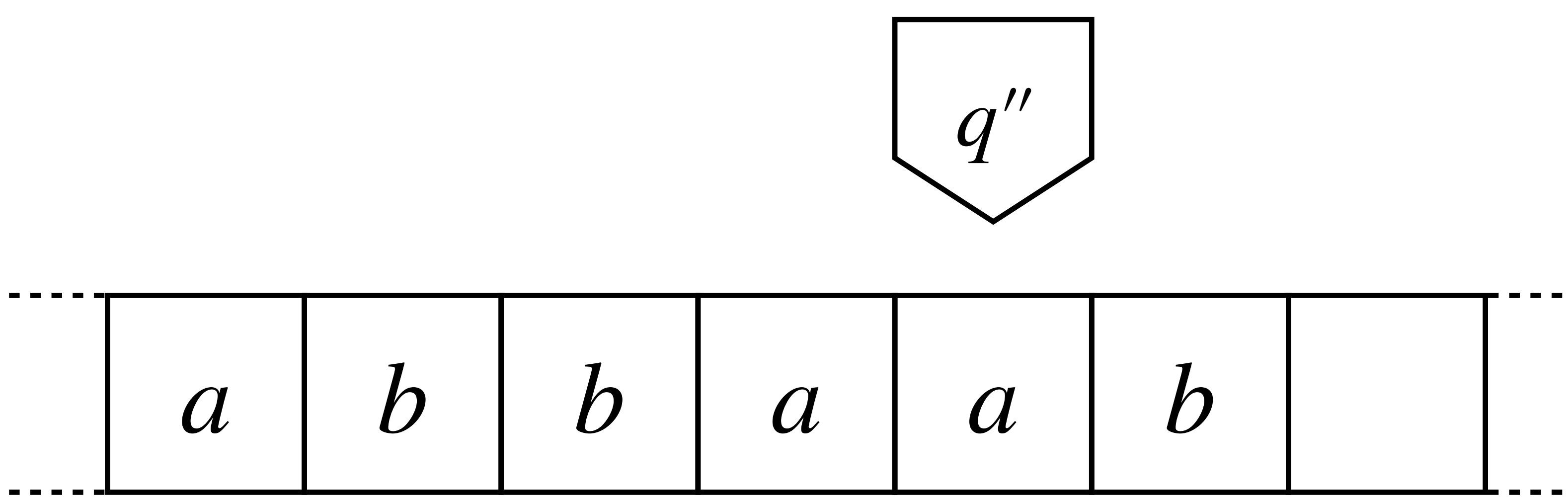
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



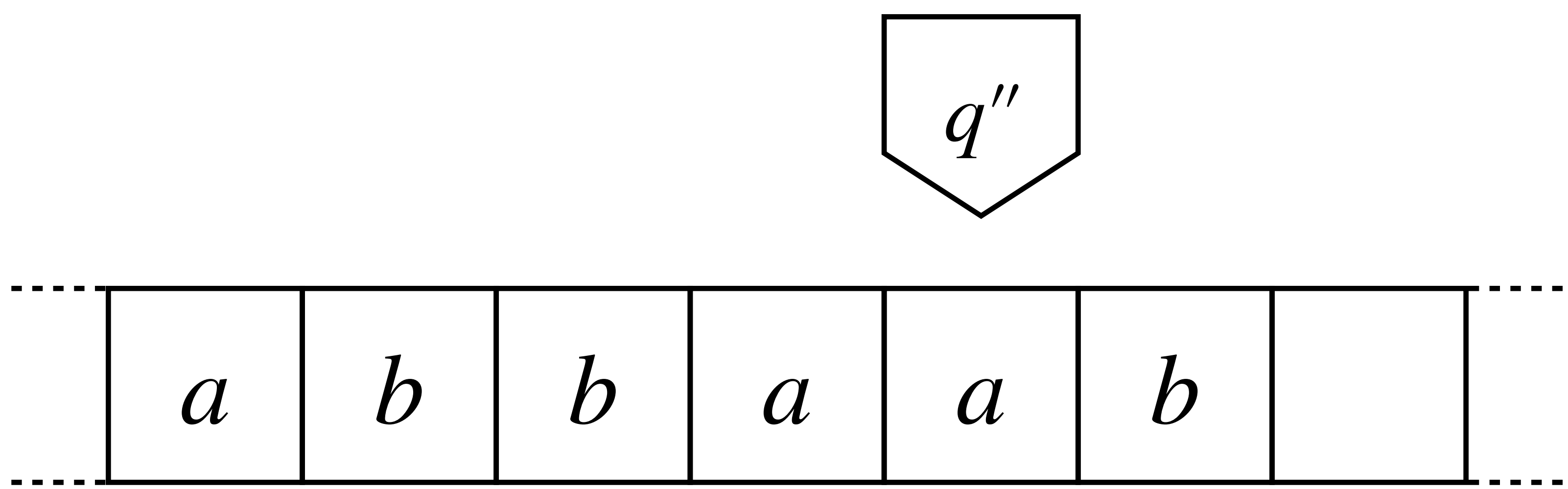
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



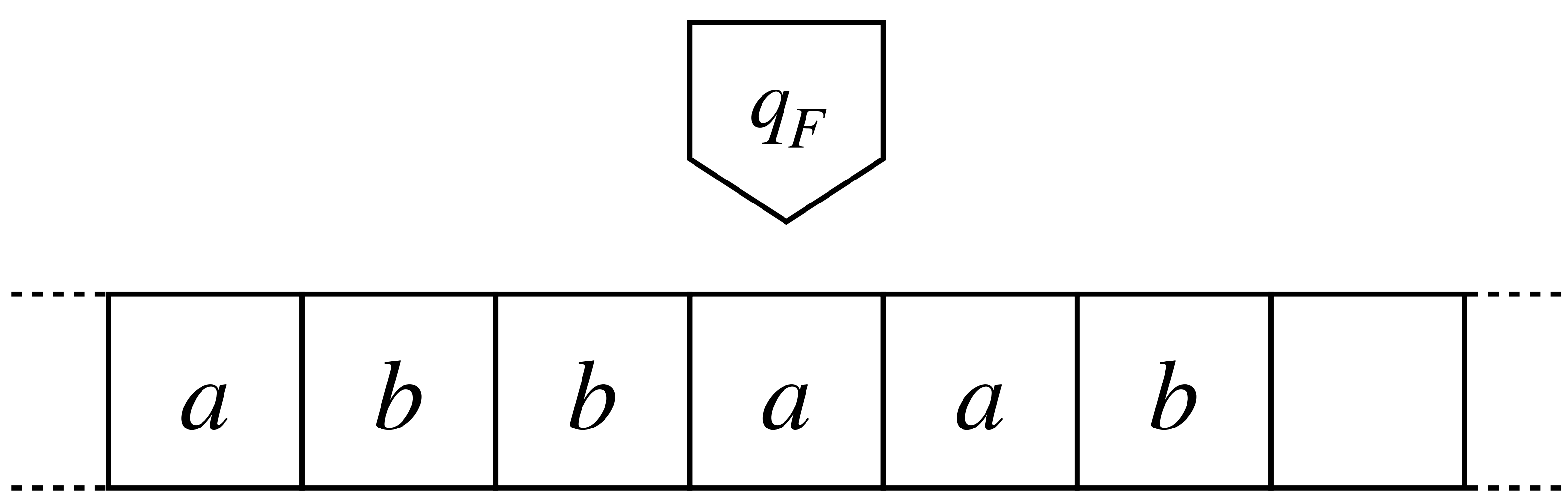
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



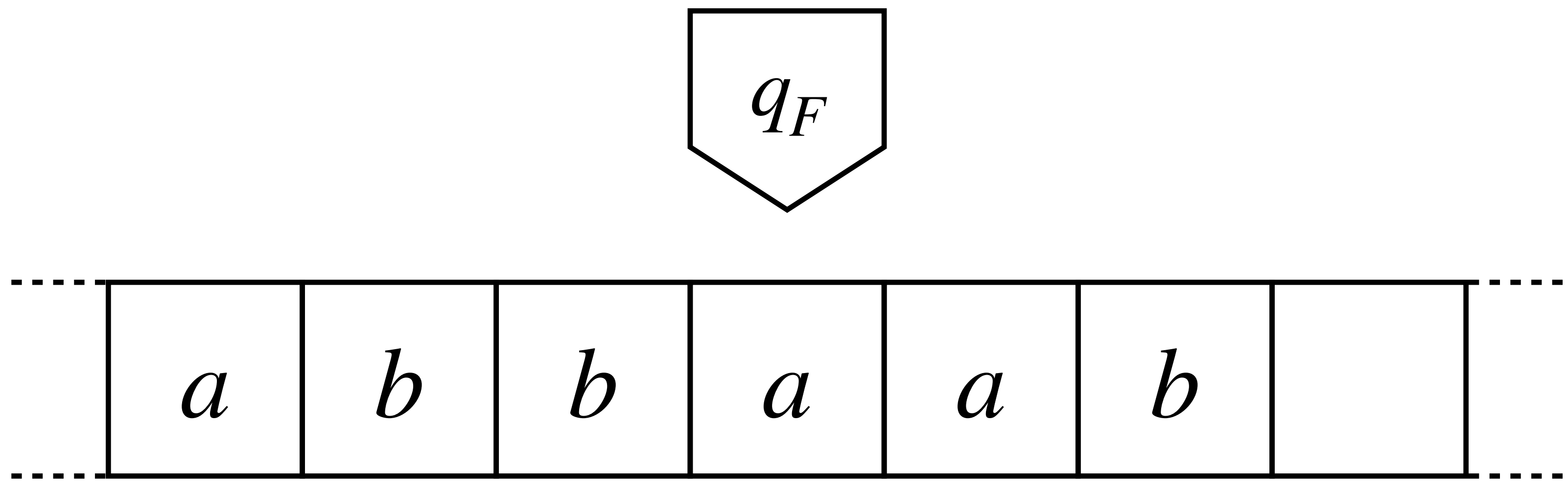
Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



Exercice 2.2

Donner l'exécution (la séquence des descriptions instantanées des configurations) de la machine que vous avez définie en question 1 sur l'entrée $abbaab$



Exercice 2.3

Exercice 2.3

Exercice 2.3

Peut-on déduire de la question 1 que L_1 est :
(a) récursif ? (b) récursivement énumérable ?

Exercice 2.3

Peut-on déduire de la question 1 que L_1 est :
(a) récursif ? (b) récursivement énumérable ?

- L_1 est reconnu par une machine qui s'arrête toujours, donc il est (a) récursif

Exercice 2.3

Peut-on déduire de la question 1 que L_1 est :
(a) récursif ? (b) récursivement énumérable ?

- L_1 est reconnu par une machine qui s'arrête toujours, donc il est (a) récursif
- Un langage récursif est toujours (b) récursivement énumérable (exercice 1.5), donc L_1 l'est aussi

Exercice 3

Réduction many-one Turing

Exercice 3

Réduction many-one Turing

Exercice 3.1

Exercice 3.1

Exercice 3.1

Montrer que $L_{\bar{u}} \leq_m^T L_2$, avec $L_{\bar{u}} = \{\langle M \rangle \# w \mid M$ n'accepte pas $w\}$ et $L_2 = \{\langle M \rangle \# w \mid M(w) \uparrow \}$

Exercice 3.1

Montrer que $L_{\bar{u}} \leq_m^T L_2$, avec $L_{\bar{u}} = \{\langle M \rangle \# w \mid M$ n'accepte pas $w\}$ et $L_2 = \{\langle M \rangle \# w \mid M(w) \uparrow \}$

- Soit $f\big(\langle M \rangle \# w\big) = \langle M' \rangle \# w$ avec M' définie par

  $M'(x) =$

  simuler M sur x ;

  si M accepte x **alors** accepter, **sinon** boucler à l'infini

Exercice 3.1

Montrer que $L_{\bar{u}} \leq_m^T L_2$, avec $L_{\bar{u}} = \{\langle M \rangle \# w \mid M$ n'accepte pas $w\}$ et $L_2 = \{\langle M \rangle \# w \mid M(w) \uparrow \}$

- Soit $f\big(\langle M \rangle \# w\big) = \langle M' \rangle \# w$ avec M' définie par

  $M'(x) =$

  simuler M sur x ;

  si M accepte x **alors** accepter, **sinon** boucler à l'infini

- f est calculable par une machine de Turing ; il suffit de modifier le codage $\langle M \rangle$ en $\langle M' \rangle$ et recopier w

Exercice 3.1

Montrer que $L_{\bar{u}} \leq_m^T L_2$, avec $L_{\bar{u}} = \{\langle M \rangle \# w \mid M$ n'accepte pas $w\}$ et $L_2 = \{\langle M \rangle \# w \mid M(w) \uparrow \}$

- Soit $f\big(\langle M \rangle \# w\big) = \langle M' \rangle \# w$ avec M' définie par

$M'(x) =$

simuler M sur x ;

si M accepte x **alors** accepter, **sinon** boucler à l'infini

- f est calculable par une machine de Turing ; il suffit de modifier le codage $\langle M \rangle$ en $\langle M' \rangle$ et recopier w

- Maintenant il faut montrer que $\langle M \rangle \# w \in L_{\bar{u}}$ ssi $f\big(\langle M \rangle \# w\big) = \langle M' \rangle \# w \in L_2$

Exercice 3.1

Exercice 3.1

- Si $\langle M \rangle \# w \in L_{\bar{u}}$ alors M n'accepte pas w, soit parce qu'elle ne termine pas, soit parce qu'elle s'arrête sans accepter

Exercice 3.1

- Si $\langle M \rangle \# w \in L_{\bar{u}}$ alors M n'accepte pas w, soit parce qu'elle ne termine pas, soit parce qu'elle s'arrête sans accepter

- Donc M' ne s'arrête pas sur l'entrée w :

$M'(w) =$

simuler M sur w ;

si M accepte w **alors** accepter, **sinon** boucler à l'infini

Exercice 3.1

- Si $\langle M \rangle \# w \in L_{\bar{u}}$ alors M n'accepte pas w, soit parce qu'elle ne termine pas, soit parce qu'elle s'arrête sans accepter

- Donc M' ne s'arrête pas sur l'entrée w :

  $M'(w) =$

  simuler M sur w ;

  si M accepte w **alors** accepter, **sinon** boucler à l'infini

- Soit la simulation de M sur w ne termine pas (donc M' non plus), soit elle termine en rejetant et M' entre dans une boucle infinie

Exercice 3.1

- Si $\langle M \rangle \# w \in L_{\bar{u}}$ alors M n'accepte pas w, soit parce qu'elle ne termine pas, soit parce qu'elle s'arrête sans accepter

- Donc M' ne s'arrête pas sur l'entrée w :

  $M'(w) =$

  simuler M sur w ;

  si M accepte w **alors** accepter, **sinon** boucler à l'infini

- Soit la simulation de M sur w ne termine pas (donc M' non plus), soit elle termine en rejetant et M' entre dans une boucle infinie

- Dans le deux cas on obtient $\langle M' \rangle \# w \in L_2$

Exercice 3.1

Exercice 3.1

- Si, au contraire, $\langle M \rangle \# w \notin L_{\bar{u}}$ alors M accepte w et en particulier elle s'arrête

Exercice 3.1

- Si, au contraire, $\langle M \rangle \# w \notin L_{\bar{u}}$ alors M accepte w et en particulier elle s'arrête

- Donc M' s'arrête aussi sur l'entrée w :

$M'(w) =$

simuler M sur w ;

si M accepte w **alors** accepter, **sinon** boucler à l'infini

Exercice 3.1

- Si, au contraire, $\langle M \rangle \# w \notin L_{\bar{u}}$ alors M accepte w et en particulier elle s'arrête

- Donc M' s'arrête aussi sur l'entrée w :

$M'(w) =$

simuler M sur w ;

si M accepte w **alors** accepter, **sinon** boucler à l'infini

- La simulation de M sur w termine et donc M' accepte

Exercice 3.1

- Si, au contraire, $\langle M \rangle \# w \notin L_{\bar{u}}$ alors M accepte w et en particulier elle s'arrête

- Donc M' s'arrête aussi sur l'entrée w :

  $M'(w) =$

  simuler M sur w ;

  si M accepte w **alors** accepter, **sinon** boucler à l'infini

- La simulation de M sur w termine et donc M' accepte

- Donc on obtient $\langle M' \rangle \# w \notin L_2$

Exercice 3.2

Exercice 3.2

Exercice 3.2

Pourquoi peut-on en déduire que L_2 n'est pas récursif ?

Exercice 3.2

Pourquoi peut-on en déduire que L_2 n'est pas récursif ?

- On vient de démontrer que $L_{\bar{u}} \leq_m^T L_2$

Exercice 3.2

Pourquoi peut-on en déduire que L_2 n'est pas récursif ?

- On vient de démontrer que $L_{\bar{u}} \leq_m^T L_2$
- Dans ce cas, si L_2 était récursif, alors $L_{\bar{u}}$ serait récursif aussi (exercice 1.4)

Exercice 3.2

Pourquoi peut-on en déduire que L_2 n'est pas récursif ?

- On vient de démontrer que $L_{\bar{u}} \leq_m^T L_2$
- Dans ce cas, si L_2 était récursif, alors $L_{\bar{u}}$ serait récursif aussi (exercice 1.4)
- Mais $L_{\bar{u}}$ n'est pas récursivement énumérable, donc pas récursif non plus (exercice 1.5, réponse (a))

Exercice 4
Théorème de Rice

Exercice 4
Théorème de Rice

Exercice 4.1

Exercice 4.1

Exercice 4.1

Qu’est-ce qu’une propriété *non triviale* ?

Exercice 4.1

Qu'est-ce qu'une propriété *non triviale* ?

- C'est une famille (ensemble) de langages $P \subseteq \mathscr{P}(\Sigma^\star)$ telle que

Exercice 4.1

Qu’est-ce qu’une propriété *non triviale* ?

- C’est une famille (ensemble) de langages $P \subseteq \mathscr{P}(\Sigma^\star)$ telle que
    - il existe une machine M_1 telle que $L(M_1) \in P$

Exercice 4.1

Qu’est-ce qu’une propriété *non triviale* ?

- C’est une famille (ensemble) de langages $P \subseteq \mathscr{P}(\Sigma^\star)$ telle que
  - il existe une machine M_1 telle que $L(M_1) \in P$
  - il existe une machine M_2 telle que $L(M_2) \notin P$

Exercice 4.2

Exercice 4.2

Exercice 4.2

Donner un exemple de propriété *triviale*

Exercice 4.2

Donner un exemple de propriété *triviale*

- L'ensemble vide $P = \varnothing$ est une propriété triviale, parce que pour chaque machine M on a $L(M) \notin P$

Exercice 4.2

Donner un exemple de propriété *triviale*

- L'ensemble vide $P = \varnothing$ est une propriété triviale, parce que pour chaque machine M on a $L(M) \notin P$

- La famille de tous les langages $P = \mathscr{P}(\Sigma^\star)$ est une propriété triviale, parce que pour chaque machine M on a $L(M) \in P$

Exercice 4.2

Donner un exemple de propriété *triviale*

- L'ensemble vide $P = \varnothing$ est une propriété triviale, parce que pour chaque machine M on a $L(M) \notin P$

- La famille de tous les langages $P = \mathscr{P}(\Sigma^\star)$ est une propriété triviale, parce que pour chaque machine M on a $L(M) \in P$

- La famille de langages $P = \{L_{\bar{u}}\}$ est aussi une propriété triviale, parce que $L_{\bar{u}} \neq \mathbf{RE}$, donc pour aucune machine M on a $L(M) \in P$

Exercice 4.3

Exercice 4.3

Exercice 4.3

Cette propriété (celle de votre réponse à la question 2) est-elle intéressante ?

Exercice 4.3

Cette propriété (celle de votre réponse à la question 2) est-elle intéressante ?

- Les propriétés $P = \varnothing$ et $P = \mathscr{P}(\Sigma^\star)$ sont triviales pour des raisons banales

Exercice 4.3

Cette propriété (celle de votre réponse à la question 2) est-elle intéressante ?

- Les propriétés $P = \varnothing$ et $P = \mathscr{P}(\Sigma^\star)$ sont triviales pour des raisons banales
- En revanche, $P = \{L_{\bar{u}}\}$ est triviale mais peut-être pour des raisons un peu plus intéressantes…

Exercice 4.4

Exercice 4.4

Exercice 4.4

Donner un exemple de propriété *non triviale*

Exercice 4.4

Donner un exemple de propriété *non triviale*

- $P = \{L \mid \text{tous les mots de } L \text{ ont longueur paire}\}$

Exercice 4.4

Donner un exemple de propriété *non triviale*

- $P = \{L \mid \text{tous les mots de } L \text{ ont longueur paire}\}$
- Il existe une machine de Turing M_1 qui accepte exactement tous les mots de longueur paire, donc $L(M_1) \in P$

Exercice 4.4

Donner un exemple de propriété *non triviale*

- $P = \{L \mid \text{tous les mots de } L \text{ ont longueur paire}\}$
- Il existe une machine de Turing M_1 qui accepte exactement tous les mots de longueur paire, donc $L(M_1) \in P$
- Il existe aussi une machine de Turing M_2 qui accepte exactement tous les mots de longueur impaire, donc $L(M_2) \notin P$

Exercice 4.4

Donner un exemple de propriété *non triviale*

- $P = \{L \mid \text{tous les mots de } L \text{ ont longueur paire}\}$
- Il existe une machine de Turing M_1 qui accepte exactement tous les mots de longueur paire, donc $L(M_1) \in P$
- Il existe aussi une machine de Turing M_2 qui accepte exactement tous les mots de longueur impaire, donc $L(M_2) \notin P$
- En général, il suffit de trouver une famille de langages qui contient au moins un langage $\mathbf{RE}$ mais pas la totalité de $\mathbf{RE}$

Exercice 4.5

Exercice 4.5

Exercice 4.5

Que dit le théorème de Rice de cette propriété (celle de votre réponse à la question 4) ? Répondre en complétant la phrase suivante : Il n’existe pas de machine de Turing qui prenne en entrée…

Exercice 4.5

Que dit le théorème de Rice de cette propriété (celle de votre réponse à la question 4) ? Répondre en complétant la phrase suivante : Il n’existe pas de machine de Turing qui prenne en entrée…

- …le code $\langle M \rangle$ d’une machine de Turing M et décide, en s’arrêtant toujours, si M n’accepte que des mots de longueur paire (autrement dit, si le langage de M appartient à P)

Exercice 4.5

Que dit le théorème de Rice de cette propriété (celle de votre réponse à la question 4) ? Répondre en complétant la phrase suivante : Il n'existe pas de machine de Turing qui prenne en entrée…

- …le code $\langle M \rangle$ d'une machine de Turing M et décide, en s'arrêtant toujours, si M n'accepte que des mots de longueur paire (autrement dit, si le langage de M appartient à P)

- En général, le langage $L_P = \{\langle M \rangle \mid L(M) \in P\}$ n'est pas récursif pour toute propriété P non triviale

Exercice 5
Bonus

Exercice 5
Bonus

Exercice 5.1

Exercice 5.1

Exercice 5.1

Montrer que $L_2 \leq_m^T L_\infty$, avec $L_2 = \{\langle M \rangle \# w \mid M(w) \uparrow \}$ et $L_\infty = \{\langle M \rangle \mid M(w) \uparrow \text{ pour tout } w \in \Sigma^\star\}$

Exercice 5.1

Montrer que $L_2 \leq_m^T L_\infty$, avec $L_2 = \{\langle M \rangle \# w \mid M(w) \uparrow\ \}$ et $L_\infty = \{\langle M \rangle \mid M(w) \uparrow\ \text{pour tout } w \in \Sigma^\star\}$

- L_2 parle de machines M qui ne s'arrêtent pas sur un certain mot w, alors que L_∞ parle de machines qui ne s'arrêtent jamais

Exercice 5.1

Montrer que $L_2 \leq_m^T L_\infty$, avec $L_2 = \{\langle M \rangle \# w \mid M(w) \uparrow\ \}$ et $L_\infty = \{\langle M \rangle \mid M(w) \uparrow$ pour tout $w \in \Sigma^\star\}$

- L_2 parle de machines M qui ne s'arrêtent pas sur un certain mot w, alors que L_∞ parle de machines qui ne s'arrêtent jamais

- L'astuce pour cette réduction est de transformer la machine M en une machine M' qui ne s'arrête jamais si M ne s'arrête pas sur w ; sinon M' doit s'arrêter sur au moins un mot… c'est simple de choisir w lui-même !

Exercice 5.1

Exercice 5.1

- On fait la réduction (fonction calculable) $f\big(\langle M \rangle \# w\big) = \langle M' \rangle$ avec M' définie par

$M'(x) =$ **si** $x = w$ **alors** simuler M sur x **sinon** boucler à l'infini

Exercice 5.1

- On fait la réduction (fonction calculable) $f\big(\langle M \rangle \# w\big) = \langle M' \rangle$ avec M' définie par

  $M'(x) =$ **si** $x = w$ **alors** simuler M sur x **sinon** boucler à l'infini

- La machine M' ne s'arrête jamais, sauf éventuellement sur w

Exercice 5.1

- On fait la réduction (fonction calculable) $f\big(\langle M\rangle \# w\big) = \langle M'\rangle$ avec M' définie par

$M'(x) =$ **si** $x = w$ **alors** simuler M sur x **sinon** boucler à l'infini

- La machine M' ne s'arrête jamais, sauf éventuellement sur w

- Si $\langle M\rangle \# w \in L_2$ alors M ne s'arrête pas sur w, donc M' ne s'arrête sur aucun mot d'entrée, donc $\langle M'\rangle \in L_\infty$

Exercice 5.1

- On fait la réduction (fonction calculable) $f\big(\langle M\rangle\#w\big) = \langle M'\rangle$ avec M' définie par

$$M'(x) = \textbf{si } x = w \textbf{ alors } \text{simuler } M \text{ sur } x \textbf{ sinon } \text{boucler à l'infini}$$

- La machine M' ne s'arrête jamais, sauf éventuellement sur w
- Si $\langle M\rangle\#w \in L_2$ alors M ne s'arrête pas sur w, donc M' ne s'arrête sur aucun mot d'entrée, donc $\langle M'\rangle \in L_\infty$
- Si $\langle M\rangle\#w \notin L_2$ alors M s'arrête sur w, donc M' s'arrête sur exactement un mot (notamment w), donc $\langle M'\rangle \notin L_\infty$

Exercice 5.2

Exercice 5.2

Exercice 5.2

Que dire de $L_\infty \leq_m^T L_2$?

Exercice 5.2

Que dire de $L_\infty \leq_m^T L_2$?

- Peut-on faire aussi une réduction dans l'autre direction ?

Exercice 5.2

Que dire de $L_\infty \leq_m^T L_2$?

- Peut-on faire aussi une réduction dans l'autre direction ?

- Il faut transformer une entrée $\langle M \rangle$ de L_∞ en une entrée $\langle M' \rangle \# w$ tel que M' ne s'arrête pas sur le mot w ssi M ne s'arrête sur aucune entrée

Exercice 5.2

Que dire de $L_\infty \leq_m^T L_2$?

- Peut-on faire aussi une réduction dans l'autre direction ?
- Il faut transformer une entrée $\langle M \rangle$ de L_∞ en une entrée $\langle M' \rangle \# w$ tel que M' ne s'arrête pas sur le mot w ssi M ne s'arrête sur aucune entrée
- Ça nous demande de simuler M sur toutes les entrées possibles jusqu'à en trouver une sur laquelle M s'arrête (ou continuer a simuler a l'infini si ça n'arrive jamais)

Exercice 5.2

Exercice 5.2

- Soit $w_1, w_2, \ldots$ une énumération de tous les mots de $\Sigma^\star$

Exercice 5.2

- Soit $w_1, w_2, \ldots$ une énumération de tous les mots de $\Sigma^\star$
- Par exemple, d'abord le mot de longueur 0, puis les mots de longueur 1 en ordre lexicographique, puis les mots de longueur 2, etc…

Exercice 5.2

- Soit $w_1, w_2, \ldots$ une énumération de tous les mots de $\Sigma^\star$
- Par exemple, d'abord le mot de longueur 0, puis les mots de longueur 1 en ordre lexicographique, puis les mots de longueur 2, etc…
- Sur l'alphabet $\Sigma = \{a, b\}$ ça donne l'énumération

$$\begin{aligned} &\epsilon, \quad a, b, \quad aa, ab, ba, bb, \\ &\qquad\quad aaa, aab, aba, abb, baa, bab, bba, bbb, \ldots \end{aligned}$$

Exercice 5.2

Exercice 5.2

Exercice 5.2

- On commence en simulant 1 étape de M sur le mot w_1 ; si M accepte en 1 étape, on s'arrête aussi en acceptant

Exercice 5.2

- On commence en simulant 1 étape de M sur le mot w_1 ; si M accepte en 1 étape, on s'arrête aussi en acceptant

- Sinon, on simule 2 étapes de M sur le mot w_1, puis 2 étapes sur le mot w_2 ; si M accepte l'un des deux mots en 2 étapes, on s'arrête aussi en acceptant

Exercice 5.2

- On commence en simulant 1 étape de M sur le mot w_1 ; si M accepte en 1 étape, on s'arrête aussi en acceptant

- Sinon, on simule 2 étapes de M sur le mot w_1, puis 2 étapes sur le mot w_2 ; si M accepte l'un des deux mots en 2 étapes, on s'arrête aussi en acceptant

- Sinon, on simule 3 étapes de M sur les mots w_1, w_2 et w_3, en acceptant si elle accepte l'un des mots

Exercice 5.2

- On commence en simulant 1 étape de M sur le mot w_1 ; si M accepte en 1 étape, on s'arrête aussi en acceptant

- Sinon, on simule 2 étapes de M sur le mot w_1, puis 2 étapes sur le mot w_2 ; si M accepte l'un des deux mots en 2 étapes, on s'arrête aussi en acceptant

- Sinon, on simule 3 étapes de M sur les mots w_1, w_2 et w_3, en acceptant si elle accepte l'un des mots

- …

Exercice 5.2

- On commence en simulant 1 étape de M sur le mot w_1 ; si M accepte en 1 étape, on s'arrête aussi en acceptant

- Sinon, on simule 2 étapes de M sur le mot w_1, puis 2 étapes sur le mot w_2 ; si M accepte l'un des deux mots en 2 étapes, on s'arrête aussi en acceptant

- Sinon, on simule 3 étapes de M sur les mots w_1, w_2 et w_3, en acceptant si elle accepte l'un des mots

- …

- Sinon, on simule t étapes de M sur les mots $w_1, w_2, \ldots, w_t$, en acceptant si elle accepte l'un des mots

Exercice 5.2

Exercice 5.2

- Ça revient à faire la réduction (fonction calculable) $f\big(\langle M\rangle\big) = \langle M'\rangle\#w$ avec n'importe quel w fixé, par exemple $w = abba$, et M' définie par

Exercice 5.2

- Ça revient à faire la réduction (fonction calculable) $f\big(\langle M\rangle\big) = \langle M'\rangle\#w$ avec n'importe quel w fixé, par exemple $w = abba$, et M' définie par

$M'(x) =$

si $x = w$ **alors**

soit $w_1, w_2, \ldots$ l'énumération de $\Sigma^\star$

pour $t := 1$ **à** ∞ **faire**

pour $i := 1$ **à** t **faire**

simuler t étapes de M sur w_i

si M s'arrête **alors** accepter

sinon accepter

Exercice 5.2

Exercice 5.2

- Si $\langle M \rangle \in L_\infty$ alors M ne s'arrête sur aucune entrée w_i, c'est-à-dire que pour tout $t \in \mathbb{N}$, M ne s'arrête pas en t étapes sur aucun mot

Exercice 5.2

- Si $\langle M \rangle \in L_\infty$ alors M ne s'arrête sur aucune entrée w_i, c'est-à-dire que pour tout $t \in \mathbb{N}$, M ne s'arrête pas en t étapes sur aucun mot

$M'(x) =$

si $x = w$ **alors**

soit $w_1, w_2, \ldots$ l'énumération de $\Sigma^\star$

pour $t := 1$ **à** ∞ **faire**

pour $i := 1$ **à** t **faire**

simuler t étapes de M sur w_i

si M s'arrête **alors** accepter

sinon accepter

Exercice 5.2

- Si $\langle M \rangle \in L_\infty$ alors M ne s'arrête sur aucune entrée w_i, c'est-à-dire que pour tout $t \in \mathbb{N}$, M ne s'arrête pas en t étapes sur aucun mot

$M'(x) =$

  si $x = w$ **alors**

    soit $w_1, w_2, \ldots$ l'énumération de $\Sigma^\star$

    pour $t := 1$ **à** ∞ **faire**

      pour $i := 1$ **à** t **faire**

        simuler t étapes de M sur w_i

        si M s'arrête **alors** accepter

  sinon accepter

- Donc M' ne s'arrête pas sur w, ce qui implique $\langle M' \rangle \# w \in L_2$

Exercice 5.2

Exercice 5.2

- Si $\langle M \rangle \notin L_\infty$ alors M s'arrête sur une entrée w_i, c'est-à-dire que pour quelque $t \in \mathbb{N}$, M s'arrête en t étapes sur w_i

Exercice 5.2

- Si $\langle M \rangle \notin L_\infty$ alors M s'arrête sur une entrée w_i, c'est-à-dire que pour quelque $t \in \mathbb{N}$, M s'arrête en t étapes sur w_i

$M'(x) =$

si $x = w$ **alors**

soit $w_1, w_2, \ldots$ l'énumération de $\Sigma^\star$

pour $t := 1$ **à** ∞ **faire**

pour $i := 1$ **à** t **faire**

simuler t étapes de M sur w_i

si M s'arrête **alors** accepter

sinon accepter

Exercice 5.2

- Si $\langle M \rangle \notin L_\infty$ alors M s'arrête sur une entrée w_i, c'est-à-dire que pour quelque $t \in \mathbb{N}$, M s'arrête en t étapes sur w_i

$M'(x) =$

si $x = w$ **alors**

soit $w_1, w_2, \ldots$ l'énumération de $\Sigma^\star$

pour $t := 1$ **à** ∞ **faire**

pour $i := 1$ **à** t **faire**

simuler t étapes de M sur w_i

si M s'arrête **alors** accepter

sinon accepter

- Donc M' s'arrête en acceptant sur w, ce qui implique $\langle M' \rangle \# w \notin L_2$

Exercice 5.2

- Si $\langle M \rangle \notin L_\infty$ alors M s'arrête sur une entrée w_i, c'est-à-dire que pour quelque $t \in \mathbb{N}$, M s'arrête en t étapes sur w_i

$M'(x) =$
    si $x = w$ **alors**
        soit $w_1, w_2, \ldots$ l'énumération de $\Sigma^\star$
        pour $t := 1$ **à** ∞ **faire**
            pour $i := 1$ **à** t **faire**
                simuler t étapes de M sur w_i
                si M s'arrête **alors** accepter
    sinon accepter

- Donc M' s'arrête en acceptant sur w, ce qui implique $\langle M' \rangle \# w \notin L_2$

N’hésitez pas à poser des questions !

aeporreca.org/forum-calculabilite

N’hésitez pas à poser des questions !

aeporreca.org/forum-calculabilite

Bon courage !

Bon courage !