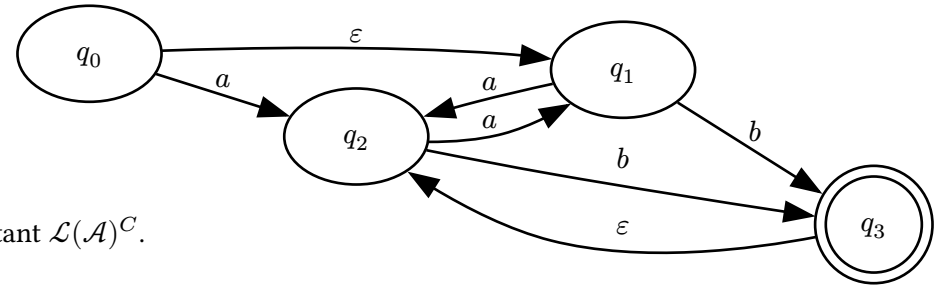
Sujet IMT-2



I - Automates

On considère l'automate $\mathcal{A}$ ci-contre.

1. Déterminiser et compléter l'automate.
2. Donner une expression régulière dénotant $\mathcal{L}(\mathcal{A})^C$.

II - Verrous

0. Rappeler les conditions que doit satisfaire un verrou utilisable.

II.a - Deux fils d'éxecution

L'objectif est de créer un verrou pour deux fils d'execution.

II.a.1 - Un premier essai

```
1: function CreateLock()
2:     return {flag = [false, false]}

1: function Unlock(m, t)
2:     m.flag[t] ← false
```

```
1: function Lock(m, t)
2:     other ← 1 − t
3:     while m.flag[other] do
4:         nothing
5:     m.flag[t] ← true
```

1. Cet essai satisfait-il l'exclusion mutuelle ?

II.a.2 - Une autre méthode

```
1: function CreateLock()
2:     return {turn = 0}

1: function Unlock(m, t)
2:     m.turn ← 1 − t
```

```
1: function Lock(m, t)
2:     other ← 1 − t
3:     while m.turn == other do
4:         nothing
```

2. Cet essai satisfait-il l'exclusion mutuelle ?
3. Que dire de l'absence d'interblocage ?

II.a.3 - Encore une autre méthode

```
1: function CreateLock()
2:     return {turn = 0, want = {false,false}}

1: function Unlock(m, t)
2:     m.turn ← 1 − t
3:     m.want[t] ← false
```

```
1: function Lock(m, t)
2:     other ← 1 − t
3:     m.turn ← t
4:     m.want[t] ← true
5:     while m.turn == other ∧ m.want[other] do
6:         nothing
```

4. Cet essai satisfait-il l'exclusion mutuelle ?
5. Modifier cette version pour que cela constitue un verrou utilisable.
6. Montrer qu'elle garantit l'absence de famine. En déduire sur l'absence d'interblocage.

II.b - Plusieurs fils d'execution

7. Nommer et décrire précisemment un algorithme permettant un verou pour n fils d'éxecution.