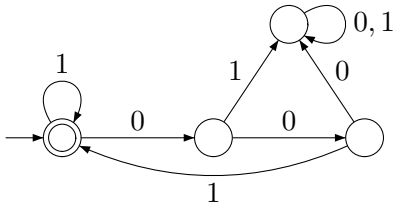
Examen du cours «Analyse d'algorithmes» Session 2

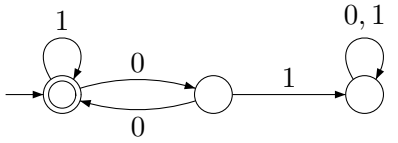
Rappel : seule une feuille A4 de notes personnelles est autorisée pendant l'examen. Tout autre document est interdit.

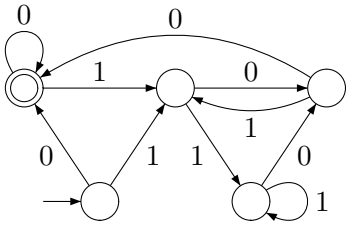
Soyez concis et précis. On rappelle que le soin apporté aux justifications et aux preuves est déterminant.

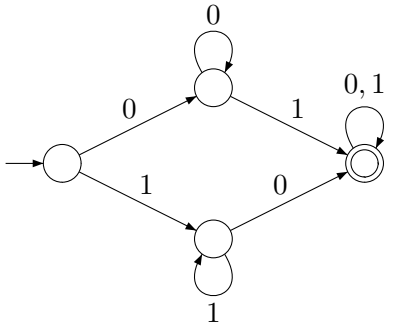
1 Automates finis

Indiquez pour les automates finis et les langages réguliers suivants, quel langage est reconnu par quel automate. Attention, l'un des automates ne reconnait aucun des langages listés ici, et inversement, l'un des langages n'est reconnu par aucun des automates dessinés ci-dessous.

1.


2.


3.


4.


A. le langage des mots sur $\{0,1\}$ qui ne contiennent soit que des 0, soit que des 1.

B. le langage des mots sur $\{0,1\}$ qui représentent les encodages binaires des multiples de 4.

C. le langage des mots sur $\{0,1\}$ qui contiennent au moins un 0 et au moins un 1.

D. le langages des mots sur $\{0,1\}$ où tous les 0 apparaissent par paires, et où toute paire de 0 est séparée de la suivante par au moins un 1.

2 Preuve d'algorithmes

On considère le while-program suivant qui calcule le produit de `a` par `b` par la méthode dichotomique.

```
1    p := 0; b' := b; a' := a;
2    tantque (b' > 0) faire
3      si (b' mod 2 = 1) alors
4        p := p + a'
5      sinon
6        skip
7      finsi;
8      a' := a' * 2;
9      b' := b' div 2
10   fintantque
```

Les opérateurs `mod` et `div` calculent respectivement le reste et le quotient de la division entière. Par conséquent, `b' mod 2 = 1` est vraie lorsque la valeur de `b'` est impaire.

2.1 Correction

Nous cherchons à prouver que ce while-program est correct vis-à-vis de la précondition et de la postcondition suivantes :

- PRECOND : $(\mathtt{a} \in \mathbb{N}) \wedge (\mathtt{b} \in \mathbb{N})$
- POSTCOND : $(\mathtt{p} = \mathtt{a} \times \mathtt{b})$

1. Prenons par exemple $\mathtt{a} = 13$ et $\mathtt{b} = 7$, le tableau suivant donne les valeurs de `p`, `a'` et `b'` à chaque itération. L'étape i correspond à la valeur des variables en ligne 2 du while-program après i exécutions des lignes 3 à 9.

étape	p	a'	b'
0	0	13	7
1	13	26	3
2	39	52	1
3	91	104	0

   En vous basant sur cet exemple, donnez une formule R liant le produit de `a` par `b` aux valeurs de `p`, `a'` et `b'` qui soit vraie à chaque étape.

2. Prouvez à l'aide de la logique de Hoare que la formule $I = R \wedge (\mathtt{b}' \in \mathbb{N})$ est invariante par la boucle `tantque` (lignes 3 à 9).

3. En vous appuyant sur la règle de Hoare pour l'instruction `tantque`, montrez que la formule I permet de déduire la postcondition du while-program.

4. Enfin, pour terminer la preuve de correction du while-program, nous devons considérer les trois affectations de la ligne 1. Quelle formule doit-on substituer à Q ci-dessous pour conclure la preuve ? On ne demande pas de prouver votre réponse, mais de bien l'argumenter.

   $\{(\mathtt{a} \in \mathbb{N}) \wedge (\mathtt{b} \in \mathbb{N})\}$
   `p := 0 ; b' := b ; a' := a ;`
   $\{Q\}$

2.2 Terminaison

Nous voulons maintenant montrer que le while-program termine à l'aide de la méthode des ensembles bien fondés.

1. Quelle mesure de terminaison proposez-vous ? Justifiez votre réponse.
2. À l'aide de la mesure que vous avez proposée à la question précédente et de la méthode des ensembles bien fondés, prouvez que le while-program termine.

3 Complexité algorithmique

On s'intéresse au while-programme suivant qui calcule le produit de `a` par `b`, pour `a` et `b` naturels, par la méthode itérative.

```
1   p := 0;
2   tantque (b > 0) faire
3      p := p + a;
4      b := b - 1
5   fintantque
```

1. Dans un premier temps, nous considérons que le coût des algorithmes réside dans le nombre d'itérations de boucles nécessaires au calcul.

   (a) Quelle est la complexité temporelle asymptotique de ce while-program ? Justifiez votre réponse.

   (b) Considérons maintenant le while-programme de l'exercice 2. Calculez sa complexité temporelle asymptotique (votre calcul justifiera votre réponse).

2. Afin de comparer les deux while-programs, nous devons prendre en compte le coût des opérations qu'ils réalisent dans leurs boucles respectives. Sachant que les nombres sont encodés en binaire, quel algorithme recommandez-vous et pourquoi ?

Règles de la logique de Hoare

Règle de conséquence :

$$\frac{P' \Rightarrow P, \{P\}\, S\, \{Q\}\ ,\ Q \Rightarrow Q'}{\{P'\}\, S\, \{Q'\}}$$

Règle d'affectation :

$$\frac{}{\{P[e/x]\}\, x{:=}e\, \{P\}}$$

Règle de séquence :

$$\frac{\{P\}\, S_1\, \{Q\}\ ,\ \{Q\}\, S_2\, \{R\}}{\{P\}\, S_1;S_2\, \{R\}}$$

Règle du «`si`» :

$$\frac{\{P \wedge B\}\, S_1\, \{R\}\ ,\ \{P \wedge \neg B\}\, S_2\, \{R\}}{\{P\}\, \texttt{si}\ B\ \texttt{alors}\ S_1\ \texttt{sinon}\ S_2\ \texttt{finsi}\, \{R\}}$$

Règle du «`tantque`» :

$$\frac{\{P \wedge B\}\, S\, \{P\}}{\{P\}\, \texttt{tantque}\ (\texttt{B})\ \texttt{faire}\ \texttt{S}\ \texttt{fintantque}\, \{P \wedge \neg B\}}$$

Terminaison des while-programs

On rappelle que pour prouver la terminaison d'une boucle :

```
tantque B faire
  S
fintantque
```

Il s'agit de trouver un terme t appelé «mesure» qui satisfait :

1. t est à valeurs dans un ensemble bien fondé $(\mathbb{D}, \leq)$: $\{B \wedge t \in \mathbb{D} \wedge P\}\, S\, \{t \in \mathbb{D}\}$
2. t est strictement décroissant : $\{B \wedge t = X \wedge P\}\, S\, \{t < X\}$

où P est n'importe quelle formule invariante : $\{P \wedge B\}\, S\, \{P\}$, X est une variable qui n'apparaît pas dans le while-program, et $t < X$ est défini par $t \leq X \wedge X \not\leq t$.