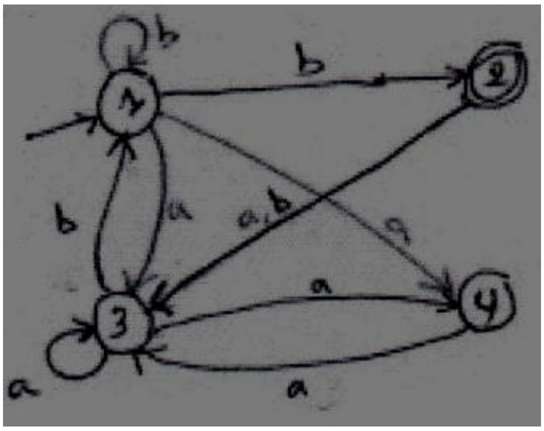
Université de Batna 2
Département d'informatique
2ème année Licence académique
Module : théorie des langages

Correction de la série de la semaine:
L'automate d'états finis

Objectif du TD **: est d'appliquer quelques algorithmes sur un automate d'états finis tel que l'algorithme de *déterminisation* et l'algorithme de *minimisation*.**

<u>Exercice</u> :

Soit l'automate d'états finis caractérisé par le graphe de transition suivant :



☞**L'étudiant doit savoir quand est qu'un automate d'états finis n'est pas <u>déterministe</u>.**
On dit qu'un automate d'états finis n'est pas <u>déterministe</u> s'il contient plus d'un état initial ou plus d'une transition à partir d'un même état en lisant le même symbole.

Juste une remarque : On remarque qu'il y a un arc de *l'état 2* vers *l'état 3* étiqueté par a, b
Ça veut dire, on fait le passage de *l'état 2* vers *l'état 3* en lisant a ou b

1) Cet automate est déterministe ? Sinon; le déterminiser.

<u>Réponse :</u>

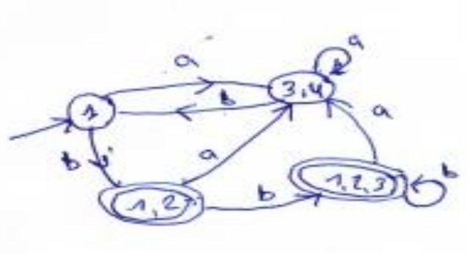
✍ L'automate n'est pas déterministe puisque il contient par exemple deux transitions sur le même symbole *a* à partir du même *état 1*

Selon l'algorithme de détermination :

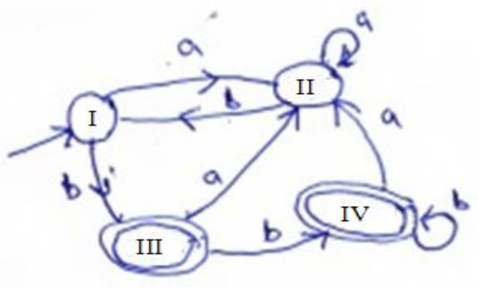
- ✓ Regroupez les états initiaux, cet ensemble devient le nouveau état initial de l'automate déterministe
  ✍ Dans notre exercice, le nouveau état initial contient seulement *l'état 1*
- ✓ Rajoutez dans la table de transition de l'automate déterministe tous les nouveaux états produits avec leurs transitions.
- ✓ Recommencer cette dernière étape jusqu'à ce qu'il n'y ait plus de nouveau état.
- ✓ Chaque nouveau état qui contient au moins un état final devient final
- ✓ On peut renommer les nouveaux états.
  ✍ Donc, on trouve cette table de transition:

état	a	b
→ {1}	{3,4}	{1,2}
{3,4}	{3,4}	{1}
{1,2}	{3,4}	{1,2,3}
{1,2,3}	{3,4}	{1,2,3}

✍ D'où le graphe de l'automate déterministe est représenté comme suit :



✍ On peut renommer maintenant notre automate déterministe comme suit :



☞**L'étudiant doit savoir quand est qu'un état est <u>accessible</u>.**
Un état est <u>accessible</u> s'il existe un chemin à partir d'un état initial vers lui.
Remarque : chaque état initial est accessible.
☞**L'étudiant doit savoir quand est qu'un état est <u>co-accessible</u>.**
Un état est co-<u>accessible</u> s'il existe un chemin à partir de cet état vers un état final.
Remarque : chaque état final est co-<u>accessible</u>.

2) Minimiser l'automate déterministe

<u>Réponse :</u>

Selon l'algorithme de minimisation :

- ✓ On élimine de l'automate tous les états inaccessibles et non co-accessibles avec leurs transitions.

✍ Dans notre exercice, on n'élimine rien puisqu'il n'y a pas d'état inaccessible
et puisqu'il n'y a pas d'état non co-accessible

- ✓ On partitionne les états en deux classes, une classe pour les états finaux et une autre classe pour le reste des états.

✍ Dans notre exercice, on trouve: A = {III, IV} B = {I, II}

- ✓ La troisième étape consiste à construire une table de transition entre les classes et voir le résultat de transition des états de chaque classe obtenue sur chaque symbole. Ensuite, on doit éclater en d'autres classes seulement la classe d'états ayant des résultats différents de transition sur un symbole quelconque.

✍ En premier temps, on a les transitions suivantes:

$\partial(III, a) =$ **II** l'état **II** où **se trouve ? il est dans la classe B**

$\partial(III, b) =$ **IV** l'état **IV** où **se trouve ? il est dans la classe A**

$\partial(IV, a) =$ **II** l'état **II** où **se trouve ? il est dans la classe B**

$\partial(IV, b) =$ **IV** l'état **IV** où **se trouve ? il est dans la classe A**

Même chose avec la suite de la table et on trouve:

		Les symboles	
Classe obtenue	Ses états	*a*	*b*
A	III	**B**	**A**
	IV	**B**	**A**
B	I	**B**	**A**
	II	**B**	**B**

✍ Dans la dernière table, on trouve que:

La *classe A* ne s'éclate pas puisque tous ses états ont sur chaque symbole le même résultat de transition. Donc A reste elle-même A = {III, IV}

La *classe B* doit être éclatée puisque ses états ont sur le symbole *b* des résultats différents de transition. C'est-à-dire la *classe A* sera éclaté en des classes de telle sorte que chacune contient des états qui ont le même résultat de transition sur chaque symbole. Ainsi La *classe B* sera éclatée en deux classes *classe B et classe C*

B = {I} C= {II}

- ✓ Recommencer la troisième étape jusqu'à ce qu'il n'y ait plus d'éclatement possible.

		Les symboles	
Classe obtenue	Ses états	*a*	*b*
A	III	**C**	**A**
	IV	**C**	**A**
B	I	**C**	**A**
C	II	**C**	**B**

✍ Comme il n'y ait plus d'éclatement, on s'arrête.

- ✓ La dernière table décrit l'automate minimal.
- ✓ Chaque classe de cette dernière table est un état de l'automate minimal.
- ✓ La classe qui contient l'état initial devient le nouveau état initial.
- ✓ Toute classe contenant un état final devient un nouveau état final.

✍ On résume la dernière table et on obtient :

	Les symboles	
Classe obtenue	*a*	*b*
A	**C**	**A**
→ **B**	**C**	**A**
C	**C**	**B**

✍ D'où le graphe de l'automate minimal est représenté comme suit :