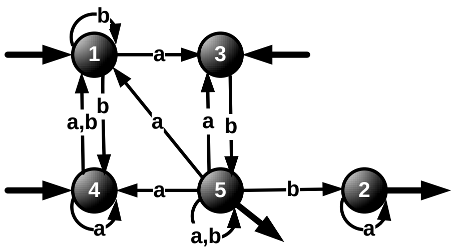
DE : Mathématiques pour l'informatique

Solutions

Exercice 1
A = {a,b} est l'alphabet.
Soit l'automate A_{init} :



	état	a	b
E	1	3	1, 4
S	2	2	--
E	3	--	5
E	4	1, 4	1
S	5	1, 3, 4, 5	2, 5

a) L'automate A_{init} est-il déterministe ? pourquoi ? Si la réponse est « non », donnez toutes les raisons.

Solution : il n'est pas détérministe car :

- Il y a 3 entrées
- Deux transitions en b prennent origine dans 1 : 1b1 et 1b4
- Deux transitions en b prennent origine dans 4 : 4a1 et 4a4

b) Construisez un automate standard A_{st} en standardisant A_{init}. Si vous pensez que A_{init} reconnaît le mot vide, construire un autre automate B_{st} qui ne le fait pas mais reconnaît tout le reste du langage reconnu par A_{init}. Si vous pensez que A_{init} ne le reconnaît pas, expliquez pourquoi.

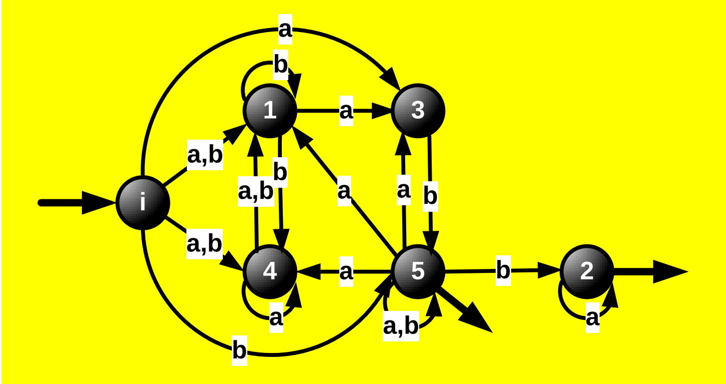
Solution :

- on ajoute un nouvel état i, l'entrée
- 1, 3 et 4 ne sont plus des entrées
- i n'est pas une sortie car aucune entrée de A_{init} ne l'est, A_{init} ne reconnait pas le mot vide
- Transitions de i :

La liste des transitions originant dans une entrée de A_{init}	Transitions de i engendrées par celles-ci
1a3	ia3
1b1	ib1
1b4	ib4
3b5	ib5
4a1	ia1
4a4	ia4
4b1	ia1

Cela résulte en l'automate standard suivant :

	état	a	b
E	i	1, 3 ,4	1, 4, 5
	1	3	1, 4
S	2	2	--
	3	--	5
	4	1, 4	1
S	5	1, 3, 4, 5	2, 5

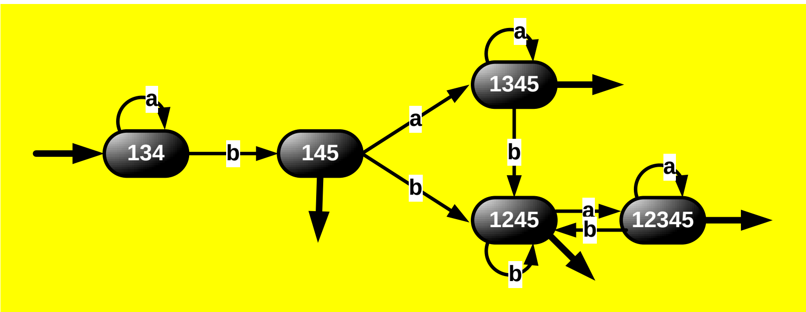


c) Indépendamment de la question (a), obtenir un automate déterministe complet équivalent à A_{init}.

Solution :

	état	a	b
E	1 3 4	1 3 4	1 4 5
S	1 4 5	1 3 4 5	1 2 4 5
S	1 3 4 5	1 3 4 5	1 2 4 5
S	1 2 4 5	1 2 3 4 5	1 2 4 5
S	1 2 3 4 5	1 2 3 4 5	1 2 4 5

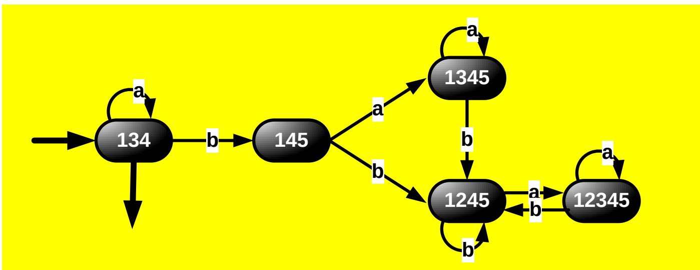
Cet automate est assez compact pour qu'on puisse le dessiner :



d) Construire un automate reconnaissant le langage complémentaire à celui reconnu par A_{init}.

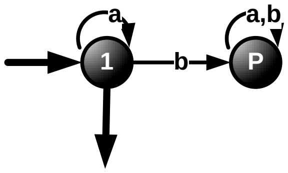
Solution : On prend l'automate déterministe complet qu'on vient d'obtenir et on y effectue l'opération T ↔NT :

	état	a	b
E/S	1 3 4	1 3 4	1 4 5
	1 4 5	1 3 4 5	1 2 4 5
	1 3 4 5	1 3 4 5	1 2 4 5
	1 2 4 5	1 2 3 4 5	1 2 4 5
	1 2 3 4 5	1 2 3 4 5	1 2 4 5

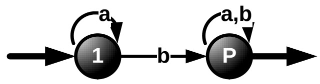


Remarque importante même si ceci n'a pas été demandé dans l'énoncé :

On voit très facilement que l'automate complémentaire reconnait L=a*. Donc l'automate d'origine reconnait A* \ a* : tous les mots sur {a,b} sauf le mot vide et sauf des séquences de a. L'automate reconnaissant juste a* peut immédiatement être minimisé, sans aucune procédure, comme



ce qui signifie que l'automate minimal équivalent à A_{init} est

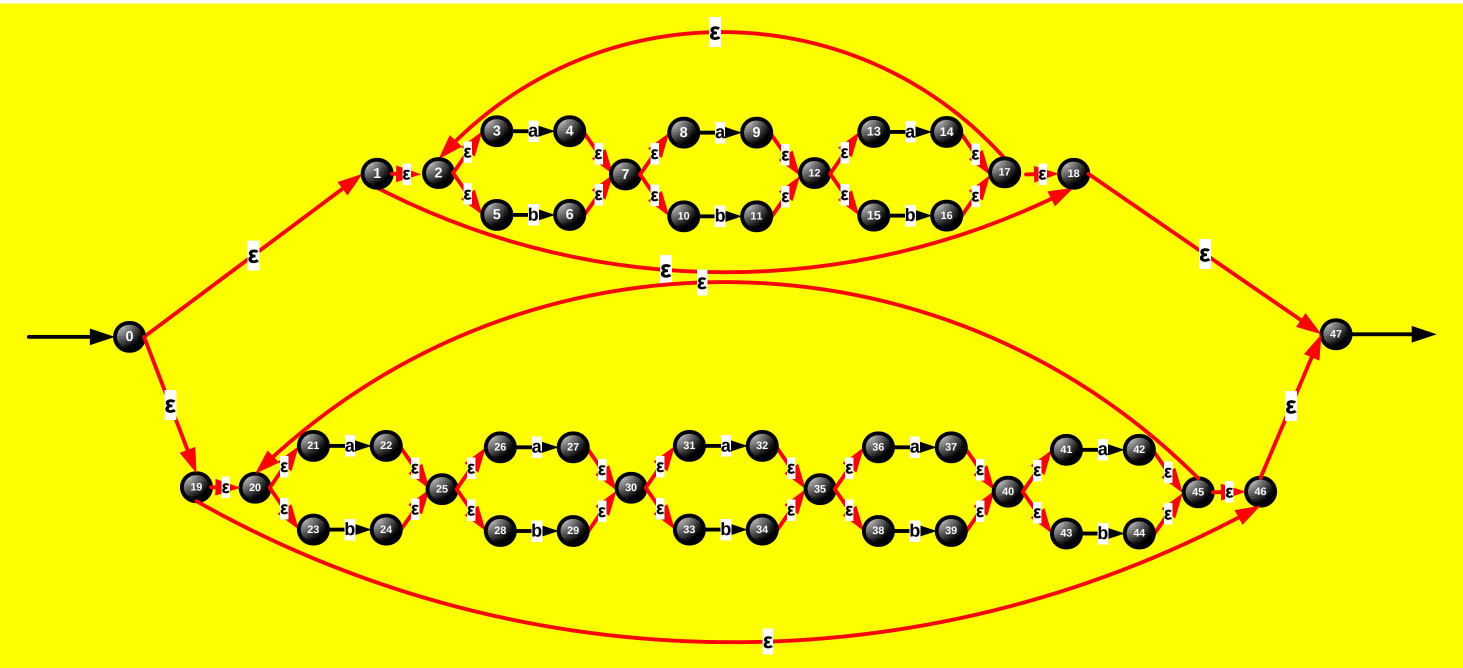


et son langage (a+b)* \ a*peut être écrit explicitement comme a*b(a+b)*. Un bonus a été accordé à ceux qui ont remarqué ceci entièrement ou partiellement.

Exercice 2

a) Construire un automate asynchrone reconnaissant le langage
$$L = \{\big((a+b)(a+b)(a+b)\big)^* + \big((a+b)(a+b)(a+b)(a+b)(a+b)\big)^*\}$$

Solution :



Ensuite**, soit**

b) déterminiser et
c) minimiser

l'automate obtenu en (a) (c'est assez compliqué !),

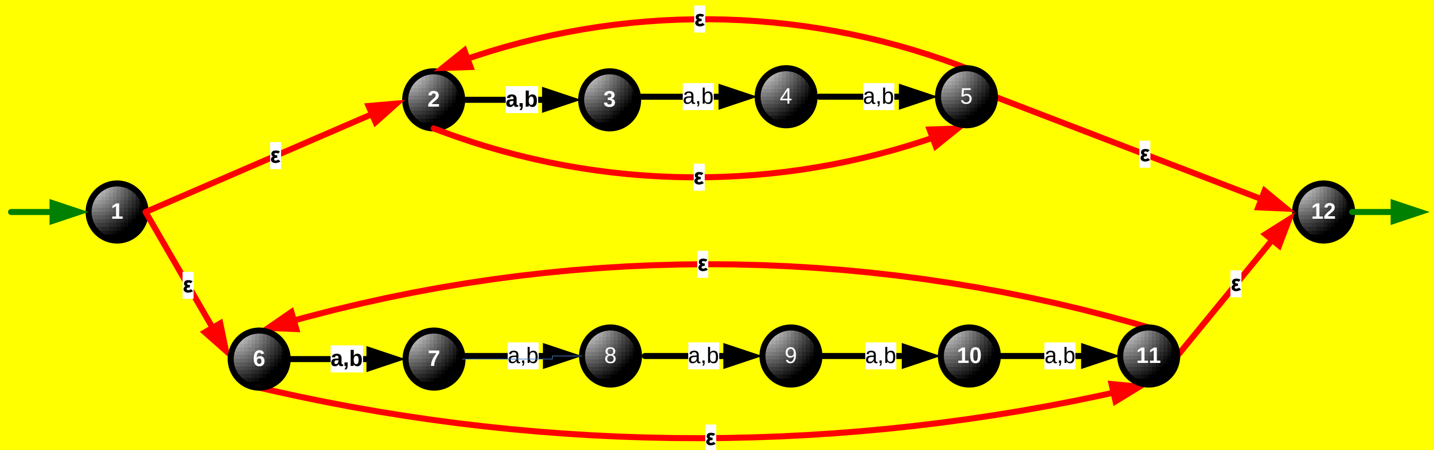
soit

b) simplifier graphiquement et déterminiser et
c) minimiser

l'automate simplifié graphiquement.

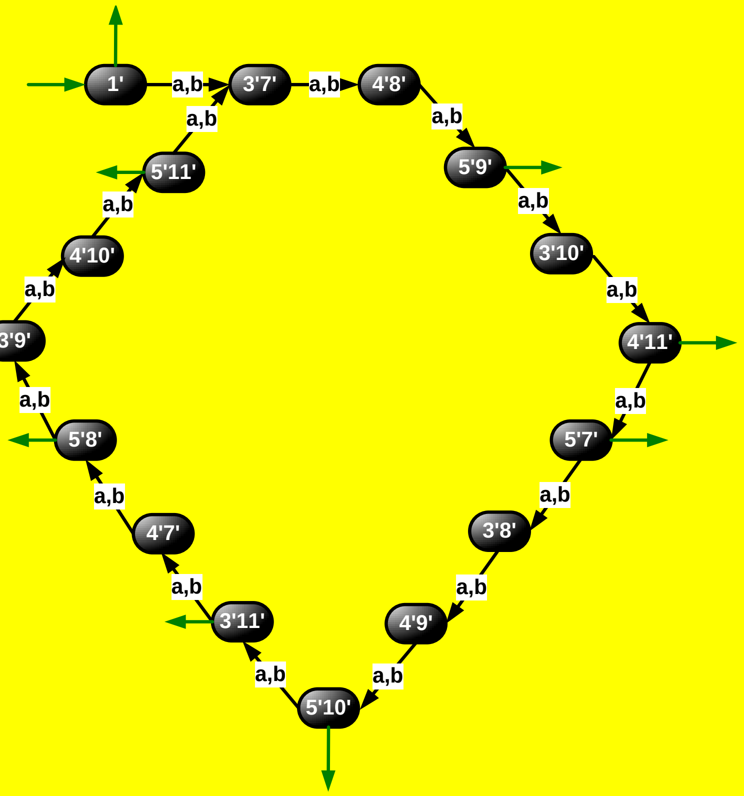
La deuxième proposition est bien plus simple à réaliser.

Solution : je choisis la deuxième option. L’automate simplifié peut avoir la forme suivante :

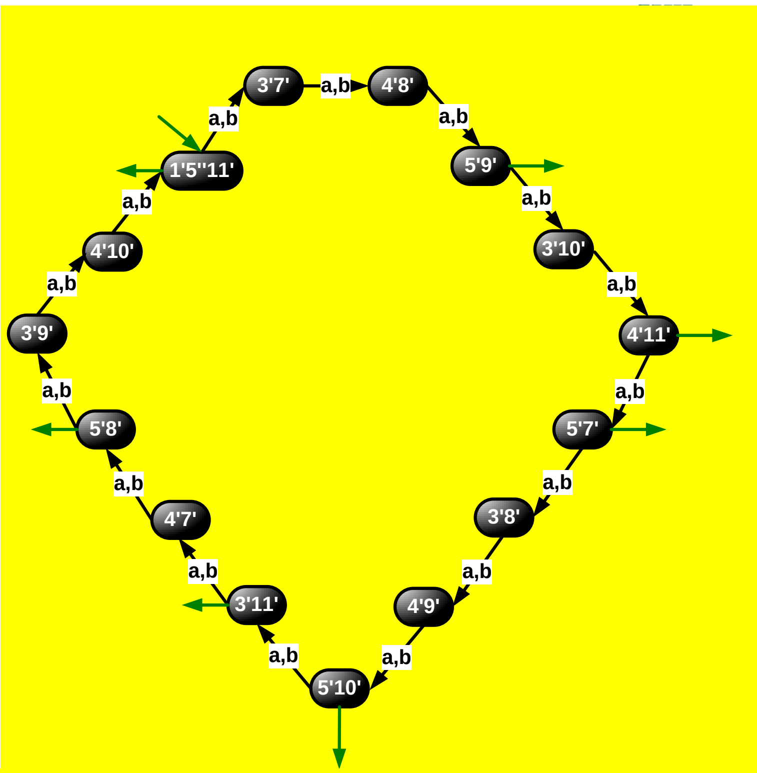


J’introduis une notation raccourcie :
1’=1 2 5 6 11 12 ; 3’=3 ; 4’=4 ; 5’=2 5 12 ; 7’=7 ; 8’=8 ; 9’=9 ; 10’=10 ; 11’=6 11 12. Dans ces termes, voici la table de déterminisation avec 16 états :

	état	a ou b
ES	1’	3’7’
	3’7’	4’8’
	4’8’	5’9’
S	5’9’	3’10’
	3’10’	4’11’
S	4’11’	5’7’
S	5’7’	3’8’
	3’8’	4’ 9’
	4’ 9’	5’10’
S	5’10’	3’11’
S	3’11’	4’7’
	4’7’	5’8’
S	5’8’	3’9’
	3’9’	4’10’
	4’10’	5’11’
S	5’11’	3’7’



Je ne vais pas détailler la minimisation, elle se passe entièrement de la même façon qu’on a vu en classe avec des automates similaires : les seuls deux états qui vont se fusionner, sont 1’ et 5’11, et on obtient un automate minimal en cycle composé de 15 états :



Remarque. Cet automate minimal aurait pu être obtenu directement, car on sait que le résultat pour une ER de la forme $(c^n)^* + (c^m)^*$, au cas où n et m sont mutuellement premier, comme ici (n=3, m=5) est un cycle dont le nombre d'états est n*m=15, av ec des sorties qui sont positionnées dans tous les multiples de 3 et tous les multiples de 5.

Exercice 3

a) Construire un automate qui reconnait tous les nombres écrits en binaire **non divisibles** par 4.

Solution

On construit un automate dont les états correspondent aux nombres qui sont 0,1,2 et 3 modulo 4 :

N	N mod 4	en collant un 0 à droite: 2N	(2N) mod 4	en collant un 1 à droite: 2N+1	
4n	0	8n	0	8n+1	1
4n+1	1	8n+2	2	8n+3	3
4n+2	2	8n+4	0	8n+5	1
4n+3	3	8n+6	2	8n+7	3

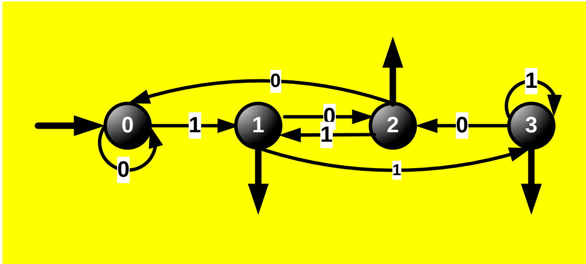
On obtient la table de transitions suivante :

état	0	1
0	0	1
1	2	3
2	0	1
3	2	3

Ici, l'entrée se fait toujours en 0, et la ou les sorties sont choisie(s) selon les restes de la division entière par 4 qu'on veut voir acceptés. Pour que le nombre ne soit pas divisible par 4, il faut des sorties en 1, 2 et 3.

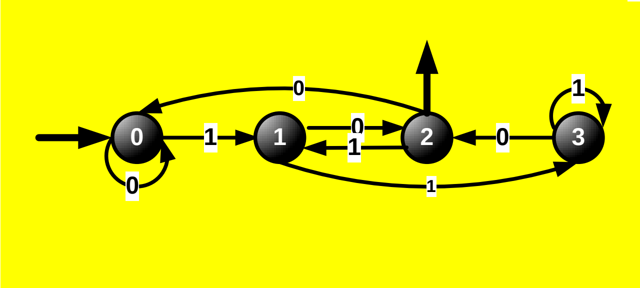
Donc la réponse à la question (a) est

	état	0	1
E	0	0	1
S	1	2	3
S	2	0	1
S	3	2	3



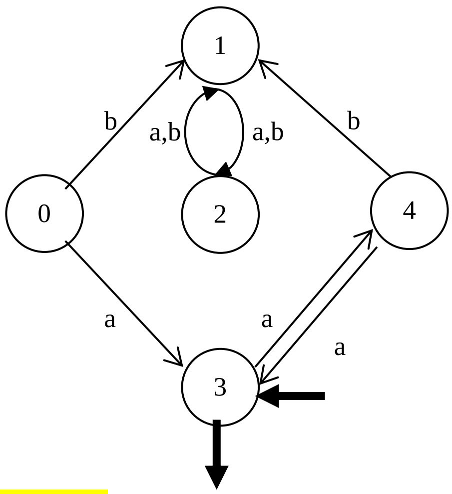
b) Construire un automate qui reconnait tous les nombres écrits en binaire **qui sont 2(mod 4)** (c'est-à-dire le reste de la division entière par 4 est 2).

Solution : Il suffit de mettre la seule sortie en 2 :

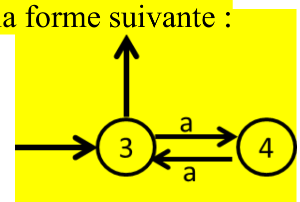


Exercice 4

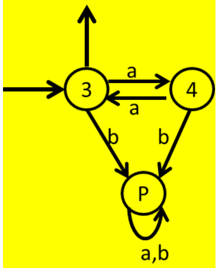
Minimiser l'automate suivant:



Solution :
Il y a deux façons de procéder : l'une est directe, où on ne « remarque » rien, l'autre intélligente, où on analyse l'automate avant de procéder à la minimisation.
Commençons par la seconde façon. Il est facile de remarquer que les états 0,1 et 2 sont inutiles, plus précisément, ils sont non coaccessibles. L'automate donc peut être réduit à la forme suivante :



Pour rendre cet automate complet sur A={a,b} , il faut ajouter une poubelle :



Il est facile à voir que cet automate est minimal.

Maintenant obtenons le même automate complet minimal par une procédure « normale » . D'abord il faut compléter l'automate :

		a	b
	0	3	1
	1	2	2
	2	1	1
E/S	3	4	P
	4	3	1
	P	P	P

La partition initiale Θ_0 = {(3), (0,1,2,4,P)}={(3), NT}. L'état-groupe 3 ne subira aucun changement. Pour les autres états,

				sous Θ_0	
NT	0	3	1	3	NT
	1	2	2	NT	NT
	2	1	1	NT	NT
	4	3	1	3	NT
	P	P	P	NT	NT

Donc Θ_1 = {(3), (0,4), (1,2,P)}. On notera les groupes par leurs composition : 3, 04, 12P. On obtient

			sous Θ_1	
0	3	1	3	12P
4	3	1	3	12P
1	2	2	12P	12P
2	1	1	12P	12P
P	P	P	12P	12P

Donc aucun séparation ne se produit, et $\Theta_2=\Theta_1=\Theta_{fin}$. On obtient la table de transitions :

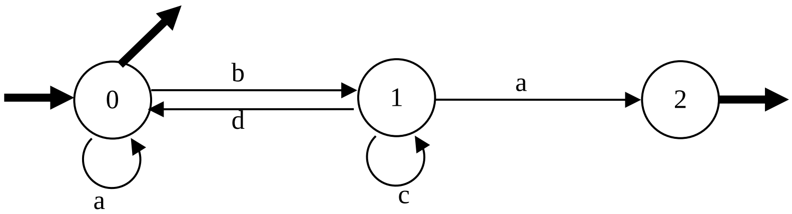
		a	b
E/S	3	04	P
	04	3	P
	12P	12P	12P

Donc l'automate qu'on a déjà obtenu suite à l'approche « intelligente ».

Attention !
1) Avant de minimiser, l'automate doit être déterministe **complet** ;
2) La partition initiale consiste toujours en DEUX groupes (dont exceptionnellement un peut être vide), pas en trois ! On ne doit pas séparer l'état poubelle dans un groupe à part pour la partition initiale ! Si la poubelle veut bien rester seule, elle le fera d'elle-même.
3) l'unicité de l'automate minimal concerne uniquement l'automate déterministe **complet**.

Exercice 5

Soit l'automate représenté ci-dessous :



a) écrire les équations correspondants et
b) les résoudre

pour obtenir une expression rationnelle correspondant au langage reconnu par cet automate.

Aucune expression « intuitive » (qui n'est pas résultat de solution d'un système d'équations) ne sera acceptée.

Solution :

a) 0=ε+0a+1d (eq 1)
1=0b+1c (eq 2)
2=1a (eq 3)

L = 0+2

b) Eq 2 ⇒ 1=0bc* (lemme d'Arden). En mettant cela dans eq 1, on obtient 0=ε+0a+0bc*d =ε+0(a+bc*d), d'où 0=ε(a+bc*d)*=(a+bc*d)*.
Alors 1= 0bc*=(a+bc*d)*bc* et 2=(a+bc*d)*bc*a.
Le langage reconnu par l'automate est L=(a+bc*d)*bc*a+(a+bc*d)*=(si l'on préfère) =(a+bc*d)*(bc*a+ε).

Exercice 6

Prouver par récurrence que

a) $7^n - 2^n$ est divisible par 5 pour tout entier $n \geq 1$.

Solution :

1) Base : n=1, 7^0 - 2^0 = 5 divisible par 5. OK.
2) Hérédité : P(n)= (∃q∈**N** | $7^n - 2^n = 5q$)
$7^{n+1} - 2^{n+1} = 7 \times 7^n - 2\times 2^n = (5+2)\, 7^n - 2\times 2^n = 5\times 7^n + 2(7^n - 2^n)$
= [if P(n)=vrai] $5\times 7^n + 2\times 5q$ ce qui est divisible par 5. OK.

Remarque. Un nombre étonnant d'élèves pensent que
$\mathbf{(7^n - 2^n)(7 - 2) = 7^{n+1} - 2^{n+1}}$.

Cela m'inquiète beaucoup. Révisez votre algèbre élémentaire.

b) $n^2 < 2^n$ pout tout entier $n \geq 4$. **Rectification : à n=4, il y a égalité. Donc il faut remplacer** $n \geq 4$ par n>4. C'est bien plus intelligent que corriger en remplaçant $n^2 < 2^n$ **par** $n^2 \leq 2^n$ car si l'on peut prouver une assertion plus forte, on ne veut pas la remplacer par une assertion faible.

Aucun point n'a été accordé à ceux qui ont juste constaté l'inexactitude de l'énoncé.

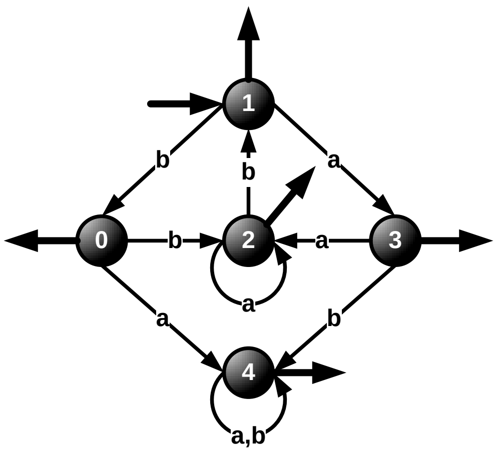
Solution :

1) Base : n=5 : $5^2=25<2^5=32$. OK.
2) Hérédité :

Soit P(n) : $n^2 < 2^n$. Alors $2n^2 < 2\times2^n=2^{n+1}$. Or, $2n^2=n^2+n^2> n^2+2n+1=(n+1)^2$ pour tout entier n≥3, car $n^2>2n+1$ pour tout réel $n>1+\sqrt{2}$, donc tout entier n≥3. Donc, pour n≥3, si $n^2 < 2^n$, alors $(n+1)^2<2^{n+1}$. Nous somme dans le cas n>4, donc c'est bon.

Questions théoriques

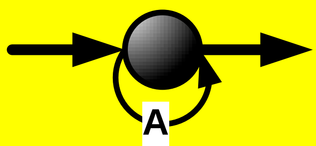
1. Voici un automate fini sur l'alphabet A={a,b}. Dire sans aucun calcul :
   a) quel langage il reconnait



Solution : c'est un ADC dont tous les états sont terminaux. Il reconnait A*=(a+b)*.

   b) toujours sans calcul, construire l'automate déterministe complet minimal équivalent.

Solution : l'automate minimal reconnaissant A* est



2. Quelles conditions doivent-elles être vérifiées pour qu'on puisse dire qu'un mot est reconnu par un automate Ã ? (4 lignes)

   Réponse : il doit y avoir un chemin correspondant à ce mot entre un état initial et un état terminal de Ã.

3. Comment peut-on déterminer que deux automates finis reconnaissent le même langage ? (3 lignes)

   Réponse : il reconnaissent le même langage ssi les automates déterministes complètes minimaux équivalents aux deux automates en question sont identiques.