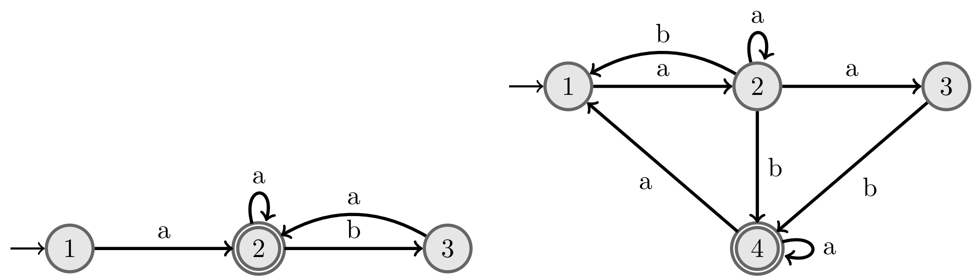


(a) Automate pour l'Exercice 2.

(b) Automate pour l'Exercice 4.

Figure 1 – Automates pour les exercices de complémentation et de déterminisation.

Licence Sciences et Technologies
Univ. Grenoble Alpes

Examen à mi-parcours
INF 302 : Langages et Automates
L2, 2025/2026

Rappel à propos des consignes et quelques conseils et remarques

— Durée : 2 heures.
— Aucune sortie avant 30 minutes. Aucune entrée après 30 minutes.
— 1 feuille A4 R/V autorisée.
— Tout dispositif électronique est interdit (calculatrice, téléphone, tablette, montre connectée, etc.).
— Le soin de la copie sera pris en compte.
— Les exercices sont indépendants et en ordre croissant de difficulté.
— Le barème est donné à titre indicatif.
— L'examen est sur 22 points, vous devez obtenir 20 points pour obtenir la note maximale.

Exercice 1 (Vrai ou Faux - 4 points)

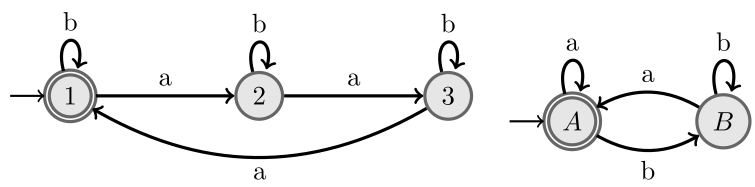
Répondre par vrai ou faux aux questions suivantes. Justifier soigneusement et de façon concise vos réponses (sans preuve). Si une proposition est fausse, répondre par un contre-exemple.

1. **(0,5 pt)** Pour construire un automate déterministe reconnaissant le complémentaire d'un langage L reconnu par un automate A, il suffit d'inverser les états accepteurs et non-accepteurs de A.
2. **(0,5 pt)** Pour tout langage à états L, si $\text{Pref}(L) = \Sigma^*$, alors $L = \Sigma^*$.
3. **(0,5 pt)** Si L_1 et L_2 sont des langages reconnus par des automates déterministes à n_1 et n_2 états respectivement, alors, en ne comptant pas ses états non-accessibles, l'automate produit reconnaissant $L_1 \cap L_2$ possède exactement $n_1 \times n_2$ états.
4. **(0,5 pt)** Pour tout automate fini déterministe, l'automate minimal équivalent est unique à un renommage des états près.
5. **(0,5 pt)** Si deux états p et q d'un automate déterministe sont équivalents alors pour tout $s \in \Sigma$, $\delta(p, s)$ est équivalent à $\delta(q, s)$.
6. **(0,5 pt)** Si un AEFD possède un cycle accessible, alors le langage qu'il reconnaît est infini.
7. **(0,5 pt)** L'automate déterministe résultant de l'algorithme de déterminisation appliqué à un automate non-déterministe est minimal.
8. **(0,5 pt)** Pour deux automates non-déterministes reconnaissant le même langage, les automates obtenus après déterminisation puis minimisation sont les mêmes, à un renommage des états prêts et en ignorant les états non-accessibles.

Exercice 2 (Automate complémentaire - 2 points)

Nous considérons l'automate représenté sur la Figure 1a.

1. Calculer et représenter l'automate complémentaire de cet automate.



(a) (b)

FIGURE 2 – Automates dont il faut calculer le produit.

Exercice 3 (Automate produit et exécution - 3 points)

Nous considérons les automates représentés sur la Figure 2.

1. **(1 pt)** Calculer et représenter l'automate produit de ces deux automates.
2. **(1 pt)** Décrire le langage reconnu par l'automate produit.
3. **(1 pt)** Est-ce que le mot *abaab* est accepté par l'automate produit ? Justifier en donnant l'exécution de ce mot sur cet automate.

Exercice 4 (Automate déterminisé - 4 points)

Nous considérons l'automate représenté sur la Figure 1b.

1. Calculer et représenter le déterminisé de cet automate.

Exercice 5 (Algorithme : impasse - 2 points)

Nous définissons une *impasse* dans un automate comme un état dans l'automate que l'on peut atteindre depuis l'état initial, mais qui ne mène à aucun état accepteur.

1. **(1 pt)** Exprimer la condition d'être sans impasse pour un automate en utilisant les états accessibles et co-accessibles.
2. **(1 pt)** Donner un algorithme, en moins de 5 lignes, qui détermine si un automate donné en entrée est sans impasse. Vous pouvez réutiliser tous les algorithmes vus en cours sans les redéfinir.

Exercice 6 (Automates avec plusieurs états initiaux - 4 points)

Nous souhaitons définir une variante d'automates à états finis non-déterministes où le non-déterminisme est introduit par la possibilité d'avoir plusieurs états initiaux. Les transitions restant comme dans le cas des automates déterministes. Lorsqu'il s'exécute, l'automate peut démarrer simultanément dans plusieurs états initiaux.

1. **(1 pt)** Donner un exemple en représentation graphique d'un tel automate avec moins de 6 états pour reconnaître l'ensemble des mots qui contiennent un nombre pair de a ou un nombre impair de b ou un nombre pair de c.
2. **(1 pt)** Donner une définition pour de tels automates en prenant soin de spécifier les contraintes sur chaque élément impliqué dans la définition.
3. **(1 pt)** Définir les configurations de tels automates.
4. **(1 pt)** Définir la relation de dérivation entre configurations de tels automates.

Exercice 7 (Algorithme : langage à longueur fixe - 3 points)

Un langage L est dit à longueur fixe s'il existe $n \in \mathbb{N} \setminus \{0\}$ tel que tous les mots de L sont de longueur n.

1. **(1 pt)** Donner un exemple et un contre-exemple de langage à longueur fixe.
2. **(2 pt)** Donner un algorithme qui détermine si un automate déterministe donné en entrée est à longueur fixe. Indice : il faut s'inspirer de l'algorithme de parcours en largeur.