Mise au propre des TDs de calculabilité

Yann Miguel

18 février 2020

Table des matières

1 TD1

1.1 Exercice 1

1.1.1 Question 1

Cinq éléments de l'ensemble ℕ x {0, 1, a} sont:

1. (0, 0)
2. (0, 1)
3. (0, a)
4. (1, 0)
5. (1, 1)

Cela revient donc à faire le produit carthésien entre l'ensemble ℕ et
l'ensemble {0, 1, a}.

1.1.2 Question 2

Une bijection de ℕ dans 2ℕ est:

$$f \;:\; x \to 2x$$

1.1.3 Question 3

Une bijection de ℕ dans ℕxℕ est :

$$g \;:\; (x, y) \to \frac{(x+y)(x+y-1)}{2} + y$$

1.1.4 Question 4

Une bijection de ℕ dans ℕxℕxℕ est:

$$h \;:\; (x, y, z) \to g(g(x, y), z)$$

1.2 Exercice 2

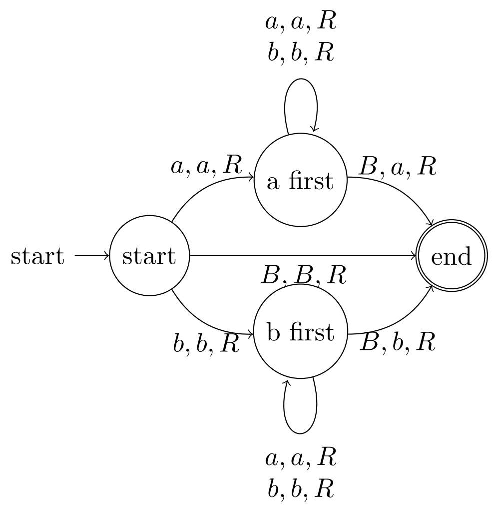

FIGURE 1 – Cette machine de turing rajoute la première lettre d'un mot à la fin du dit mot.

1.3 Exercice 3

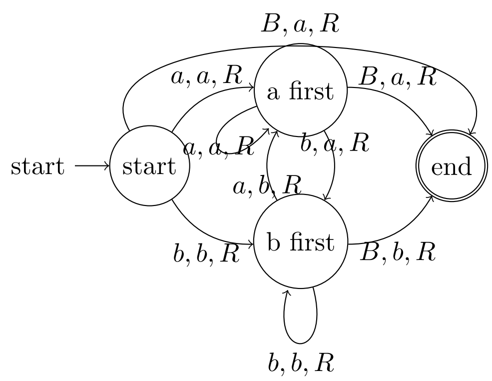

FIGURE 2 – Machinde de turing de décalage avec ajout de a au début

1.4 Exercice 4

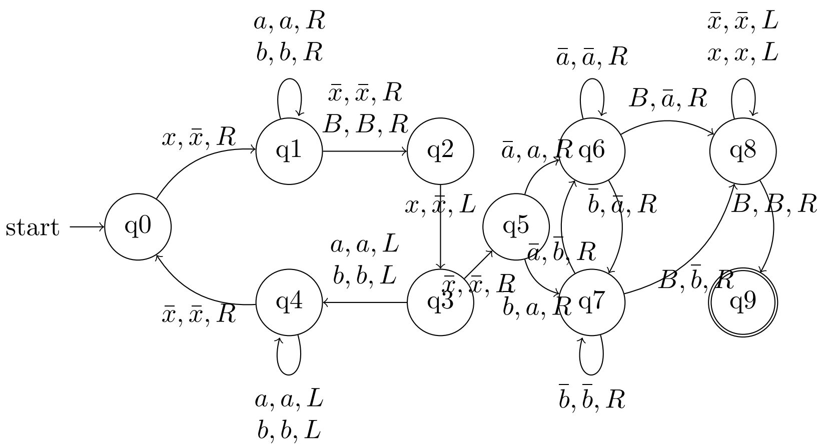

Figure 3 – Utilisation d'une MT pour en faire une autre

1.5 Exercice 5

1.5.1 Question 1

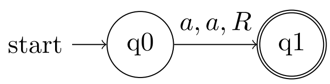

Figure 4 – Vérifie qu'un mot commence par a

1.5.2 Question 2

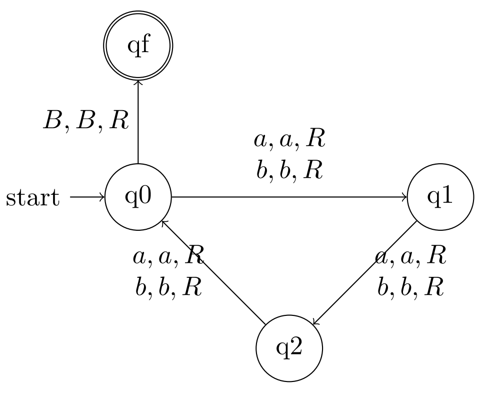

Figure 5 – Longueur du mot est modulo 3

1.5.3 Question 4

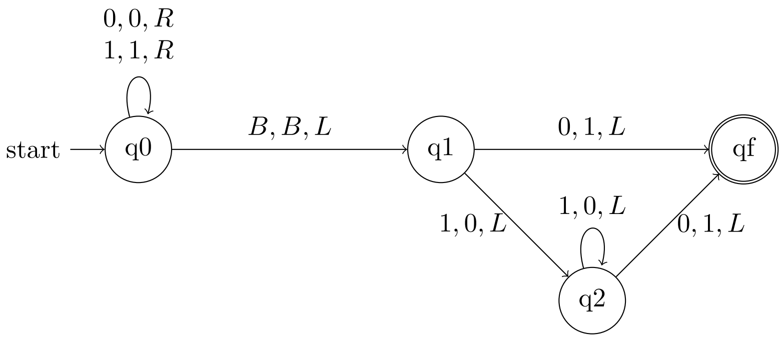

Figure 6 – Fonction d'incrémentation binaire

1.5.4 Question 5

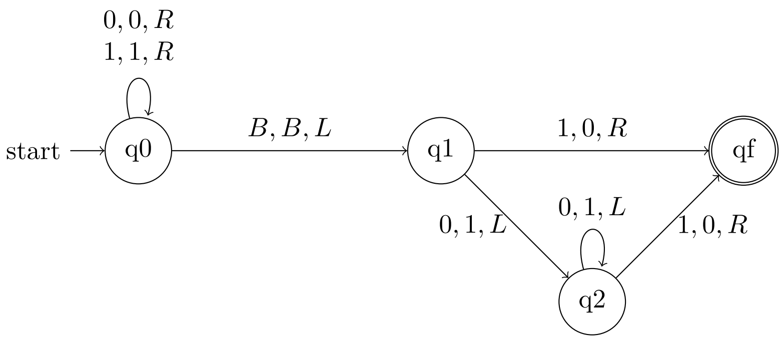

Figure 7 – Fonction de décrémentation binaire

1.5.5 Question 6

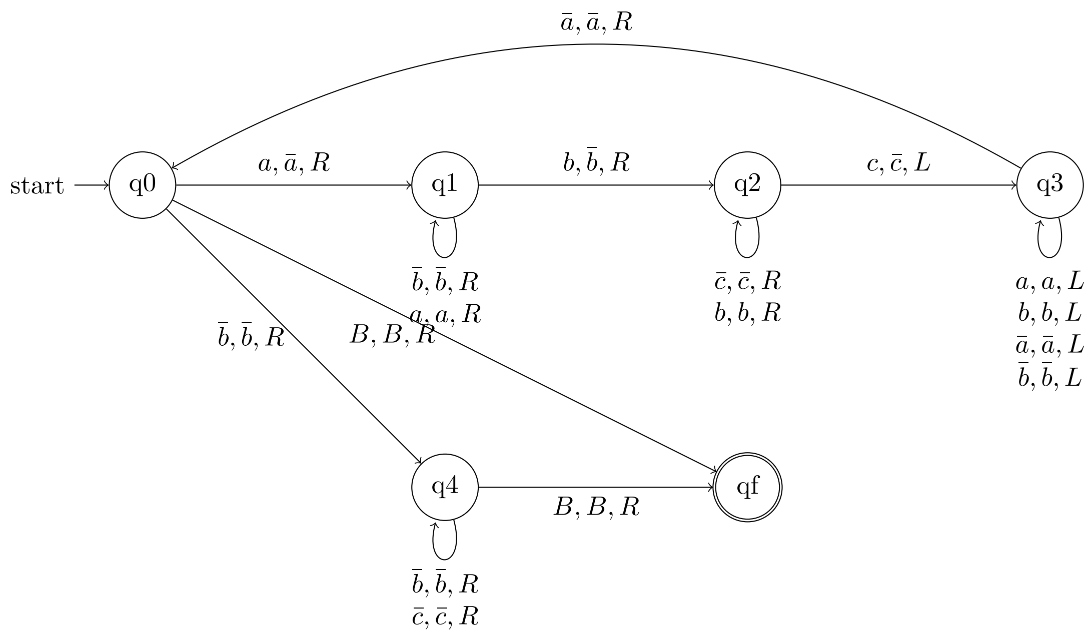

FIGURE 8 – Autant de a que de b que de c

1.5.6 Question 7

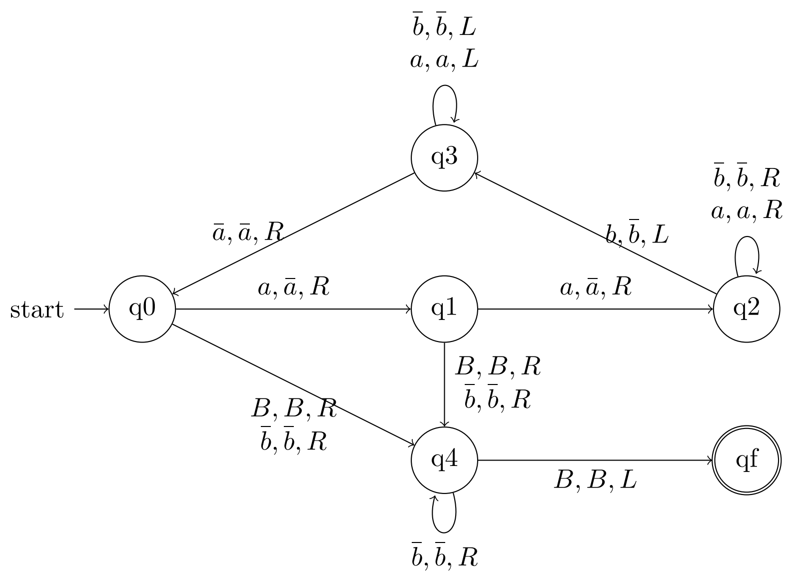

FIGURE 9 – Moitié moins de b que de a

2 TD2

2.1 Exercice 1

2.1.1 Question 1

`Oui.`

2.1.2 Question 2

`Oui.`

2.1.3 Question 3

`Oui.`

2.2 Exercice 4

2.2.1 Questions 1 et 2

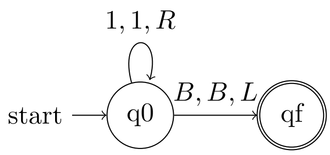

FIGURE 10 – Machine acceptant tout les mots composés que de 1 ou le mot nul

2.2.2 Question 3

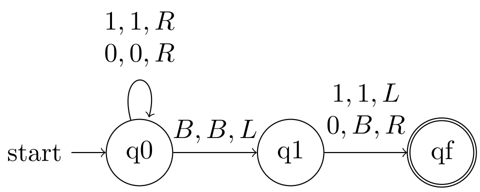

FIGURE 11 – Machine en binaire qui retourne l'entier si il est impair ou sa moitié si il est pair

2.2.3 Question 4

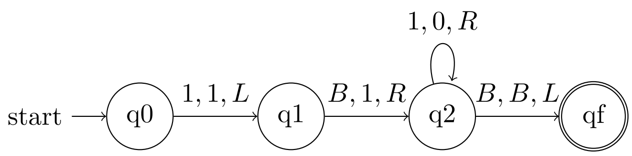

Figure 12 – Machine qui prends un nombre en unaire et renvoie son exposant de 2 en binaire

2.3 Exercice 5

2.3.1 Question 1

On peux faire cela facilement avec une machine à deux curseurs, un tout à gauche du mot, et un tout à droite.

2.3.2 Question 2

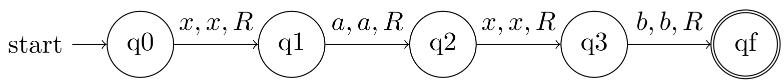

Figure 13 – Machine vérifiant que la seconde lettre du mot est un a et que la quatrième lettre du mot est un b

2.3.3 Question 3

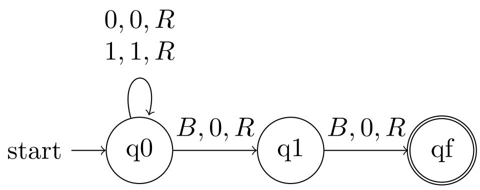

Figure 14 – Machine prenant un entier en binaire et sortant son multiple par 4 en binaire (4=2x2)

2.3.4 Question 4

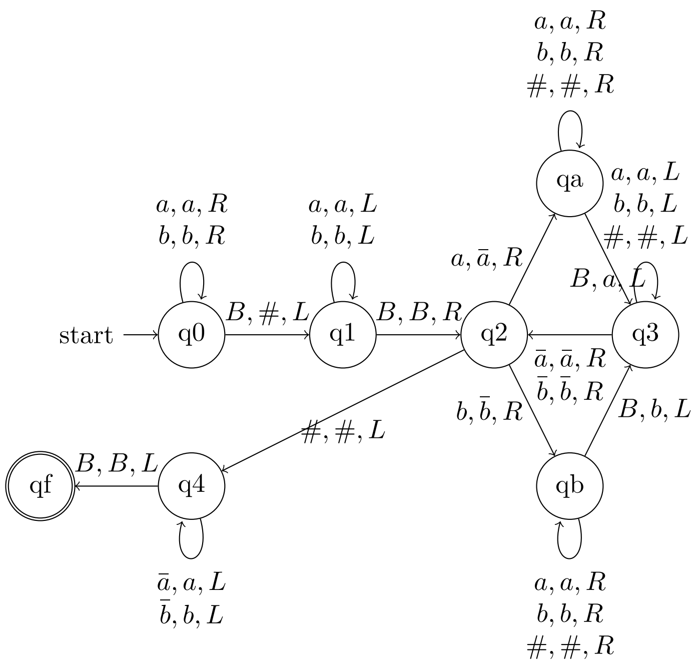

Figure 15 – Machine de Σ^* vers Γ^* telle que f(w)=w#w

2.3.5 Question 5

```
Définition machine:
 - on compare les tailles, refusé si x2 a une taille inférieure à
   x1.
 - si ils sont de tailles égales, on fait une comparaison bit à bit.
```

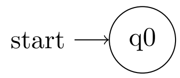

Figure 16 – Machine reconnaissant x1#x2 avec x1 < x2 (en binaire)

2.4 Exercice 6

2.4.1 Question 1

2.4.2 Question 2

2.4.3 Question 3

$$[0,1] \ \subseteq \ \mathbb{R} \ \Rightarrow \ |[0,1]| \ \leq \ |\mathbb{R}|$$
$$\arctan(2x+1) \ \textit{est une surjection de } [0,1] \textit{ vers } \mathbb{R}$$

2.4.4 Question 4

$$f \ : \ x \ \rightarrow \ (x,1) \ \textit{est une injection de } \mathbb{R} \textit{ dans } \mathbb{R}\times\mathbb{R}$$
$$\textit{Entremeller deux réels est une injection de } \mathbb{R}\times\mathbb{R} \textit{ dans } \mathbb{R}$$

2.5 Exercice 7

```
Encodage d'une machine:
 - commence par 111 pour indiquer le début de la machine
 - on mets autant de 0 que d'états
 - on mets 11 pour indiquer la fin des états
 - on mets autant de 0 que de lettres sur le ruban de travail
 - on mets 11 pour indiquer la fin des lettres
 - on encode les transitions, en les séparant par des 11, et on encode
   les états et symboles par un nombre de 0 équivalent à leur position
   (de 1 à n) tout en séparant chaque partie de la transuition par
   1
 - on note R=0 et L=00
 - on termine la machine par 111
Application:
```

$$(M) = 111 \ 000 \ 11 \ 000 \ 11 \ 01010010010 \ 11 \ 001001001010 \ 11 \ 00100010001000100 \ 111$$

```
Des espaces ont été ajoutés afin de clarifier la lecture.
```

2.6 Exercice 8

2.6.1 Question 1

```
Il faut dessiner une machine de Turing qui reconnait ce langage.
```

2.6.2 Question 2

```
Machine récursivement énumérable:
 - La machine va avoir un #, puis un état ou elle va écrire tout
   les mots du langage à gauche du #.
 - Après avoir écrit chaque mot, il va y avoir une transition vers
   une autre machine qui va recopier le mot à droite du #.
 - On remets la tête de lecture sur le #, et on entre l'état d'énumération.
```

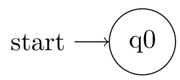

Figure 17 – Machine d'un langage récursivement énumérable

2.7 Exercice 9

Ce qu'on sait:

1. La machine parcours t étapes, ayant chacune une transition.
2. La machine peux aller à droite, ou à gauche.
3. On a un alphabet τ, avec des mots de longueur s.
4. Une machine déterministe qui repasse par une même configuration veut dire qu'elle repasse par un état.

Ce qu'on peux en conclure:

1. Donc, on peux consommer au maximum autant de cases qu'il y a eu d'étapes.
2. Un mouvement à gauche après un mouvement à droite ne consomme pas de nouvelle case, le minimum étant donc deux cases quand on ne fait que aller à droite puis à gauche.
3. On a donc au maximum τ^s mots, ce qui donne au plus $|\tau|^s(s+1)|Q|$ configurations.
4. La machine ne s'arrête jamais, donc, si la machine s'arrête, cela veut dire qu'elle ne visite qu'une seule configuration à chaque fois. Ce qui veut dire que le nombre de cases utilisées est bornée par le temps qu'on passe dans la machine, et aussi que le temps qu'on passe dans la machine est borné par le nombre de cases qu'on utilise.

3 TD3

3.1 Exercice 1

3.1.1 Question 1

L'énoncé est vrai car le théorème de l'arrêt dit exactement cela.

3.1.2 Question 2

L'énoncé est faux, car il existe une machine, la machine qui accepte tout, qui va tout le temps s'arrêter, peu impote les entrées.
Si les entrées s'arrêtent, on prends la machine qui s'arrête tout le temps. Sinon, on prends celle qui ne s'arrête jamais.
Par conséquent, $\forall\langle\ M\ \rangle\ , w\ \exists\ M_{halt}(\langle\ M\ \rangle\ , w)\ =\ halt(\langle\ M\ \rangle\ , w)$

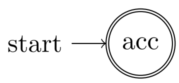

Figure 18 – Machine qui accepte tout

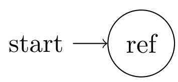

Figure 19 – Machine qui refuse tout/tourne en boucle à l'infini

3.2 Exercice 2

La réduction est une fonction $f:\ \langle\ M\ \rangle\ \rightarrow f(\langle\ M\ \rangle\ )$.

3.2.1 Question 1

g(M) est une machine qui supprime aa puis lance la machine M.
Réduction:
- Si M s'arrête sur 0, g(M) commence par supprimer aa, puis fait tourner M sur ce qu'il reste,
  c'est-à-dire 0, et donc g(M) termine sur aa.
- g(M) s'arrête sur aa. Hors, g(M) commence par supprimer aa puis lance la machine M, qui s'arrête sur 0.
  Donc, si g(M) s'arrête sur aa, M va s'arrêter sur 0, qui est le reste.

Donc, par définition de f, $\langle\ M\ \rangle\ \in L_{halte} \leftrightarrow f(\langle\ M\ \rangle\ ) \in A$.

Le langage L_{halte} n'étant pas récursif, sa réduction ne l'est pas non plus.
Cette réduction pouvant se faire dans l'autre sens, les deux problèmes sont de difficulté équivalentes.

3.2.2 Question 2

f(M) est une fonction qui prends $\langle M \rangle$ et un mot w et qui donne $\langle M' \rangle$.
Cette fonction dit que M' accepte a si et seulement si M accepte w.

On veut donc que la machine M', machine associée à f($\langle M \rangle$), supprime a, écrive w, retourne à gauche du mot, et lance M.

Réduction:

- M accepte w par définition de L_u.
  Si on lance la machine M' sur a,
  alors elle supprime le a, écrit le w, retourne à gauche du mot,
  et accepte w.
  Par conséquent, la machine M' a accepté l'entrée a, et f($\langle M \rangle$ # w) $\in$ B.
- Si f($\langle M \rangle$ # w) $\in$ B, alors la machine M' accepte a.
  Or, M' efface a, écrit w et lance M sur w, ce qui veut dire que
  M accepte w.
  Donc, $\langle M \rangle$ # w$\in L_u$.

3.2.3 Question 3

f(M) est une fonction qui prends $\langle M \rangle$ et un mot w et qui donne $\langle M' \rangle$.
Cette fonction dit que M' refuse w mais accepte bbw si et seulement si M refuse w.

On veut donc que la machine M' puisse vérifier si son entrée est bbw.
Si c'est le cas, elle l'accepte. Sinon, elle lance M.

Réduction:

- Si $\langle M \rangle$ # w $\in L_{\bar{u}}$, alors M n'accepte pas w.
  Posons $\langle M' \rangle$ # w = f($\langle M \rangle$), w $\neq$ bbw,
  alors si on lance M' sur w, elle lance M sur w, et n'accepte pas.
  De plus, par définition, de M', M' accepte bbw. Donc, f($\langle M \rangle$) $\in$ C.
- Si $\langle M' \rangle$ # w = f($\langle M \rangle$) $\in$ C.
  - M' accepte bbw
  - M' n'accepte pas w
  - Donc, M' sur w lance M sur w, donc M n'accepte pas w.

  Donc, $\langle M \rangle$ # w $\in L_{\bar{u}}$.

3.2.4 Question 4

f(M) est une fonction qui prends un mot w et qui retourne le même mot précédé par un a. Cette fonction dit que M' accepte aw si et seulement si M accepte w.

Il suffit juste que M' prenne un mot w accepté par M, lui rajoute un a en première lettre, et accepte le mot.

- Si L est récursif, on ne peux rien en déduire.
- Si aL est récursif, alors L est récursif.
- Si L n'est pas récursif, alors aL ne l'est pas.
- Si L est récursivement énumérable, alors on ne peux rien en déduire.

- Si aL est récursivement énumérable, alors L est récursivement énumérable.
- Si L n'est pas récursivement énumérable, alors aL ne l'est pas.

3.2.5 Question 5

f(M) est une fonction qui prends un mot aw, et qui retourne le mot w, $\forall$w.

Supposons $\exists$ u$\notin$L, et f(w)=u si w ne commences pas par a.

1. Montrer que, si w$\in$aL$\Rightarrow$f(w)$\in$L.
   - Soit w$\in$aL$\Rightarrow$w=av, avec v$\in$L
   - donc f(w)=v et f(w)$\in$L.
2. Montrer que f(w)$\in$L$\Rightarrow$w$\in$aL.
   - Soit w tel que f(w)$\in$L, alors f(w)$\neq$u
   - et w commence donc par a, c'est-à-dire w=av, donc f(w)=v
   - et v$\in$L donc av$\in$aL.

Donc, on a bien une réductione de aL vers L, donc:
- Si L est récursif, alors aL l'est aussi.
- Si aL est récursif, alors L l'est aussi(question précédente).
- Si L est récursivement énumérable, alors aL l'est aussi.
- Si aL est récursivement énumérable, alors L l'est aussi(question précédente).

3.2.6 Question 6

M_1 est la machine de Turing qui s'arrête tout le temps. M_2 est la machine de Turing qui tourne en boucle à L'infini.

$\langle\ M_1\ \rangle \# \in L_u . \langle\ M_2\ \rangle \# \notin L_u$.

f est la fonction telle que f(a)=$\langle\ M_1\ \rangle$ # et $\forall$ v$\neq$a, f(v)=$\langle\ M_2\ \rangle \#$.

1. Montrer que w$\in L_{stupide}$ $\Rightarrow$f(w)$\in L_u$.
   - Soit w$\in L_{stupide}$ $\Rightarrow$w=a.
   - f(w)=$\langle\ M_1\ \rangle$ # $\Rightarrow$ f(w)$\in L_u$.
2. Montreer que f(w)$\in L_u$ $\Rightarrow$w$\in L_{stupide}$
   - Soit w tel que f(w)$\in L_u$ $\Rightarrow$f(w)$\neq\langle\ M_2\ \rangle$ #.
   - $\Rightarrow$f(w)=$\langle\ M_1\ \rangle$ # donc w=a.
   - Donc, w$\in L_{stupide}$.

3.3 Exercice 3

3.3.1 Question 1

D = {$\langle\ M\ \rangle$ |M s'arrête sur ab et ba}

1. Créer une réduction entre ce langage et un non décidable.
2. Utiliser le théorème de Rice.

Cette propriété étant non-triviale, d'après le théorème de Rice, ce langage n'est pas décidable. Or, dans cet exercice, on veut faire des réductions. Donc, on doit trouver une réduction.

L_{halte} = {$\langle\ M\ \rangle$ |M s'arrête sur l'entrée vide} n'est pas décidable.

Donc, si on réduit L_{halte} à D, on prouvera que D n'est pas décidable.

On doit donc trouver une fonction qui va de L_{halte} vers D.

Soit f la fonction telle que f($\langle M \rangle$) = $\langle M' \rangle$ où M' est la machine qui:

- sur ab accepte
- sur ba efface ba, et lance M
- sur le reste refuse

De plus, $\forall$ w qui n'est pas un code de MT, f(w)=w.

1. Montrer que w$\in L_{halte} \Rightarrow$f(w)$\in$D.
   - soit w$\in L_{halte} \Rightarrow$w=$\langle M \rangle \in L_{halte}$.
   - donc, f(w) est un code de machine, et la machine associée a des propriétés similaires à M'.
   - Donc, f(w)$\in$D.
2. Montrer que $\forall$w, f(w)$\in$D$\Rightarrow$w$\Rightarrow L_{halte}$
   - f(w)=$\langle M' \rangle \Rightarrow$ w = $\langle M \rangle$
   - comme $\langle M' \rangle$ donc M' s'arrête sur ba.
   - Mais, sur ba, M' efface ba et lance M
   - donc M s'arrête sur l'entrée vide
   - donc w = $\langle M \rangle \in L_{halte}$.

Donc, on a bien $\langle M \rangle \in L_{halte} \Leftrightarrow$ f(w)$\in$D.

Étant donné que L_{halte} n'est pas décidable, la réduction prouve que D n'est pas décidable non plus.

3.3.2 Question 2

Langage E$\times$F, avec:

- E={$\langle M \rangle$ | b$\in$L(M)}
- F={$\langle M \rangle$ | a$\in$L(m) ou b$\in$L(M)}

E$\times$F est une paire de mots tel que le premier mot est dans E, et le second mot est dans F.

On va construire une réducion afin de prouver que ce langage n'est pas décidable.

On prends M_F, la machine qui accepte tout, afin d'artificiellement oublier le langage F.

On ignore la seconde coordonnée car elle accepte tout le temps.

Soit f, la fonction telle que f(M)=$\begin{pmatrix} \langle M' \rangle \\ \langle M_F \rangle \end{pmatrix}$,

où M' est la machine qui efface b et lance M, et accepte si M s'arrête.

1. Montrer que $\langle M \rangle \in L_{halte} \Rightarrow$f($\langle M \rangle$)$\in$E$\times$F.
   - Soit $\langle M \rangle \in L_{halte}$ et $\begin{pmatrix} \langle M' \rangle \\ \langle M_F \rangle \end{pmatrix}$= f($\langle M \rangle$)
   - Si on lance M' sur l'entrée b, alors elle lance M sur l'entrée vide, et M s'arrête, donc M' accepte, donc M'$\in$E.
   - Donc, f($\langle M \rangle$)$\in$E$\times$F.
2. Montrer que f($\langle M \rangle$)$\in$E$\times$F$\Rightarrow\langle M \rangle \in L_{halte}$
   - Soit $\langle M \rangle$ tel que f($\langle M \rangle$)$\in$E$\times$F.
   - $\Rightarrow$f($\langle M \rangle$)=$\begin{pmatrix} \langle M' \rangle \\ \langle M_F \rangle \end{pmatrix}$ tel que $\langle M' \rangle$)$\in$E.

- Donc, $\langle\ M'\ \rangle$) accepte b, or, sur b, $\langle\ M'\ \rangle$) efface b et lance M sur le vide, et accepte si et seulement si M s'arrête sur l'entrée vide.
- Or, vu que M' accepte, M s'arrête sur l'entrée vide.
- Donc, $\langle\ M\ \rangle\ \in$L$_{halte}$.

Donc, on a bien une réduction de L$_{halte}$ vers E$\times$F, et on sait que L$_{halte}$ n'est pas décidable.
Par conséquent, E$\times$F n'est pas décidable.

3.3.3 Question 5

$\forall$ MT M, L$_M$={w|w$\in$L(M)} est RE.
L$_M$={w| w est accepté par M} donc, L$_M$ est RE.

3.3.4 Question 7

I={$\langle\ M\ \rangle$ |$\langle\ M\ \rangle$ < $2^{2^{1024}}$ et L(M)=a}
À travers le $2^{2^{1024}}$, on limite le nombre de machines, ce qui crée un langage fini, car il a un nombre de machines finies.
Or, un langage fini est forcément décidable.
Par conséquent, ce langage est décidable.