SVAM – Introduction et LTL

Pierre-Cyrille Héam

pheam [at] femto-st.fr

Master 2 Informatique

Vol 501 Ariane 5

Vol 501 Ariane 5

Le 04 Juin 1996, le premier vol du lanceur Ariane 5 explose en vol après environ 40 secondes de vol.

- Le système inertiel envoie de mauvaises données au système de pilotage automatique,

Vol 501 Ariane 5

Le 04 Juin 1996, le premier vol du lanceur Ariane 5 explose en vol après environ 40 secondes de vol.

- Le système inertiel envoie de mauvaises données au système de pilotage automatique,
- Le système inertiel de secours se met en route et réagit exactement de la même façon,

Vol 501 Ariane 5

Le 04 Juin 1996, le premier vol du lanceur Ariane 5 explose en vol après environ 40 secondes de vol.

- Le système inertiel envoie de mauvaises données au système de pilotage automatique,
- Le système inertiel de secours se met en route et réagit exactement de la même façon,
- Le pilote automatique, ayant de mauvaises données, ordonne de façon inapproprié un changement de trajectoire violent,

Vol 501 Ariane 5

Le 04 Juin 1996, le premier vol du lanceur Ariane 5 explose en vol après environ 40 secondes de vol.

- Le système inertiel envoie de mauvaises données au système de pilotage automatique,
- Le système inertiel de secours se met en route et réagit exactement de la même façon,
- Le pilote automatique, ayant de mauvaises données, ordonne de façon inapproprié un changement de trajectoire violent,
- Ce virage serré provoque un arrachage de pièces de la fusée,

Vol 501 Ariane 5

Le 04 Juin 1996, le premier vol du lanceur Ariane 5 explose en vol après environ 40 secondes de vol.

- Le système inertiel envoie de mauvaises données au système de pilotage automatique,
- Le système inertiel de secours se met en route et réagit exactement de la même façon,
- Le pilote automatique, ayant de mauvaises données, ordonne de façon inapproprié un changement de trajectoire violent,
- Ce virage serré provoque un arrachage de pièces de la fusée,
- La fusée étant alors hors de contrôle, le système d'autodestruction s'enclenche et la fusée explose (le navigateur au sol ayant lui aussi ordonné la destruction de la fusée).

Vol 501 Ariane 5

- L'origine du problème provient du dépassement mémoire (codage sur 8 bits) d'un entier, il en aurait fallut 9.
- Ce codage vient du code Ariane 4.
- Ariane 5 est plus puissance, d'où le dépassement.

Cependant [1]

1. pour plus d'informations
http://deschamp.free.fr/exinria/divers/ariane_501.html

Vol 501 Ariane 5

- L'origine du problème provient du dépassement mémoire (codage sur 8 bits) d'un entier, il en aurait fallut 9.
- Ce codage vient du code Ariane 4.
- Ariane 5 est plus puissance, d'où le dépassement.

Cependant [1]

- Copier-coller depuis Ariane 4 : la réutilisation de code est fréquente.
- Afin d'économiser environ 120 000 euros, des simulations n'ont pas été effectuées. Réalisées après coups, il a été montré qu'elles auraient permis de détecter le problème.
- Cahier des charges de pannes s'appuie sur des failles probabilistes. Il n'y a qu'un système de secours. Les problèmes de conception ne sont pas pris en compte.

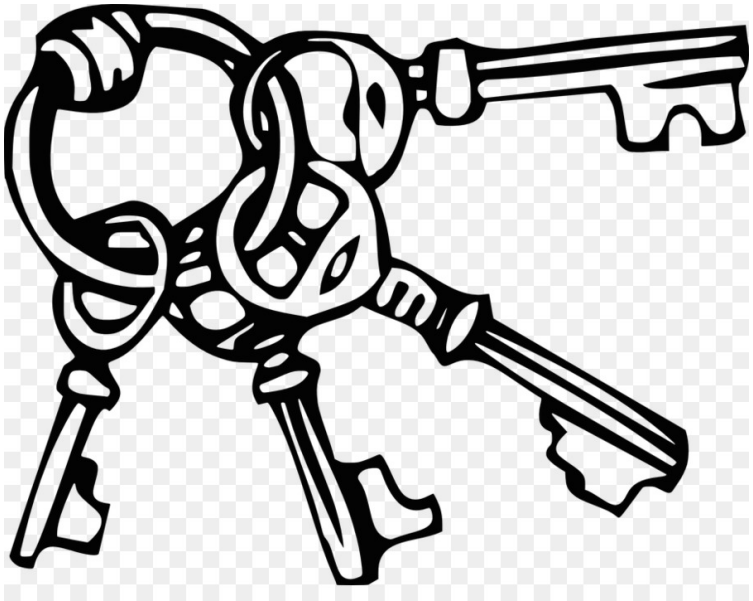
1. pour plus d'informations
http://deschamp.free.fr/exinria/divers/ariane_501.html

Mariner 1

Le 28 Juillet 1962 la fusée *Marnier 1* est détruite par le contrôle au sol au dessus de l'atlantique pour une mauvaise trajectoire : en cause une mauvaise transcription d'un caractère (il manquait un _).

Kerberos

Le système de gestion de clé Kerberos (système de sécurité) a une une faille de 1988 à 1996 générant des clés prédictibles. En cause, l'utilisation d'une mauvaise graine pour la génération pseudo-aléatoire.



Prix négatifs

- En avril 2021, en plein confinement COVID19, la demande en pétrole s'effondre.
- Les supertankers, pleins, ne savent pas où décharger. Les compagnie sont prête à payer pour se débarrasser de leur pétrole.
- Les prix du pétrole deviennent négatifs.

Prix négatifs

- En avril 2021, en plein confinement COVID19, la demande en pétrole s'effondre.
- Les supertankers, pleins, ne savent pas où décharger. Les compagnie sont prête à payer pour se débarrasser de leur pétrole.
- Les prix du pétrole deviennent négatifs.
- Le logiciel d'un société de Traders Sud-coréenne a planté, ne sachant gérer des prix négatifs : impossibilité de vendre pour les traders qui a entraîné de lourdes pertes.

Problèmes d'horloge

En 2004, le système de communication aérien de Los Angeles tombe en panne totale : plus aucune communication. Pas d'accident mais une dizaine d'avions sont passés proche.

Le problème : l'horloge Windows limité à 4 294 967 295 secondes (un peu moins de 50 jours). Les ordinateurs n'avaient pas été rebootés depuis cette durée.

Problèmes d'horloge

En 2004, le système de communication aérien de Los Angeles tombe en panne totale : plus aucune communication. Pas d'accident mais une dizaine d'avions sont passés proche.

Le problème : l'horloge Windows limité à 4 294 967 295 secondes (un peu moins de 50 jours). Les ordinateurs n'avaient pas été rebootés depuis cette durée.
En 2015, même erreur sur le Boing 787, qui doit être éteint au moins tous les 248 jours (heureusement pas de d'accident).

Autres problèmes logiciels

- Février 1991 : bug logiciel d'horodatage dans les batteries anti-missiles patriot : 28 morts.
- 2004 : logiciel de pension britannique plante : 1 milliard d'euros.
- 1988 : création involontaire d'un vers informatique (pour détecter des routes réseau). Coût 100 millions de dollars.
- Récemment : un F22 *reboot* (system out) lorsqu'il passe la ligne imaginaire de changement de jour dans le pacifique (division par zéro).
- ...

En général le problème n'est pas que logiciel, il vient d'une mauvaise organisation, d'un manque de spécification, d'un manque de tests, etc.

Plan

Schéma général

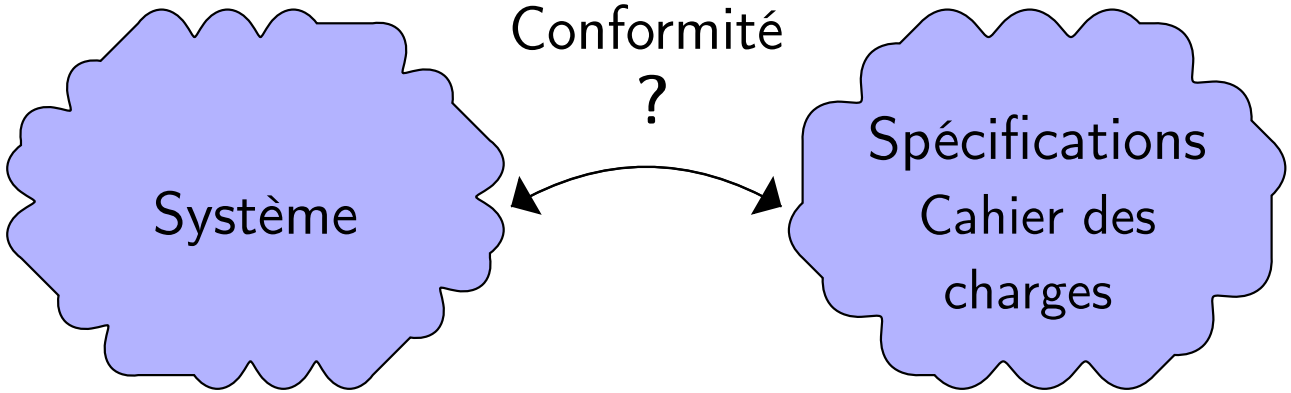


Schéma général

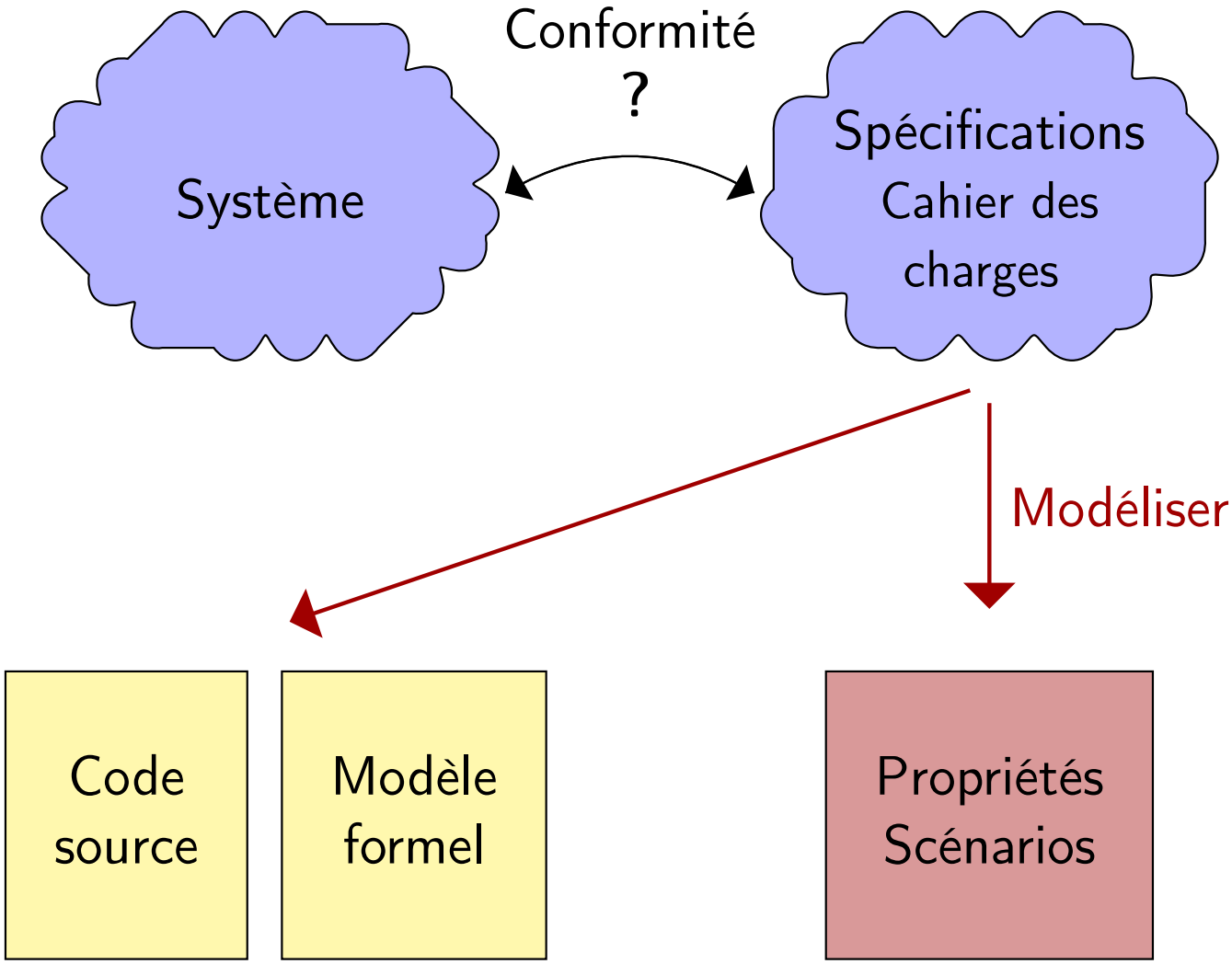
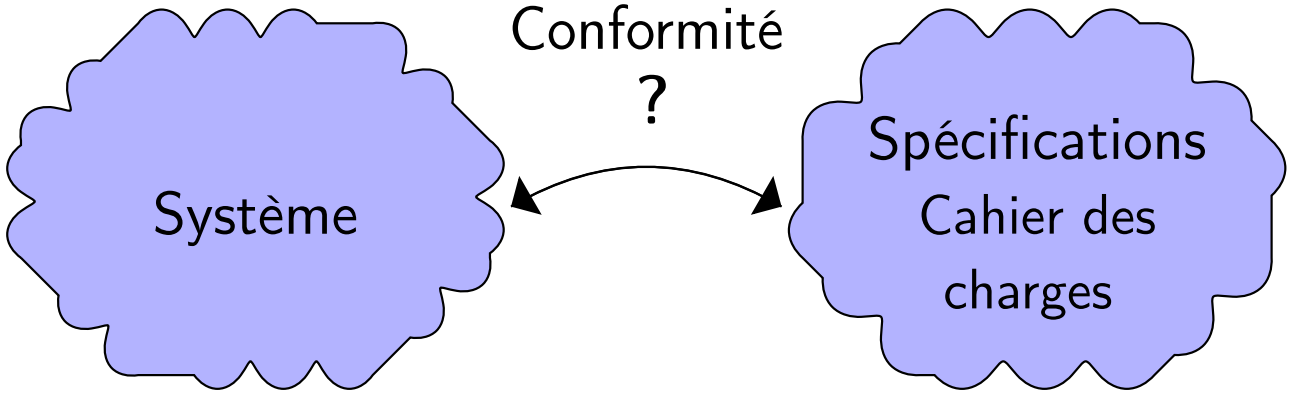


Schéma général



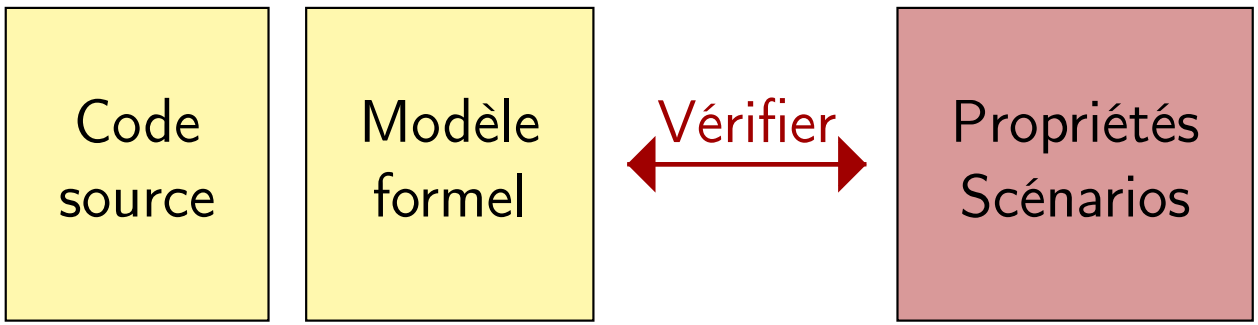
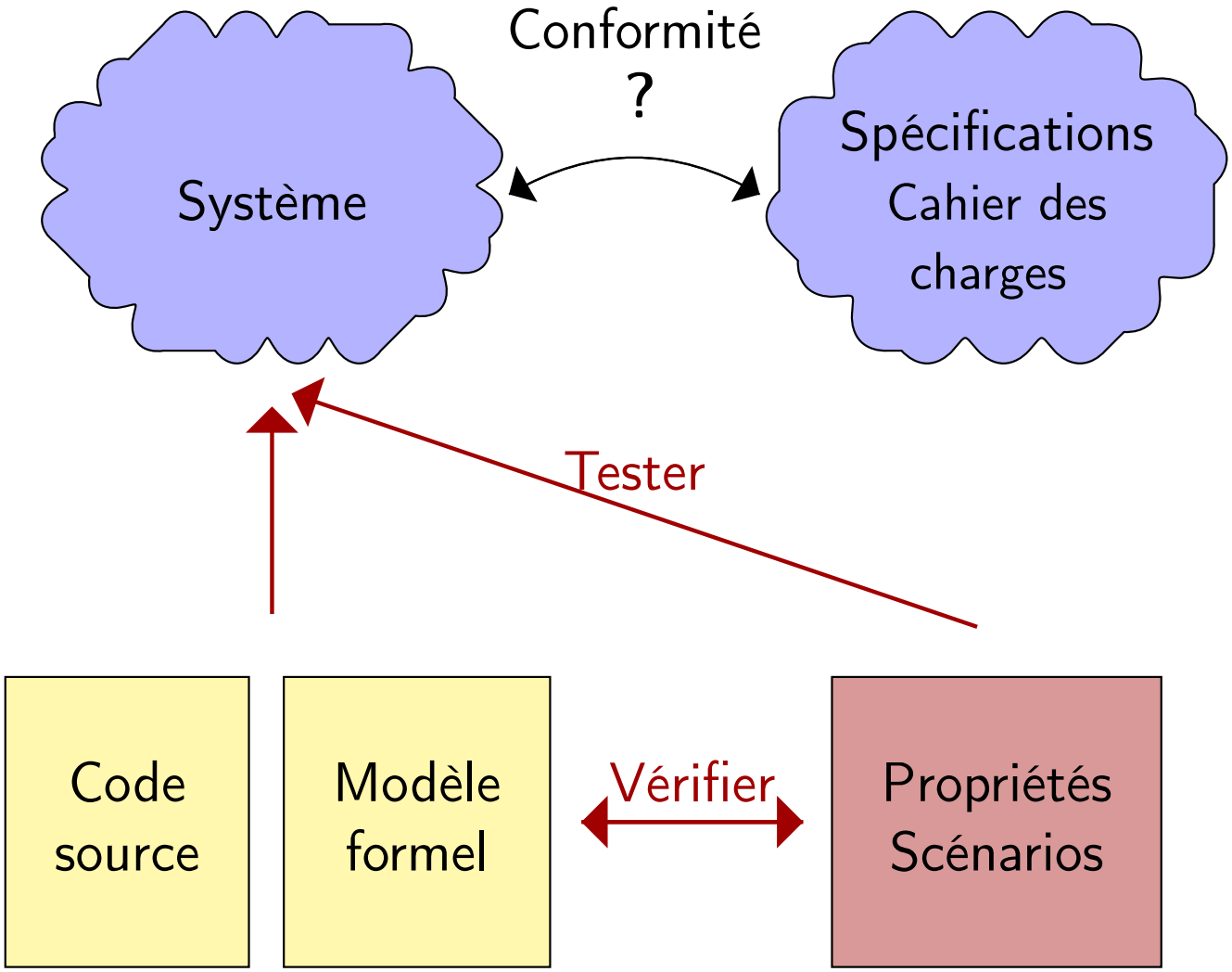


Schéma général



Approches automatiques

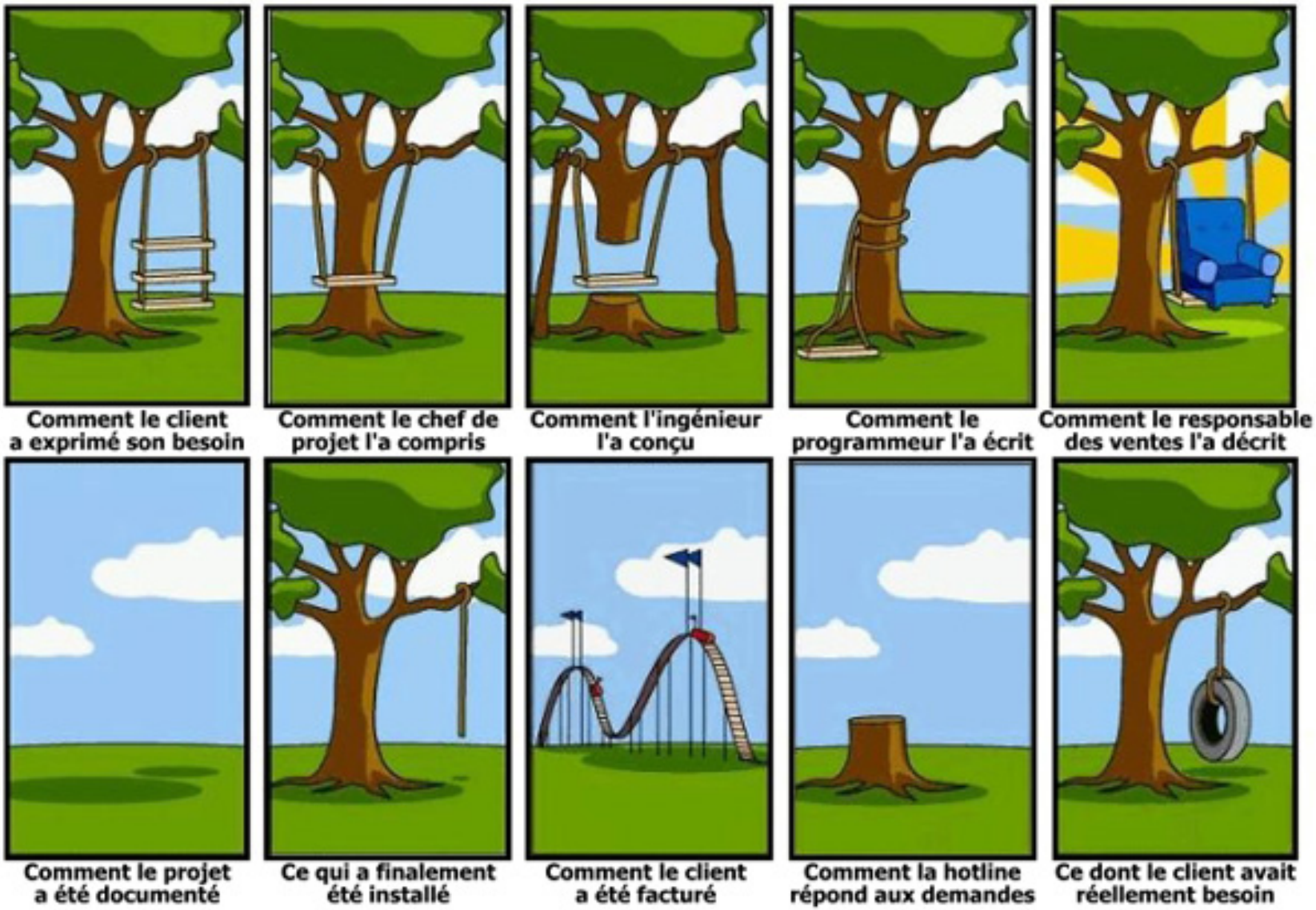
La vérification (preuve)

- Indécidabilité,
- Lourd à mettre en place (temps),
- Hautes qualifications,

La validation (test)

- Non exhaustif, sans garantie,
- Coûteux.

La conception logicielle



LTL et CTL, deux visions du temps pour modéliser les propriétés

Les propriétés sont souvent modélisés à l'aide de logiques temporelles

- LTL (temps linéaire), introduite par Pnueli,
- CTL (temps arborescent).

`image wikipedia`

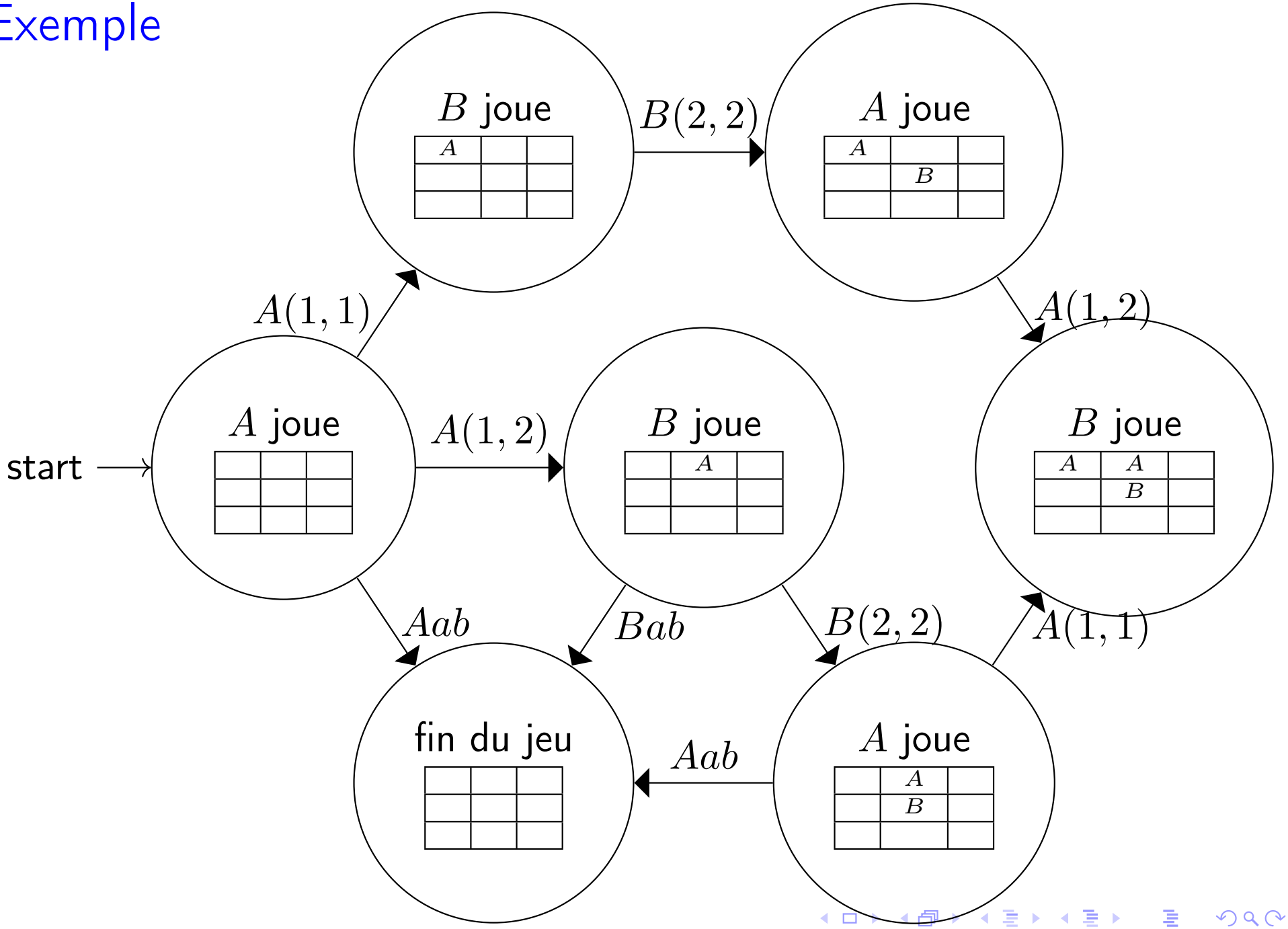
Modèles de systèmes à base de graphes

Utilisation de graphes

La plupart des modèles de systèmes ont pour base des graphes (en général finis), décorés sur les transitions (automates), les états (structures de Kripke) ou les deux (systèmes de transitions étiquetés).

- A un instant donné, les caractéristiques du système définissent un état, une configuration,
- Les évolutions du systèmes dépendent uniquement des caractéristiques de ce systèmes à un instant donné (qui peut contenir des informations sur le passé),
- Deux états sont *voisins* si par une transformation ou action élémentaires on passe de l'un à l'autre.

Exemple


B joue
A
B(2, 2)
A joue
A
B
A(1, 1)
A(1, 2)
start
A joue
A(1, 2)
B joue
A
B joue
A
A
B
Aab
Bab
B(2, 2)
A(1, 1)
fin du jeu
Aab
A joue
A
B

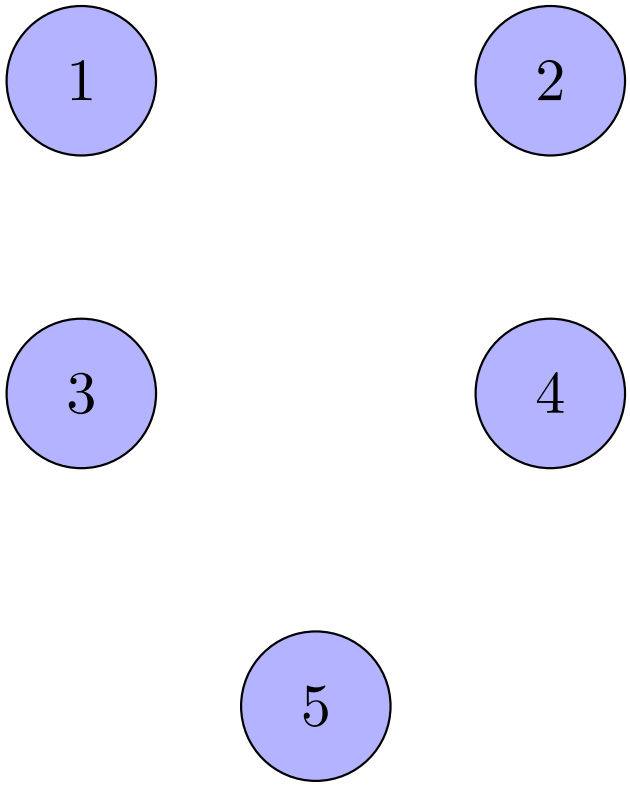
Exercices

Chercher les exercices 1 et 2 de la feuille de TD1.

Plan

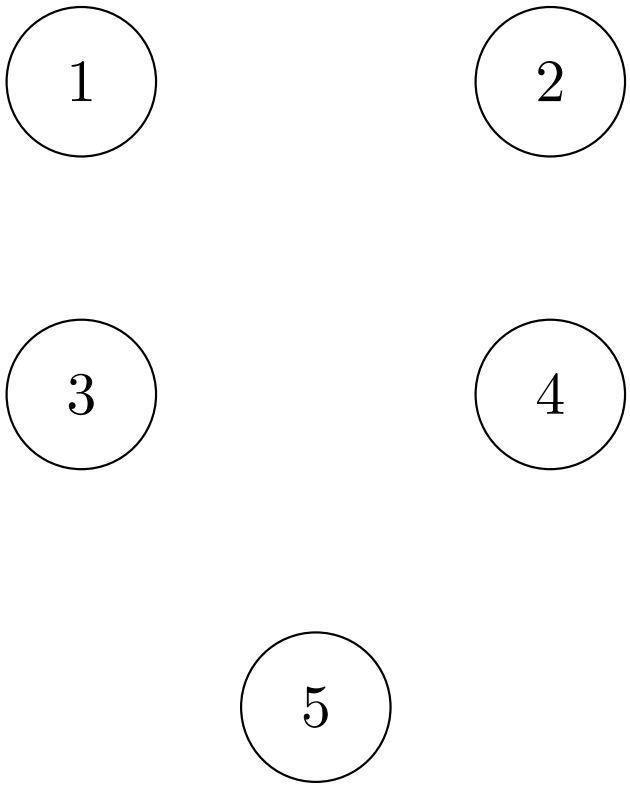
Automates finis

Automates finis



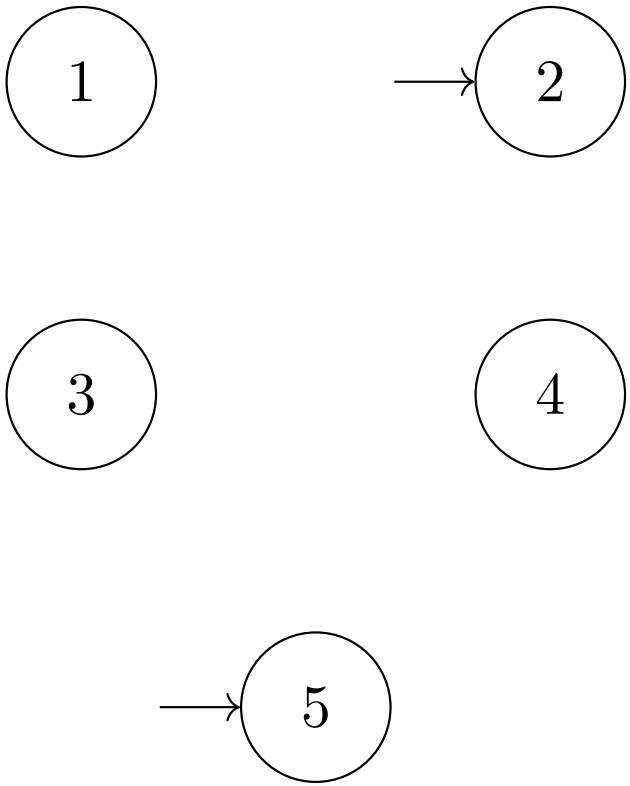
Ens. des états
$Q = \{1, 2, 3, 4, 5\}$

Automates finis



Ens. des états
$Q = \{1, 2, 3, 4, 5\}$

Automates finis



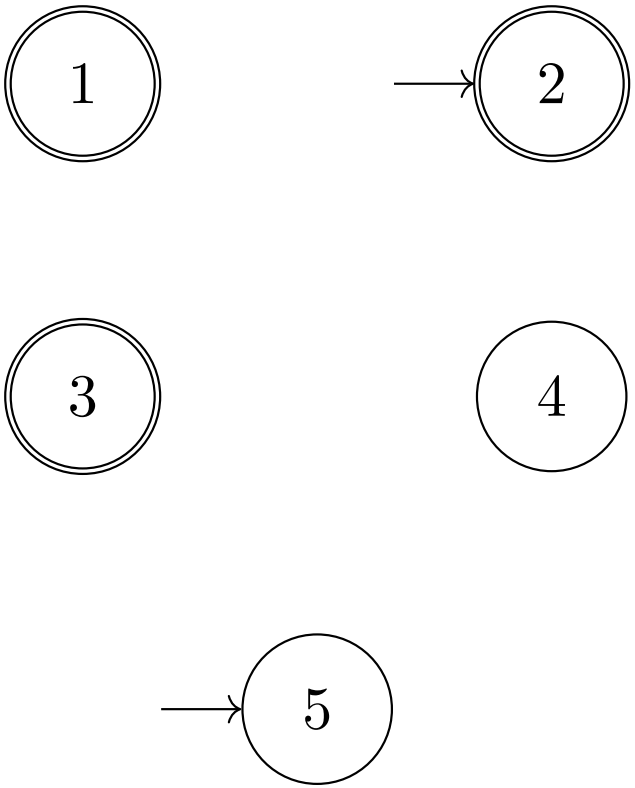
Ens. des états

$Q = \{1, 2, 3, 4, 5\}$

États initiaux

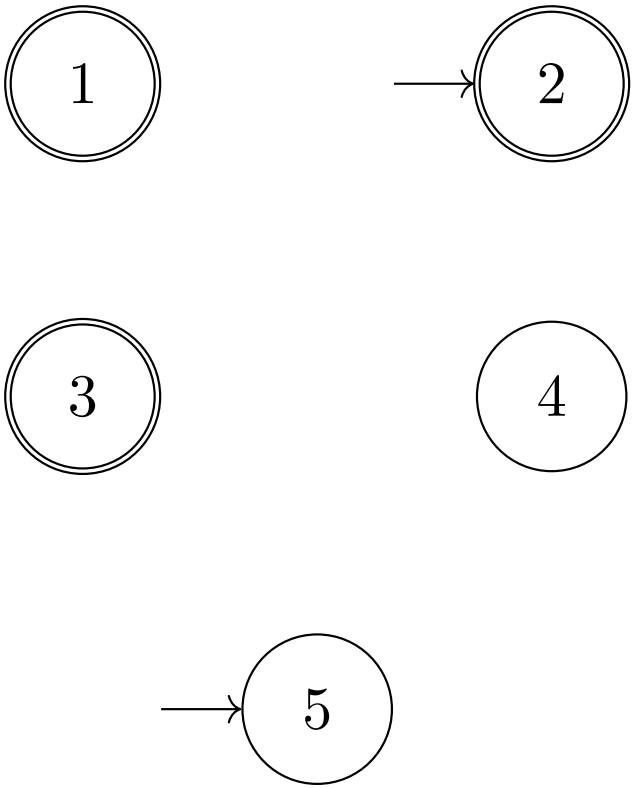
$I = \{2, 5\}$

Automates finis



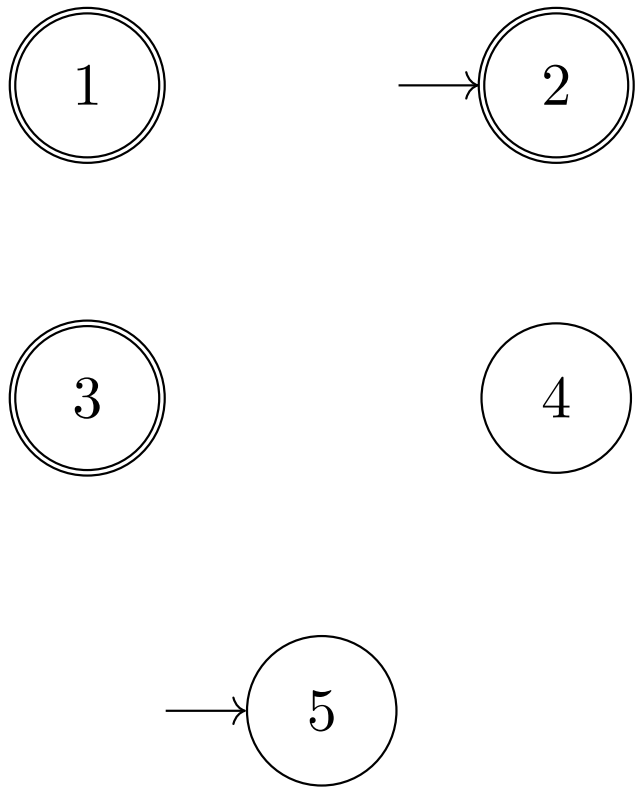
Ens. des états
$Q = \{1, 2, 3, 4, 5\}$
États initiaux
$I = \{2, 5\}$
États finaux
$F = \{1, 2, 3\}$

Automates finis



Ens. des états
$Q = \{1, 2, 3, 4, 5\}$
États initiaux
$I = \{2, 5\}$
États finaux
$F = \{1, 2, 3\}$

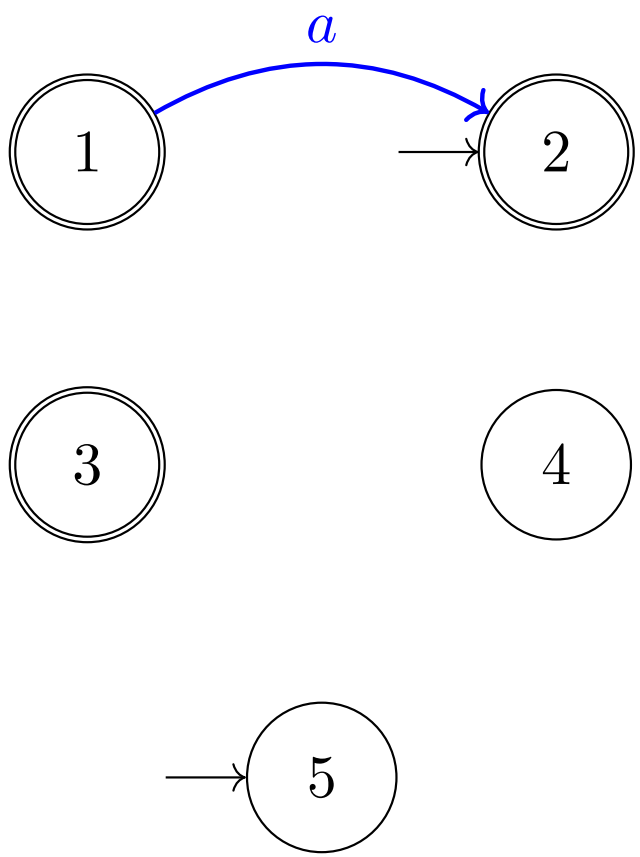
Automates finis



Ens. des états
$Q = \{1, 2, 3, 4, 5\}$
États initiaux
$I = \{2, 5\}$
États finaux
$F = \{1, 2, 3\}$

Ens. des transitions
$\Delta = \{$
$(1, a, 2), (1, a, 3), (1, b, 2),$
$(2, b, 4), (4, a, 4),$
$(5, a, 4), (5, b, 3)\}$

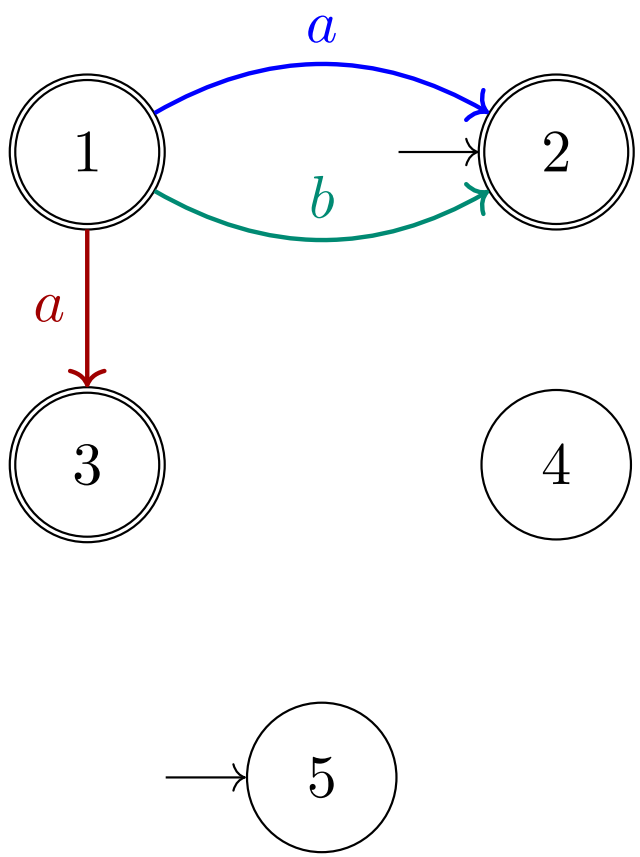
Automates finis



Ens. des états
$Q = \{1, 2, 3, 4, 5\}$
États initiaux
$I = \{2, 5\}$
États finaux
$F = \{1, 2, 3\}$

Ens. des transitions
$\Delta = \{$
$(1, a, 2), (1, a, 3), (1, b, 2),$
$(2, b, 4), (4, a, 4),$
$(5, a, 4), (5, b, 3)\}$

Automates finis



Ens. des états

$Q = \{1, 2, 3, 4, 5\}$

États initiaux

$I = \{2, 5\}$

États finaux

$F = \{1, 2, 3\}$

Ens. des transitions

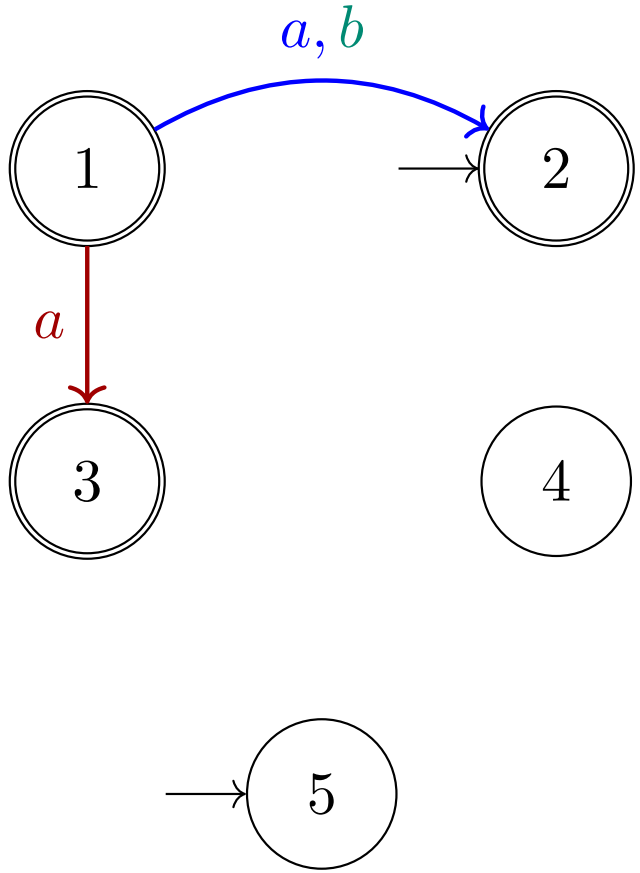
$\Delta = \{$

$(1, a, 2), (1, a, 3), (1, b, 2),$

$(2, b, 4), (4, a, 4),$

$(5, a, 4), (5, b, 3)\}$

Automates finis



Ens. des états

$Q = \{1, 2, 3, 4, 5\}$

États initiaux

$I = \{2, 5\}$

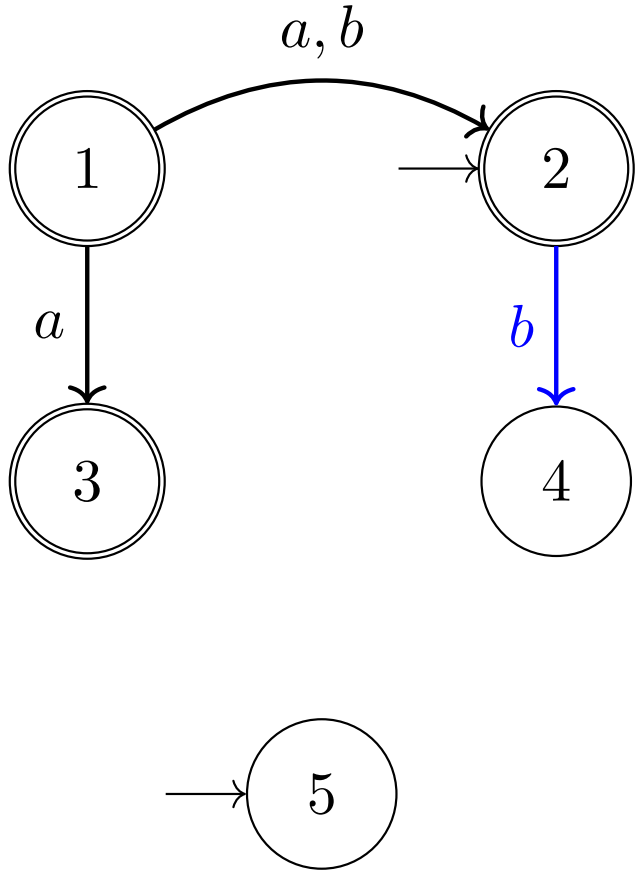
États finaux

$F = \{1, 2, 3\}$

Ens. des transitions

$\Delta = \{$
$(1, a, 2), (1, a, 3), (1, b, 2),$
$(2, b, 4), (4, a, 4),$
$(5, a, 4), (5, b, 3)\}$

Automates finis



Ens. des états

$Q = \{1, 2, 3, 4, 5\}$

États initiaux

$I = \{2, 5\}$

États finaux

$F = \{1, 2, 3\}$

Ens. des transitions

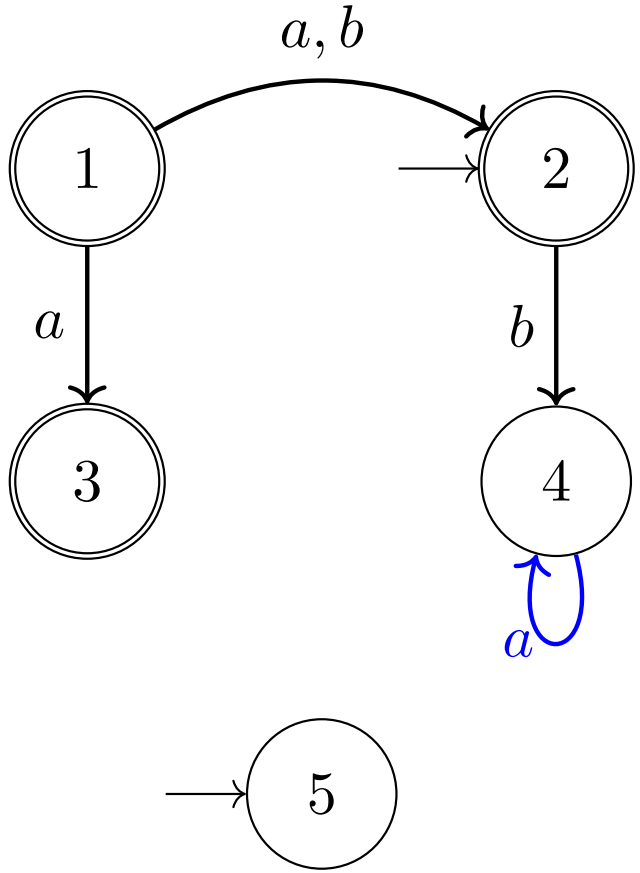
$\Delta = \{$

$(1, a, 2), (1, a, 3), (1, b, 2),$

$(2, b, 4), (4, a, 4),$

$(5, a, 4), (5, b, 3)\}$

Automates finis



Ens. des états

$Q = \{1, 2, 3, 4, 5\}$

États initiaux

$I = \{2, 5\}$

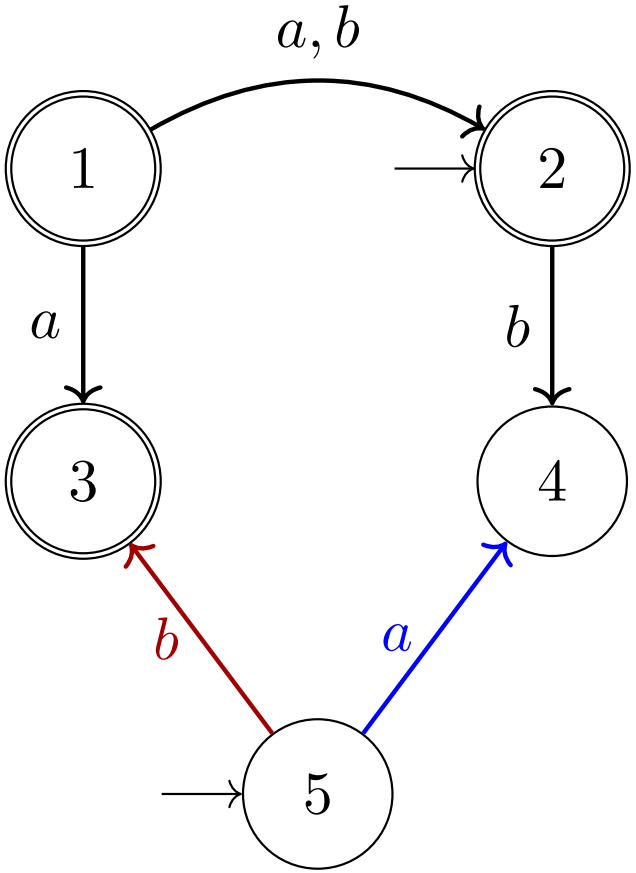
États finaux

$F = \{1, 2, 3\}$

Ens. des transitions

$\Delta = \{$
$(1, a, 2), (1, a, 3), (1, b, 2),$
$(2, b, 4), (4, a, 4),$
$(5, a, 4), (5, b, 3)\}$

Automates finis



Ens. des états

$Q = \{1, 2, 3, 4, 5\}$

États initiaux

$I = \{2, 5\}$

États finaux

$F = \{1, 2, 3\}$

Ens. des transitions

$\Delta = \{$
$(1, a, 2), (1, a, 3), (1, b, 2),$
$(2, b, 4), (4, a, 4),$
$(5, a, 4), (5, b, 3)\}$

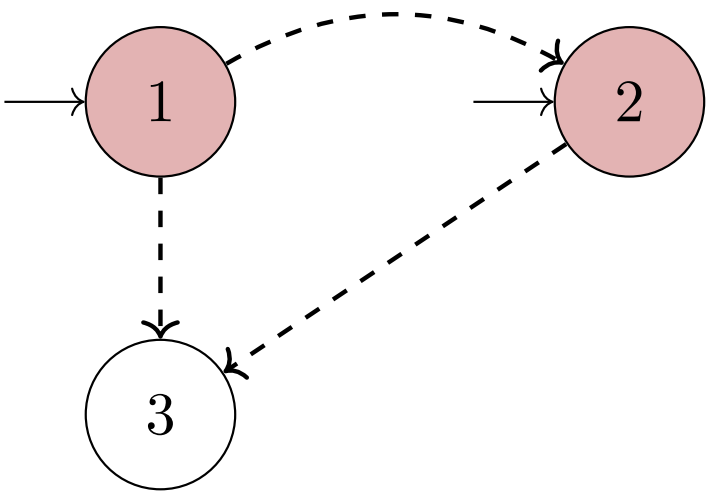
Automates finis déterministes

Un automate fini est déterministe s'il vérifie les deux conditions suivantes :

Automates finis déterministes

Un automate fini est déterministe s'il vérifie les deux conditions suivantes :

- Il possède exactement un état initial,

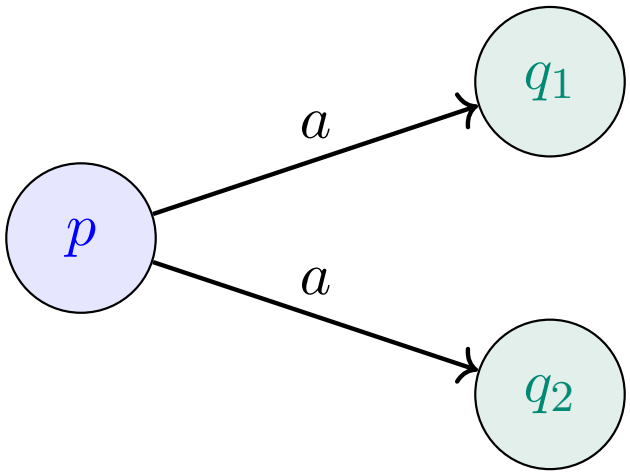


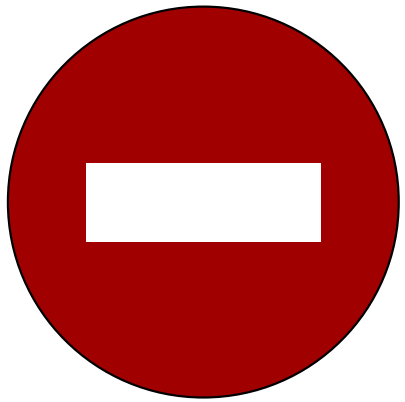
Il y a deux états initiaux 1 et 2, l'automate **ne** peut **pas** être déterministe.

Automates finis déterministes

Un automate fini est déterministe s'il vérifie les deux conditions suivantes :

- Il possède exactement un état initial,
- Pour tout état p, toute lettre a, il existe au plus un état q tel que (p, a, q) soit une transition.





Définitions

Un chemin dans un automate fini est une suite finie

$$(p_1, a_1, p_2), (p_2, a_2, p_3), \ldots, (p_k, a_k, p_{k+1})$$

de transitions consécutives.

Définitions

Un chemin dans un automate fini est une suite finie

$$(p_1, a_1, p_2), (p_2, a_2, p_3), \ldots, (p_k, a_k, p_{k+1})$$

de transitions consécutives.
L'entier k est la longueur du chemin.

Définitions

Un chemin dans un automate fini est une suite finie

$$(p_1, a_1, p_2), (p_2, a_2, p_3), \ldots, (p_k, a_k, p_{k+1})$$

de transitions consécutives.
L'entier k est la longueur du chemin.
Le mot $a_1 \ldots a_k$ est l'étiquette du chemin.

.

Définitions

Un chemin dans un automate fini est une suite finie

$$(p_1, a_1, p_2), (p_2, a_2, p_3), \ldots, (p_k, a_k, p_{k+1})$$

de transitions consécutives.
L'entier k est la longueur du chemin.
Le mot $a_1 \ldots a_k$ est l'étiquette du chemin.
Si p_1 est initial et p_{k+1} est final, le chemin est réussi ou acceptant.

.

Définitions

Un chemin dans un automate fini est une suite finie

$$(p_1, a_1, p_2), (p_2, a_2, p_3), \ldots, (p_k, a_k, p_{k+1})$$

de transitions consécutives.
L'entier k est la longueur du chemin.
Le mot $a_1 \ldots a_k$ est l'étiquette du chemin.
Si p_1 est initial et p_{k+1} est final, le chemin est réussi ou acceptant.

Un mot est reconnu ou accepté s'il est l'étiquette d'un chemin réussi.

Définitions

Un chemin dans un automate fini est une suite finie

$$(p_1, a_1, p_2), (p_2, a_2, p_3), \ldots, (p_k, a_k, p_{k+1})$$

de transitions consécutives.
L'entier k est la longueur du chemin.
Le mot $a_1 \ldots a_k$ est l'étiquette du chemin.
Si p_1 est initial et p_{k+1} est final, le chemin est réussi ou acceptant.

Un mot est reconnu ou accepté s'il est l'étiquette d'un chemin réussi.
L'ensemble des mots reconnus est appelé le langage reconnu ou accepté par l'automate.

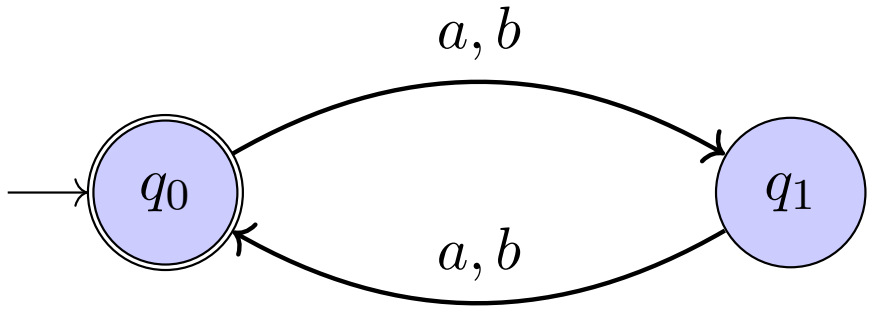
Exemples, sur l'alphabet $\{a, b\}$

- Ensemble des mots de longueur paire

- Ensemble des mots finissant par a (deux solutions)

Exemples, sur l'alphabet $\{a, b\}$

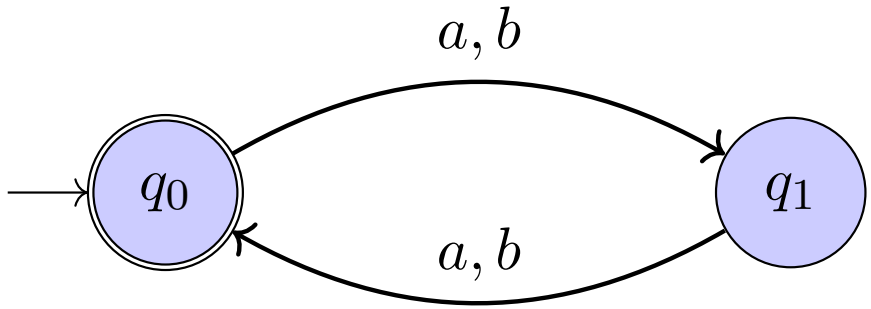
- Ensemble des mots de longueur paire



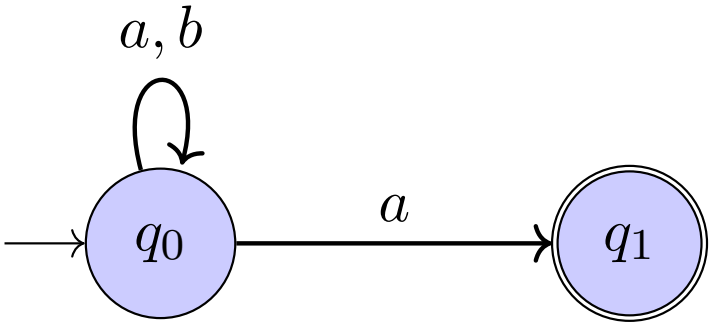
- Ensemble des mots finissant par a (deux solutions)

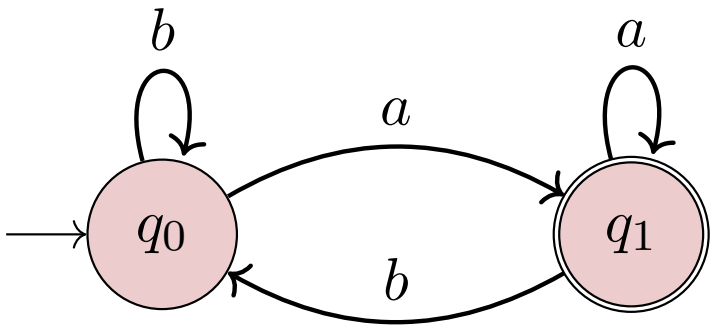
Exemples, sur l'alphabet $\{a, b\}$

- Ensemble des mots de longueur paire



- Ensemble des mots finissant par a (deux solutions)





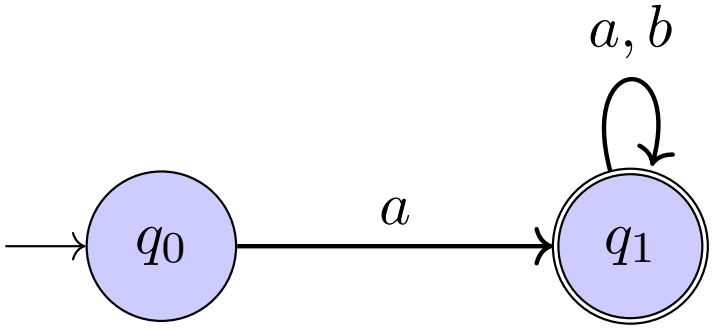
Exemples

- Ensemble des mots commençant par a

- Ensemble des mots commençant et finissant par a

Exemples

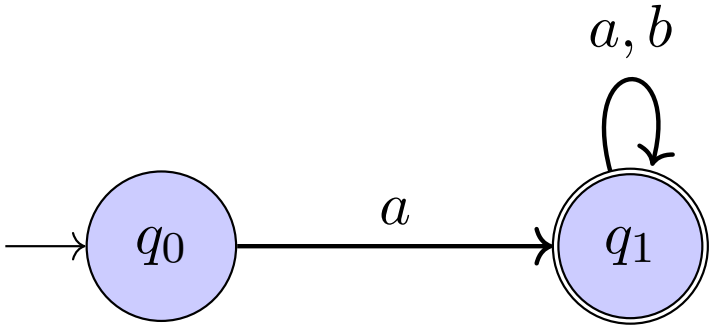
- Ensemble des mots commençant par a



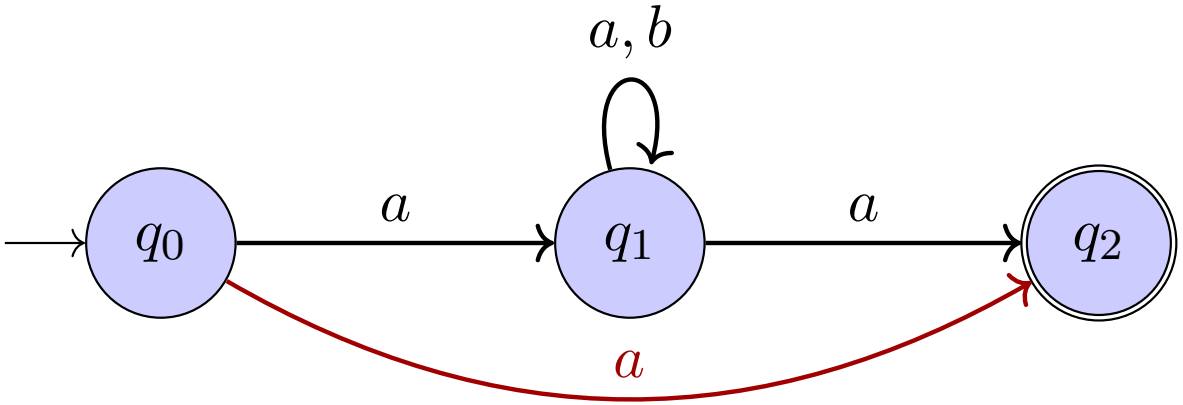
- Ensemble des mots commençant et finissant par a

Exemples

- Ensemble des mots commençant par a



- Ensemble des mots commençant et finissant par a



Exercices

Chercher les exercices 3 à 6 de la feuille de TD1.

Langages réguliers

Définition

Un langage est régulier s'il peut s'exprimer par une expression régulière (expression utilisant uniquement des langages finis, l'union, le produit et l'étoile).

Théorème

Soit L un langage de Σ^*. Il y a équivalence entre :

(1) L est reconnaissable par un automate fini,

(2) L est reconnaissable par un automate fini déterministe,

(3) L est régulier.

Propriétés

La classe des langages sur Σ reconnaissables par un automate fini est close par union, produit, étoile, intersection et complément.

Problématique

Soit $\mathcal{A} = (Q, \Sigma, E, I, F)$ un automate fini. On cherche à construire un automate $\mathcal{B}$ déterministe tel que $L(\mathcal{A}) = L(\mathcal{B})$.

Problématique

Soit $\mathcal{A} = (Q, \Sigma, E, I, F)$ un automate fini. On cherche à construire un automate $\mathcal{B}$ déterministe tel que $L(\mathcal{A}) = L(\mathcal{B})$.

- Construction des états de $\mathcal{B}$,

Problématique

Soit $\mathcal{A} = (Q, \Sigma, E, I, F)$ un automate fini. On cherche à construire un automate $\mathcal{B}$ déterministe tel que $L(\mathcal{A}) = L(\mathcal{B})$.

- Construction des états de $\mathcal{B}$,
- État initial,

Problématique

Soit $\mathcal{A} = (Q, \Sigma, E, I, F)$ un automate fini. On cherche à construire un automate $\mathcal{B}$ déterministe tel que $L(\mathcal{A}) = L(\mathcal{B})$.

- Construction des états de $\mathcal{B}$,
- État initial,
- États finaux,

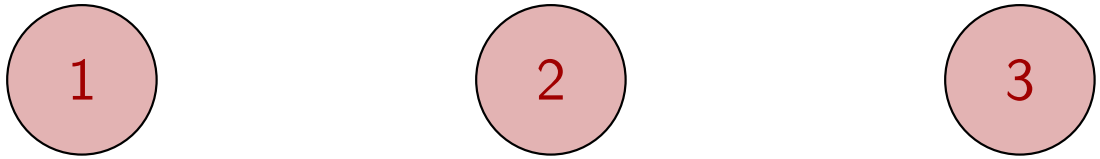
Problématique

Soit $\mathcal{A} = (Q, \Sigma, E, I, F)$ un automate fini. On cherche à construire un automate $\mathcal{B}$ déterministe tel que $L(\mathcal{A}) = L(\mathcal{B})$.

- Construction des états de $\mathcal{B}$,
- État initial,
- États finaux,
- Transitions.

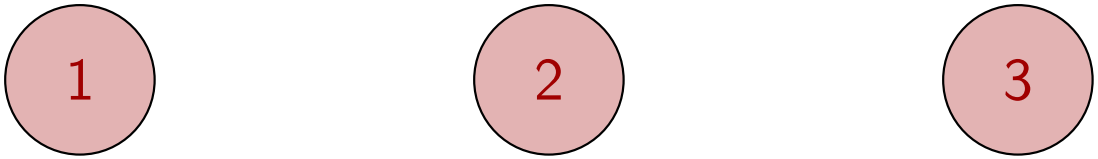
Les états

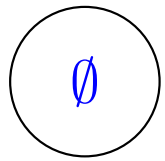
Les états de $\mathcal{B}$ sont les parties de Q (noté souvent $\mathcal{P}(Q)$ ou 2^Q).
Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, I, F)$



Les états

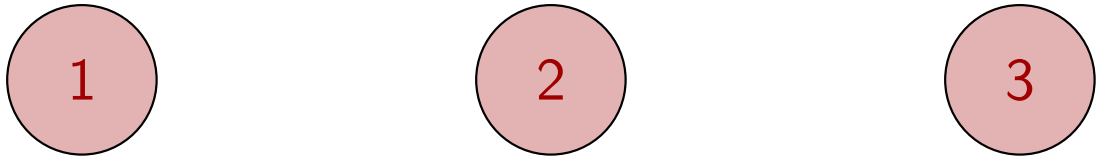
Les états de $\mathcal{B}$ sont les parties de Q (noté souvent $\mathcal{P}(Q)$ ou 2^Q).
Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, I, F)$





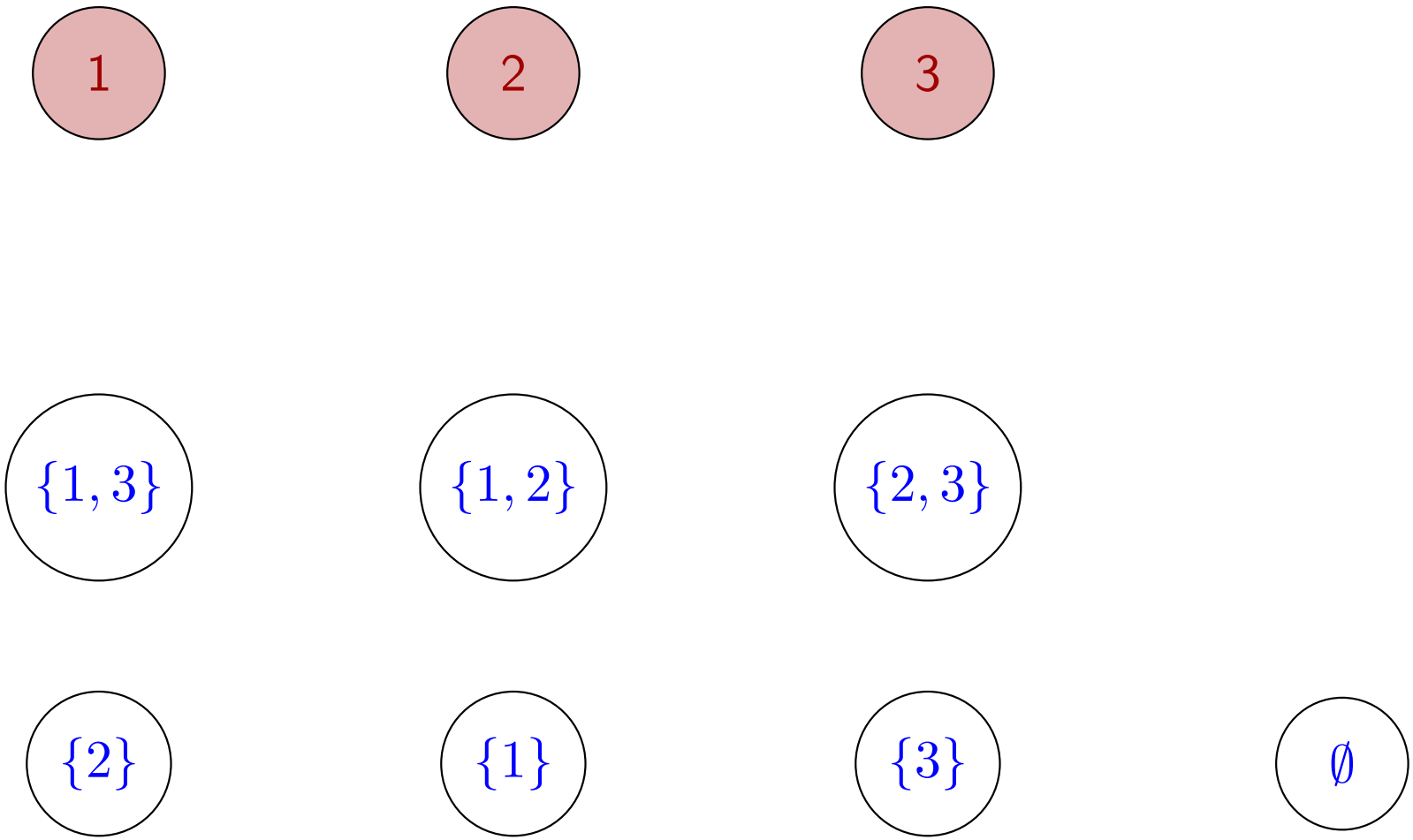
Les états

Les états de $\mathcal{B}$ sont les parties de Q (noté souvent $\mathcal{P}(Q)$ ou 2^Q).
Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, I, F)$



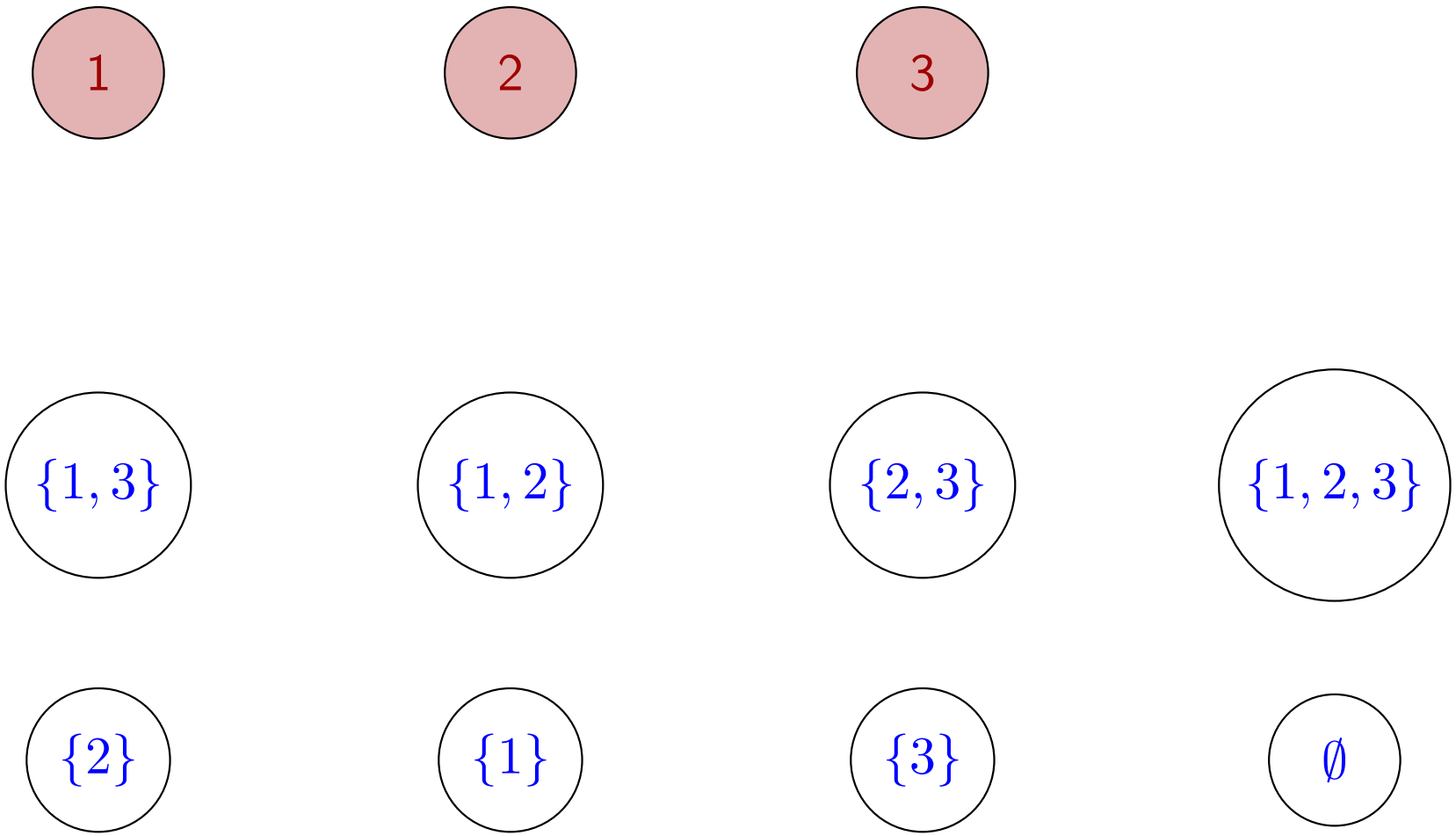
Les états

Les états de $\mathcal{B}$ sont les parties de Q (noté souvent $\mathcal{P}(Q)$ ou 2^Q).
Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, I, F)$



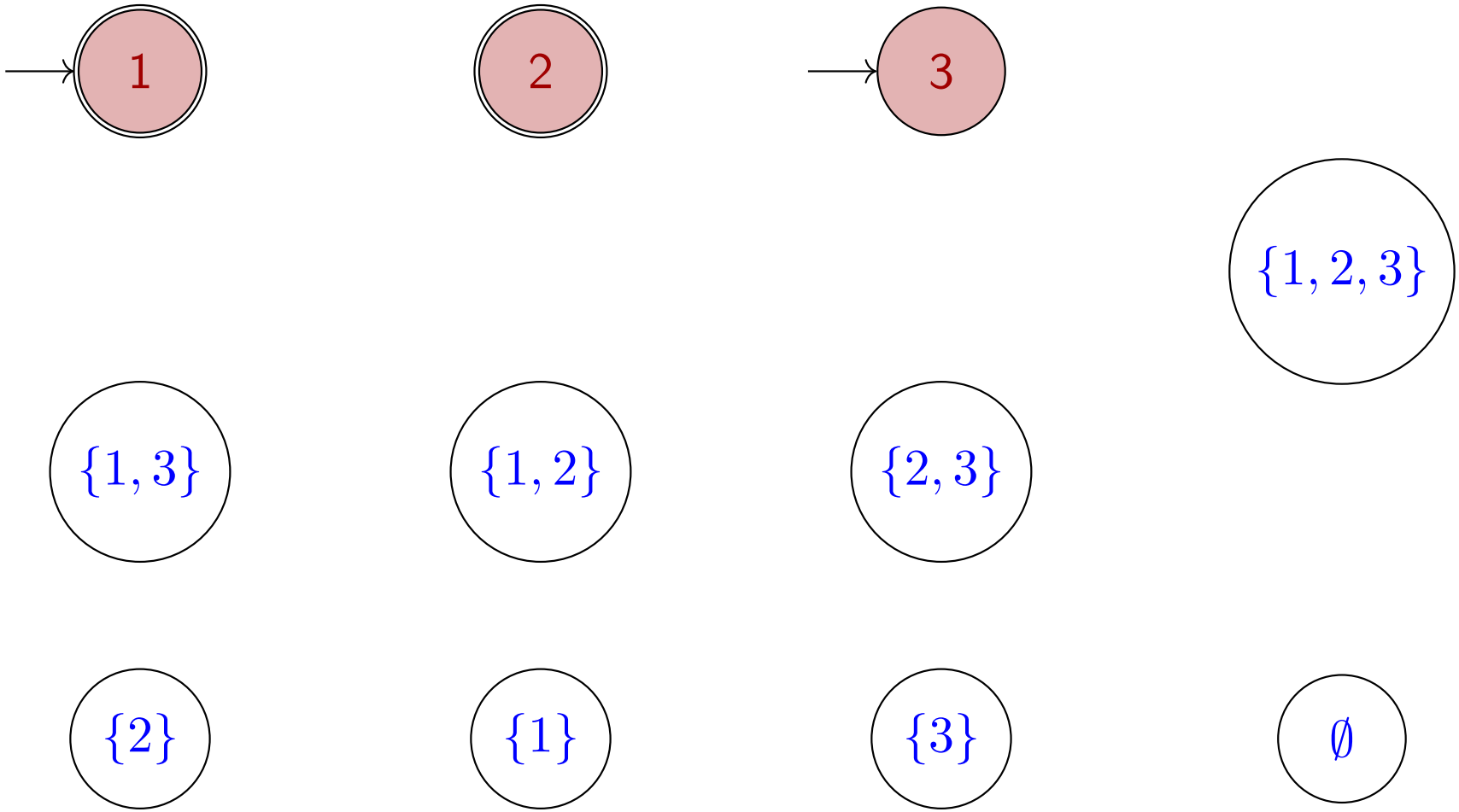
Les états

Les états de $\mathcal{B}$ sont les parties de Q (noté souvent $\mathcal{P}(Q)$ ou 2^Q).
Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, I, F)$



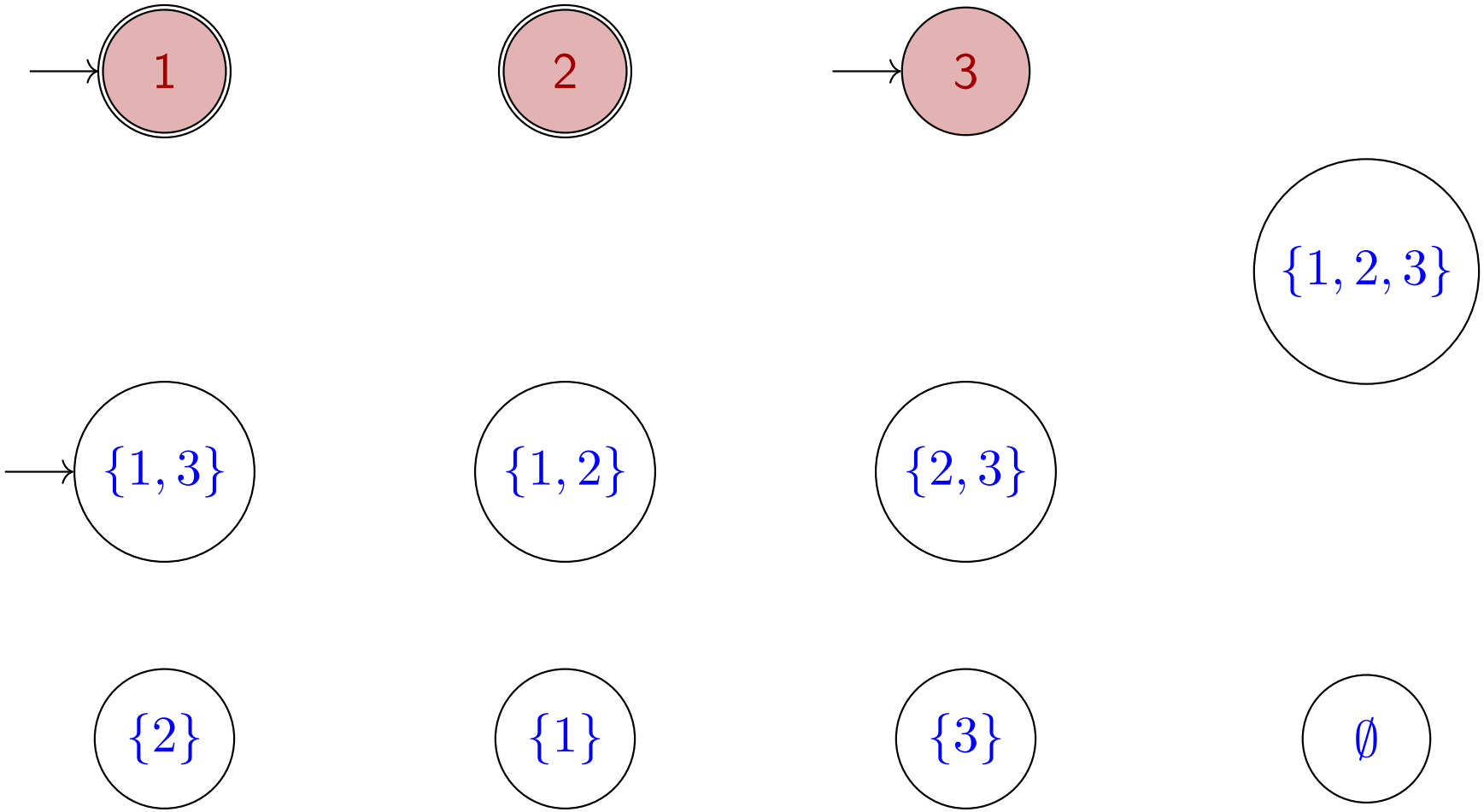
Les états initiaux – finaux

Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, \{1,3\}, \{1,2\})$
L'état initial de $\mathcal{B}$ est celui constitué de tous les états initiaux de $\mathcal{A}$.



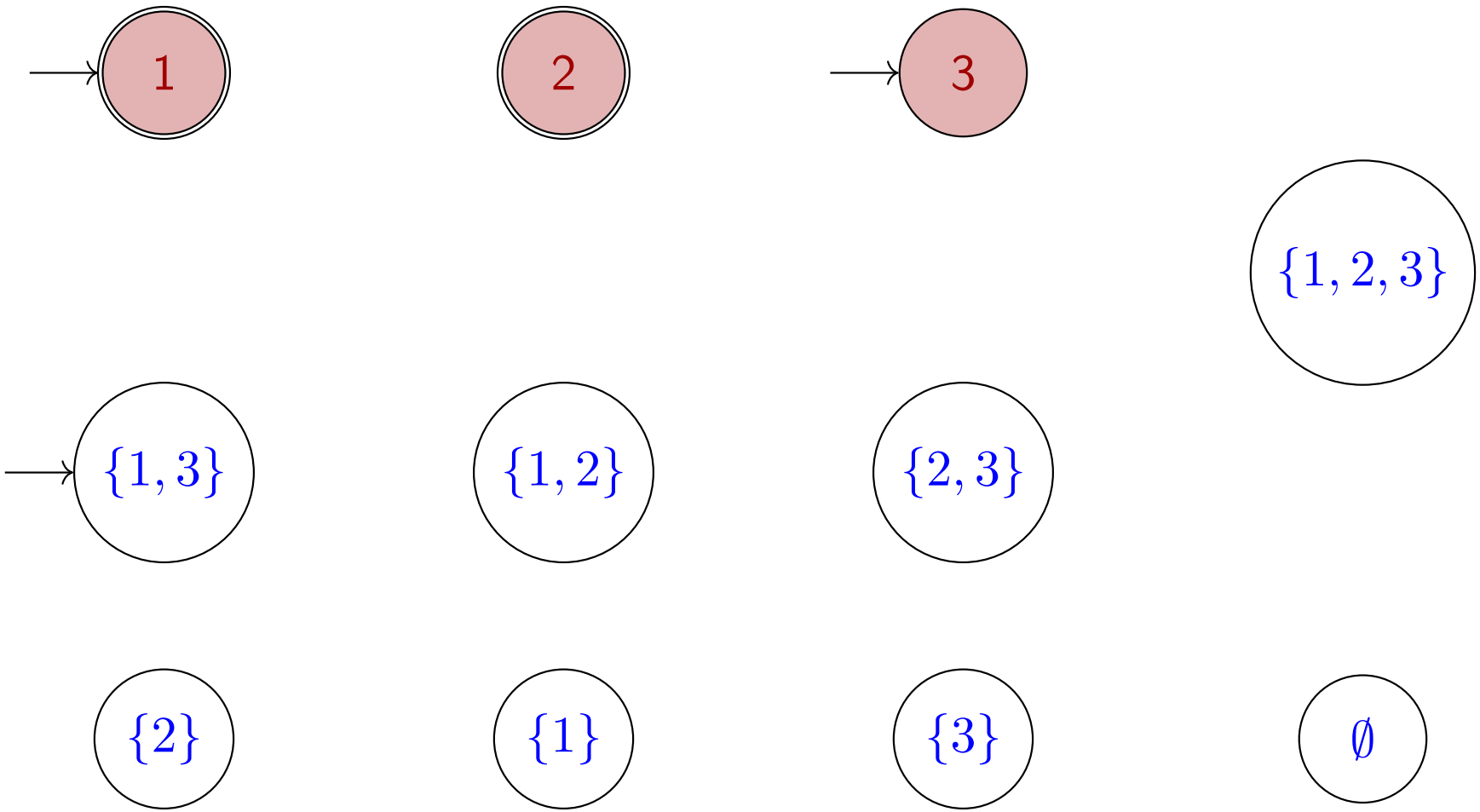
Les états initiaux – finaux

Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, \{1,3\}, \{1,2\})$
L'état initial de $\mathcal{B}$ est celui constitué de tous les états initiaux de $\mathcal{A}$.



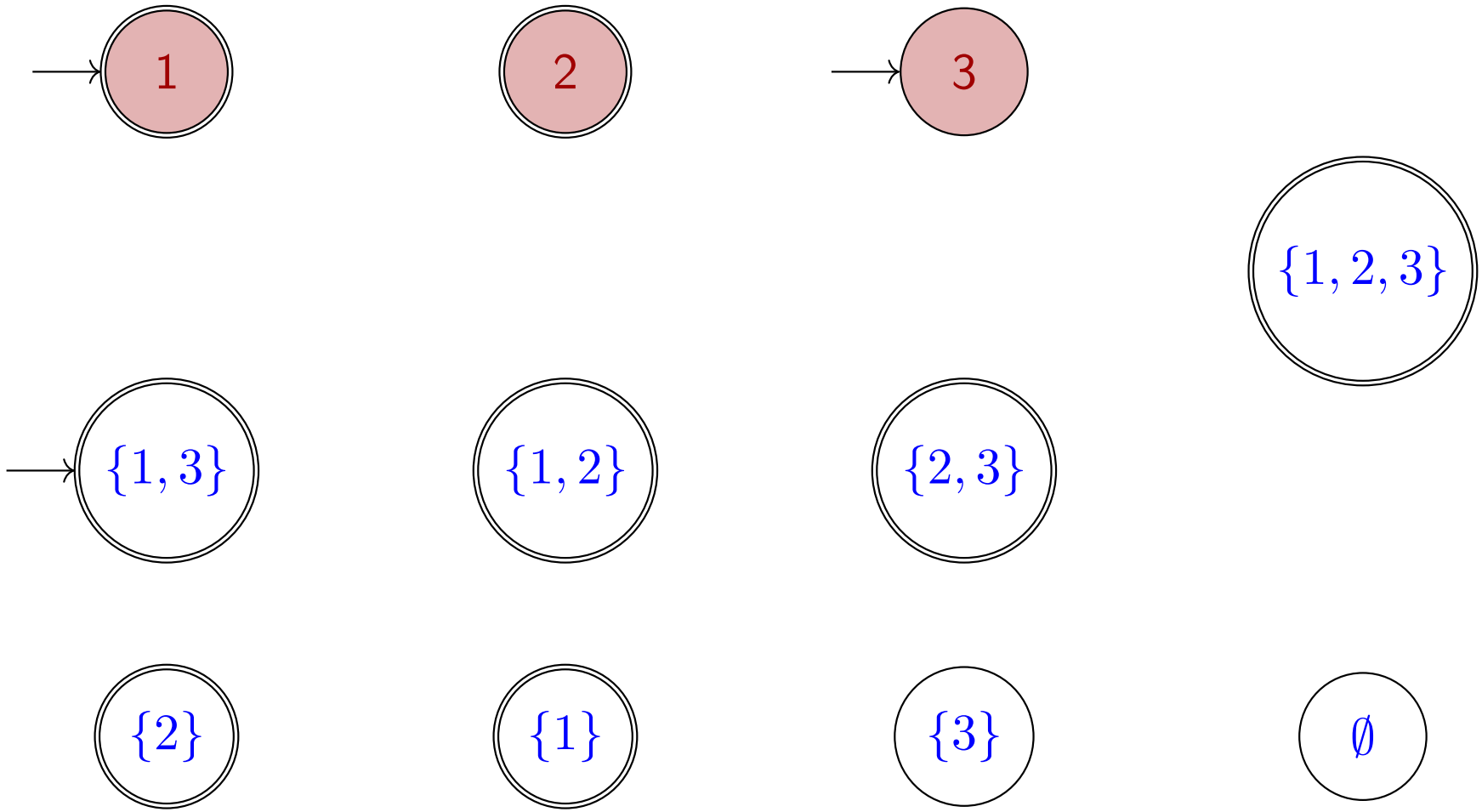
Les états initiaux – finaux

Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, \{1,3\}, \{1,2\})$
Sont finaux dans $\mathcal{B}$ tous les états qui contiennent un état final de $\mathcal{A}$.

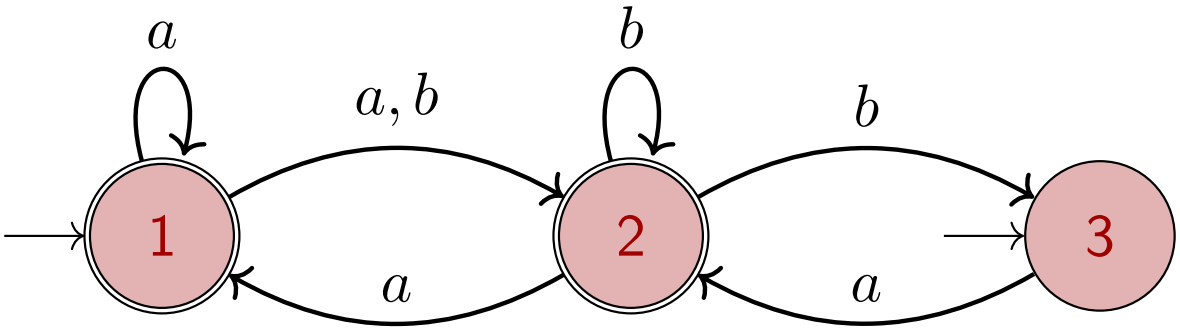


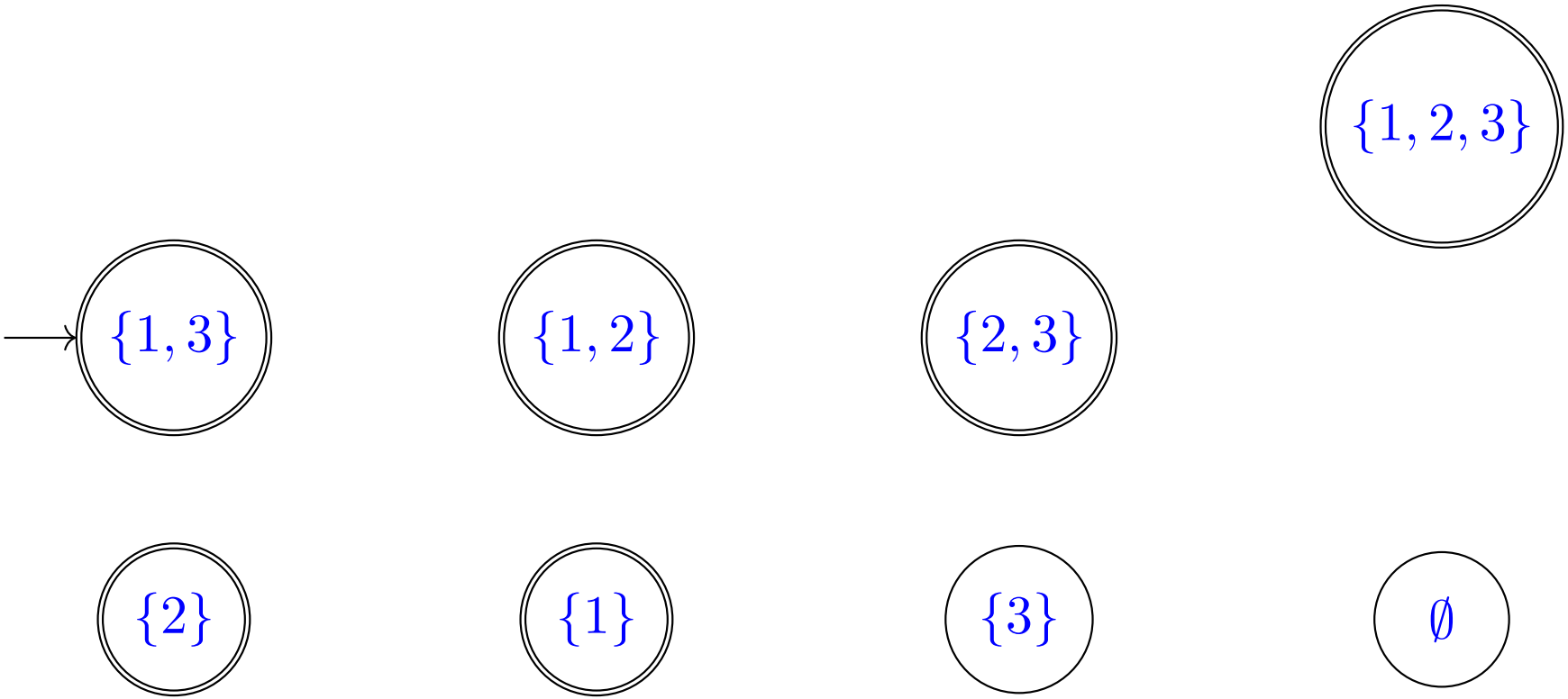
Les états initiaux – finaux

Supposons dans l'exemple que $\mathcal{A} = (\{1,2,3\}, \{a,b\}, E, \{1,3\}, \{1,2\})$
Sont finaux dans $\mathcal{B}$ tous les états qui contiennent un état final de $\mathcal{A}$.

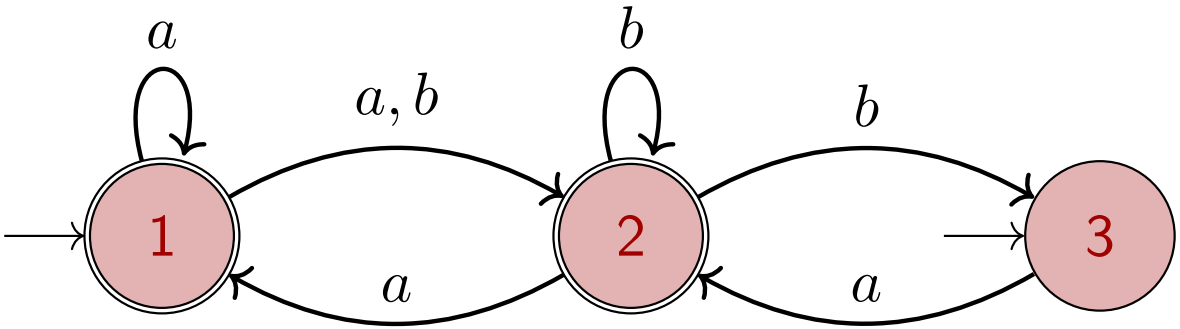


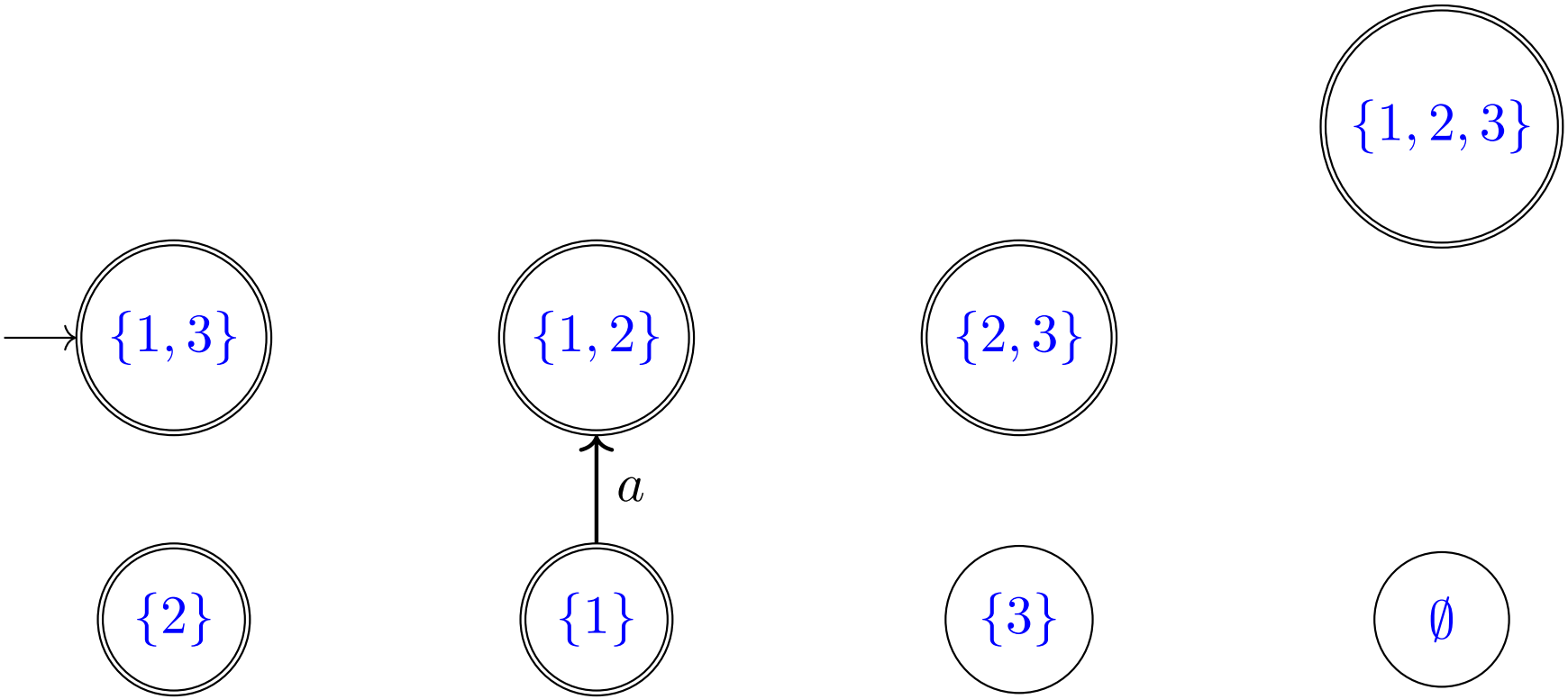
Les transitions



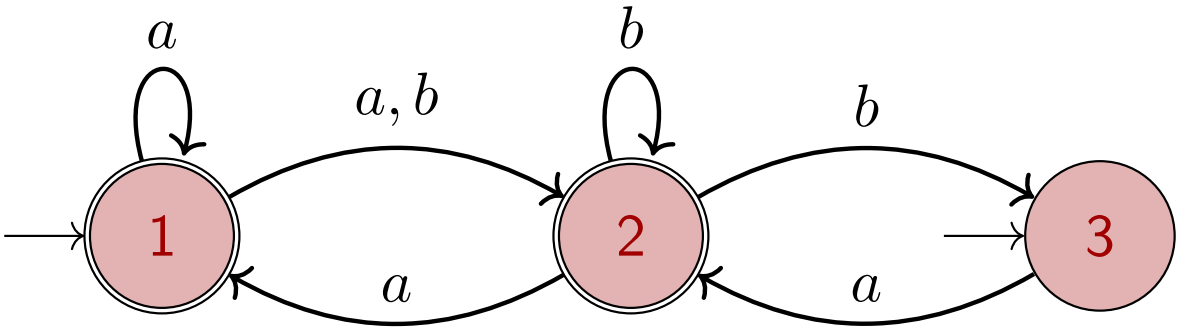


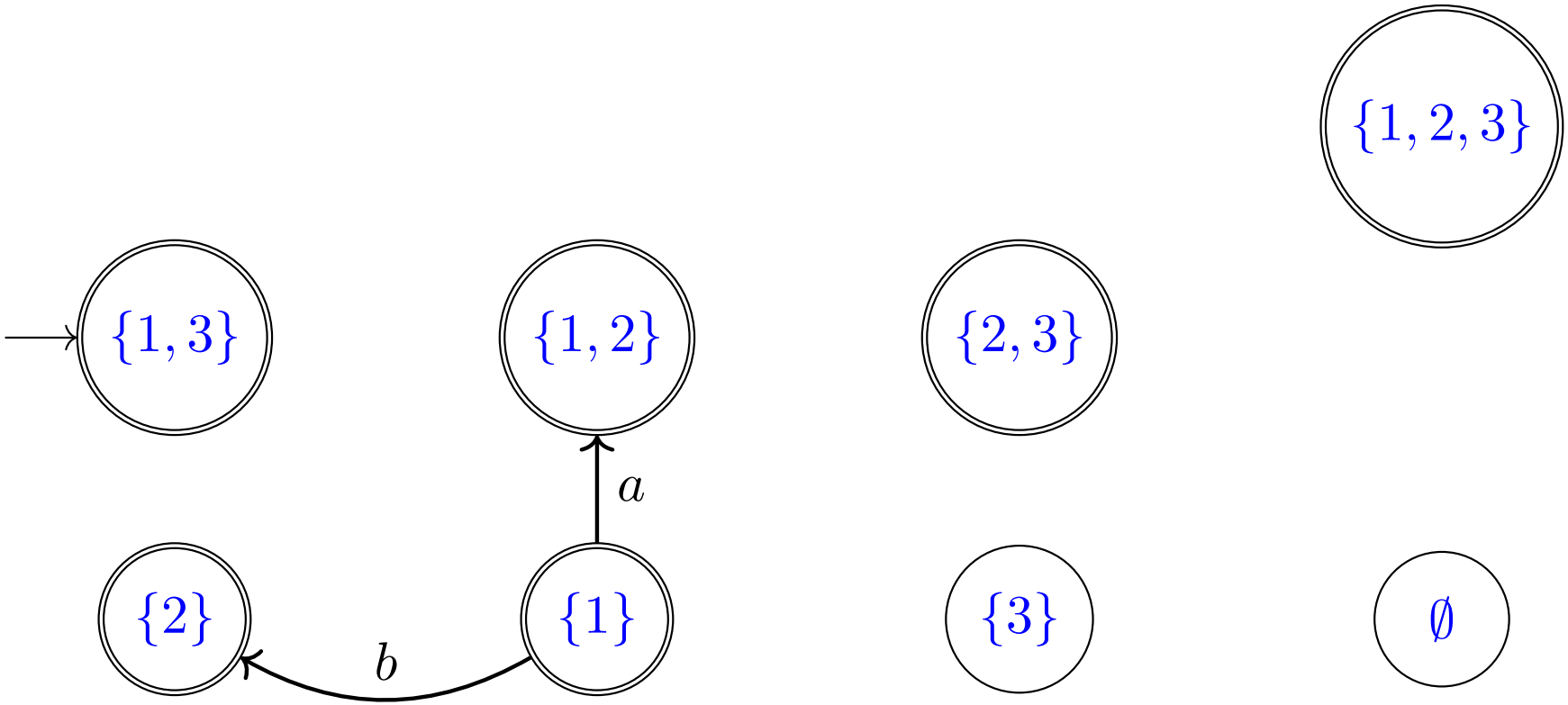
Les transitions



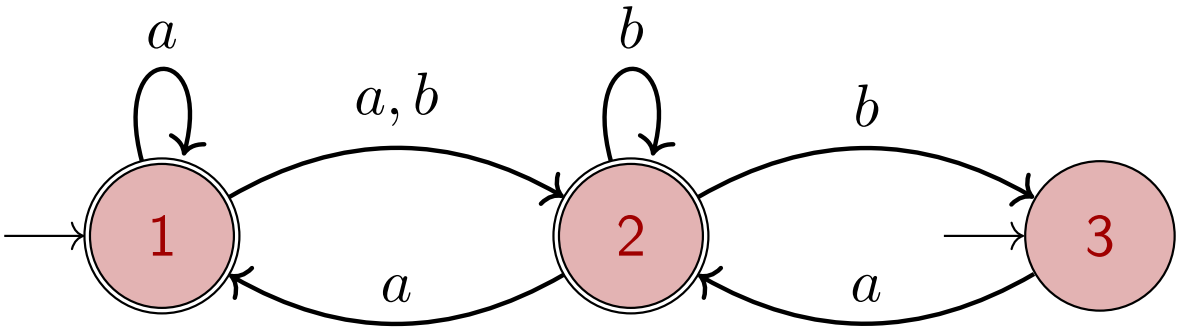


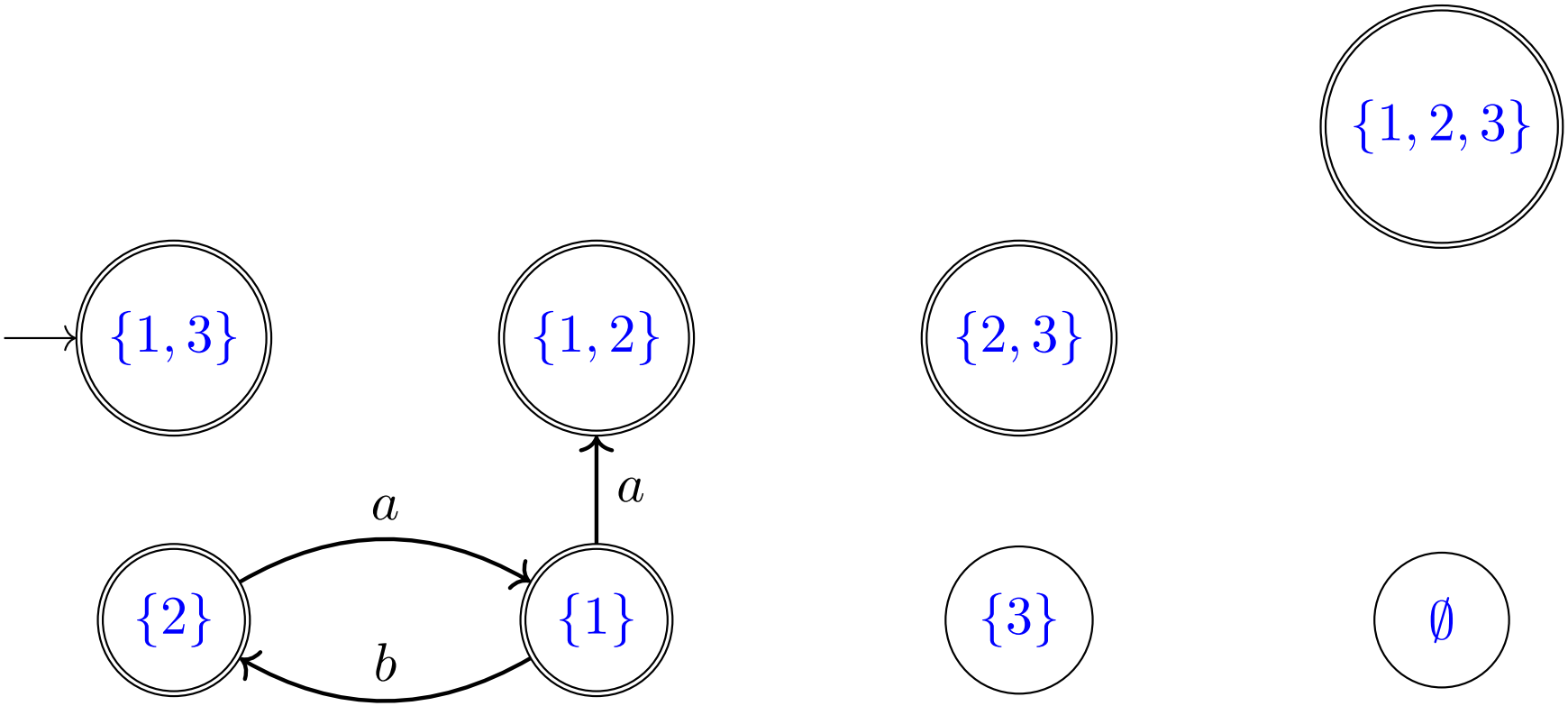
Les transitions



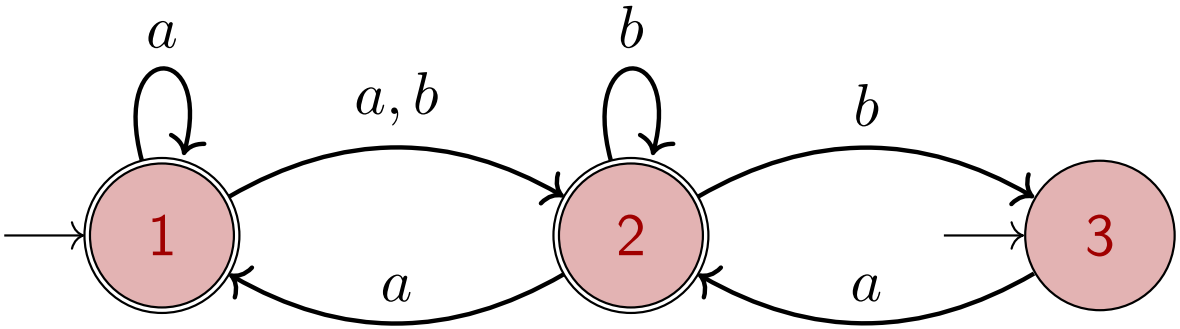


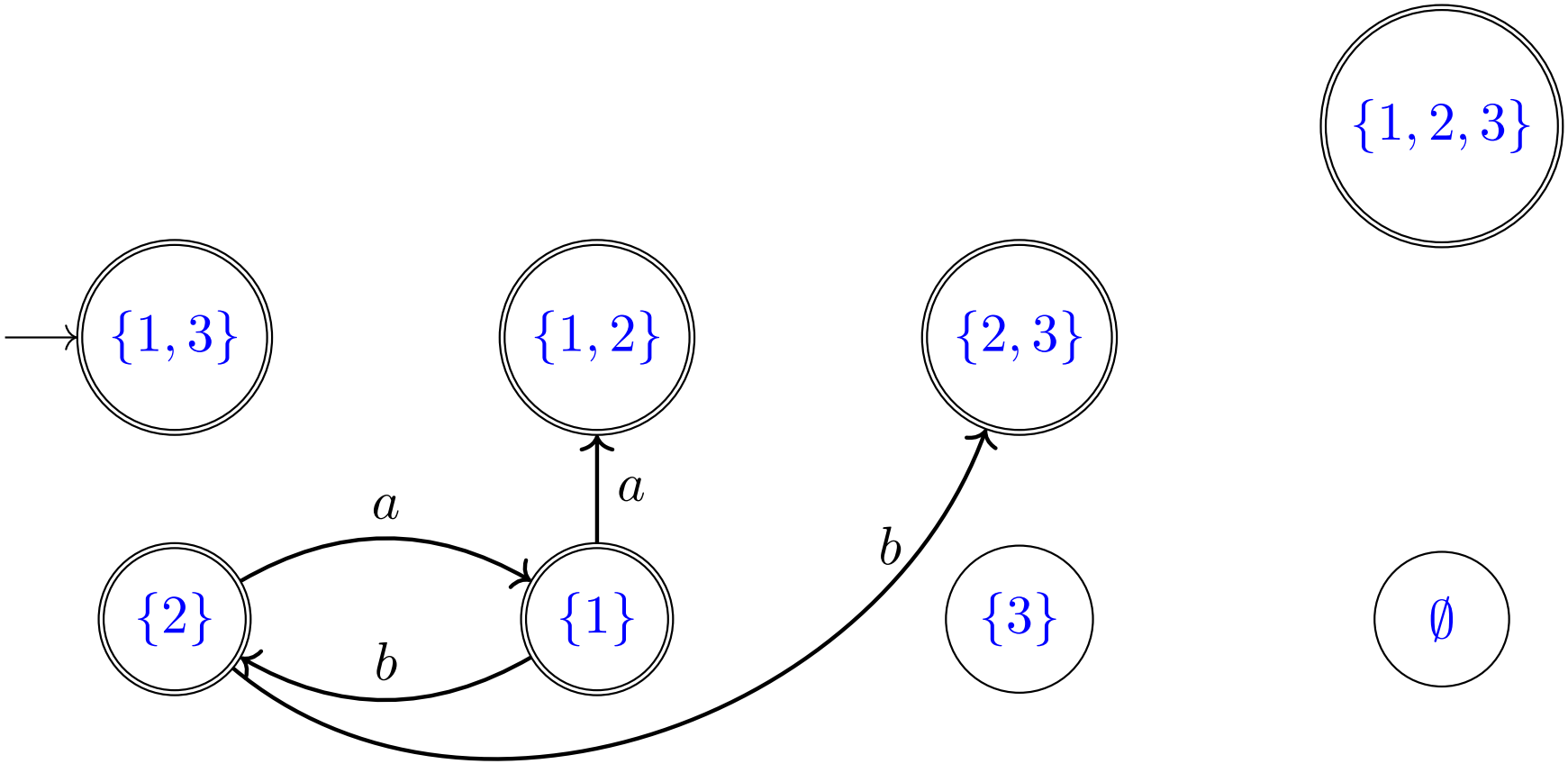
Les transitions



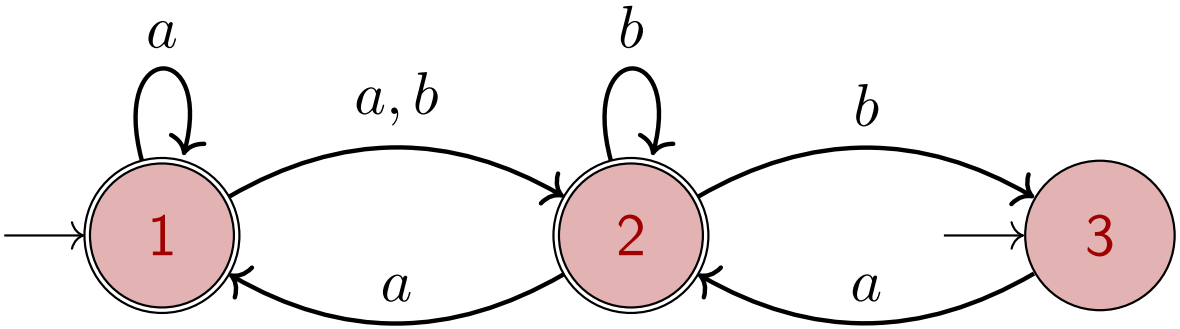


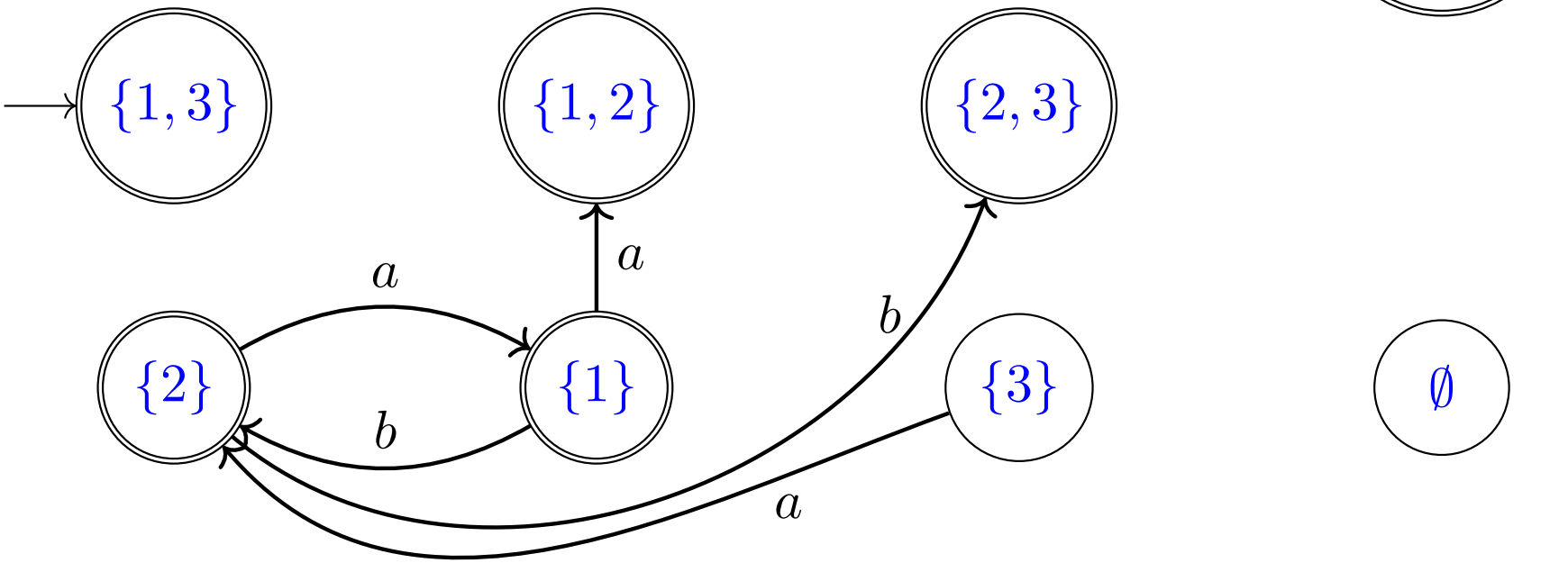
Les transitions



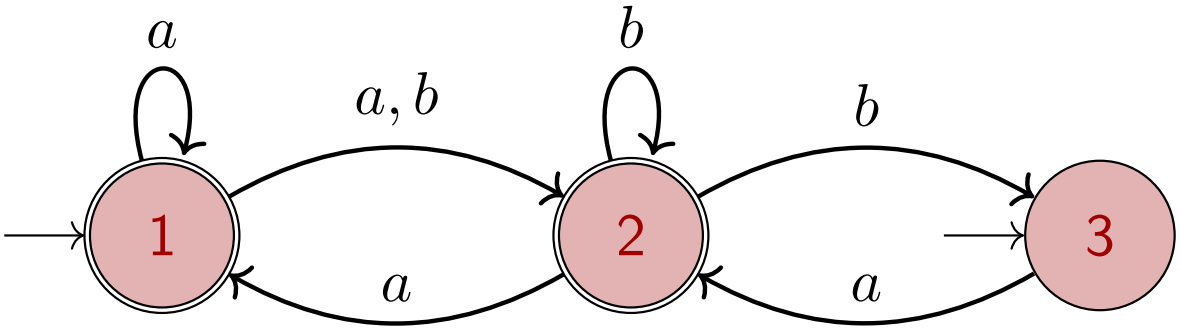


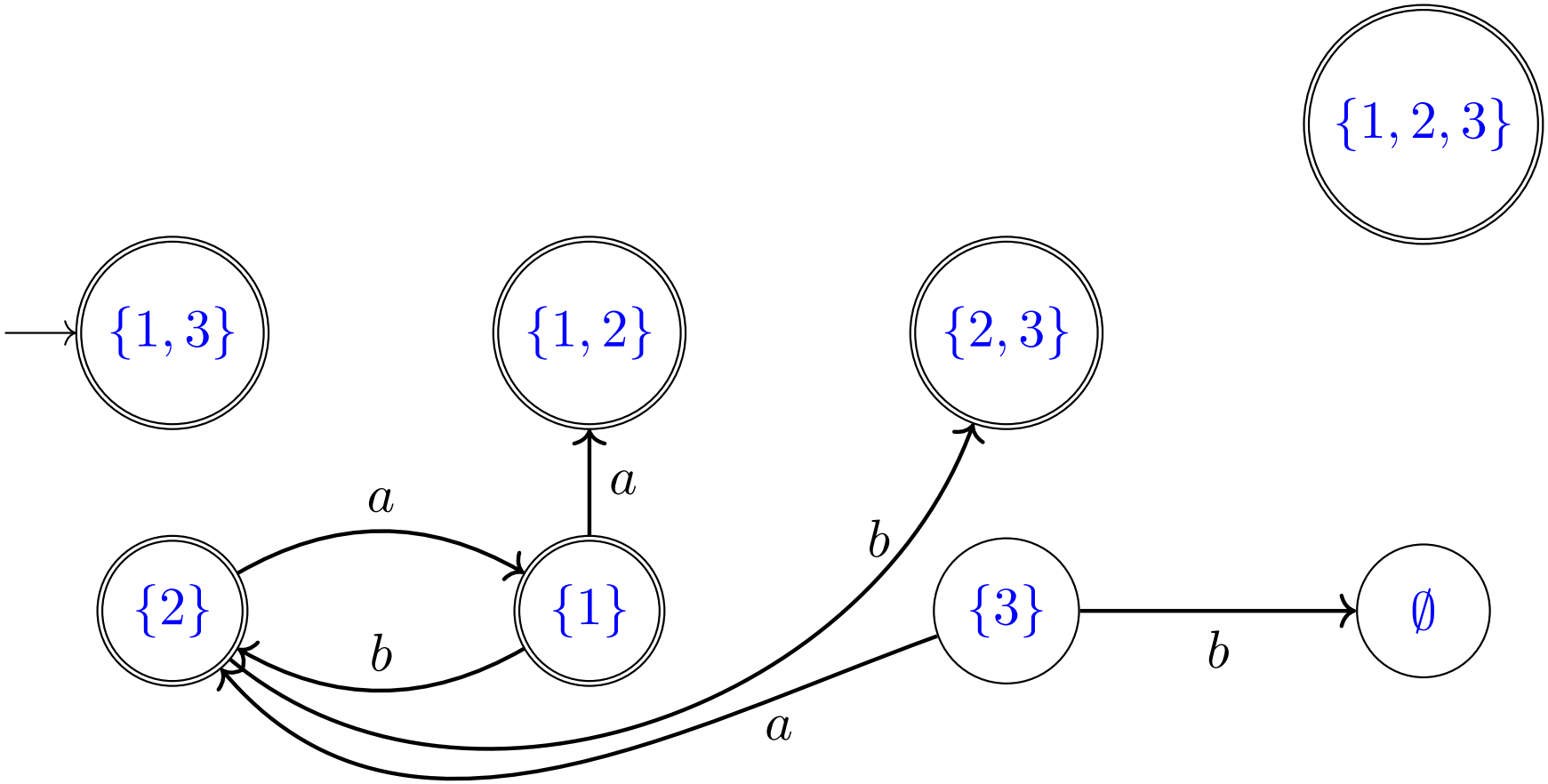
Les transitions



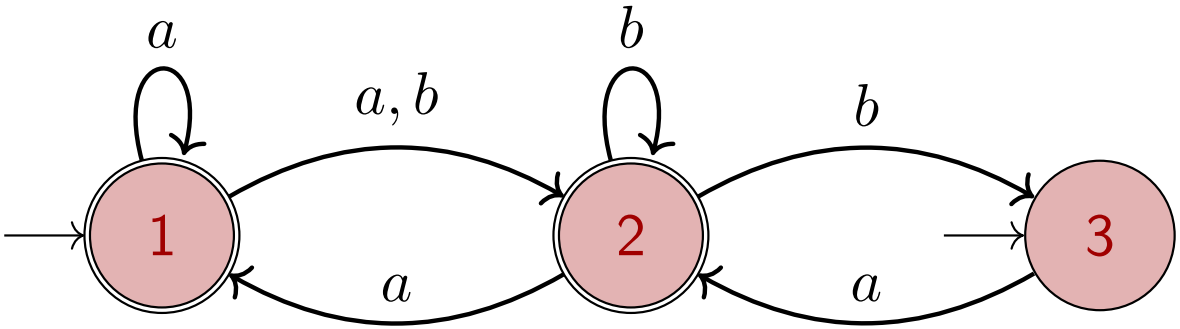


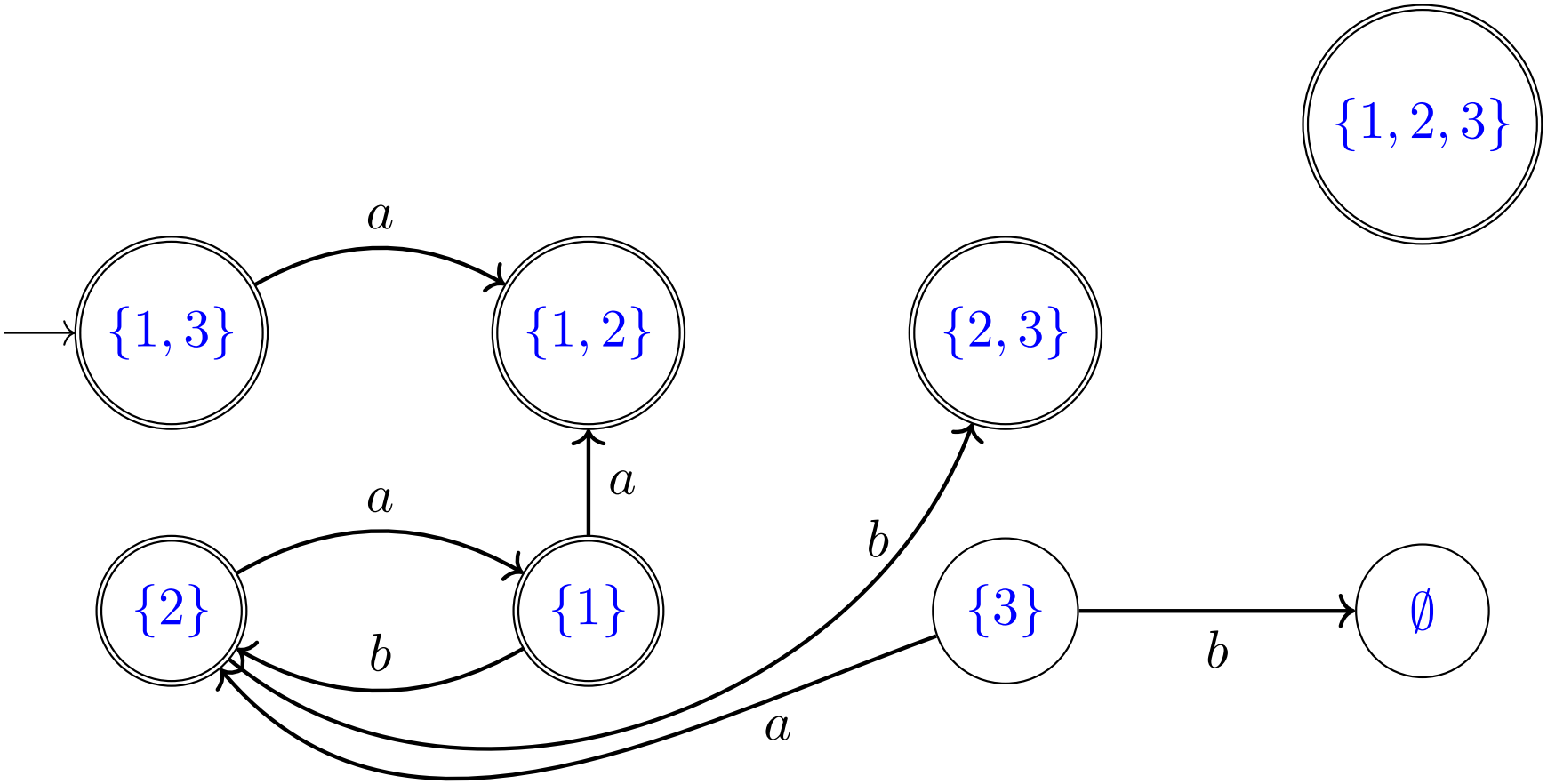
Les transitions



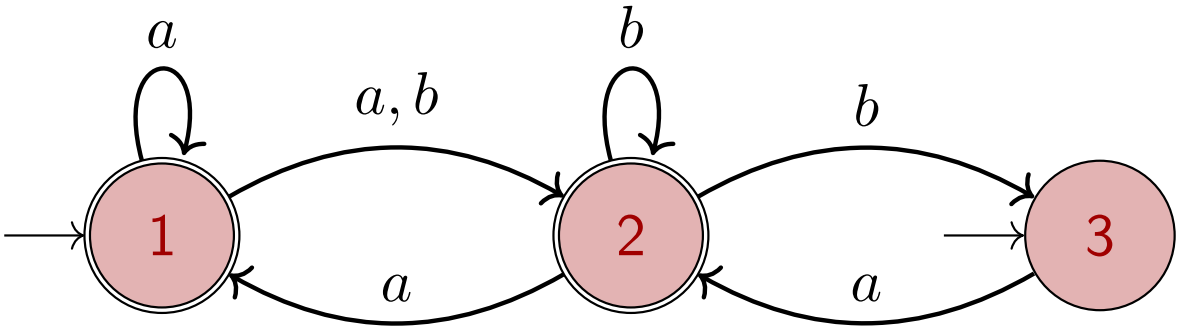


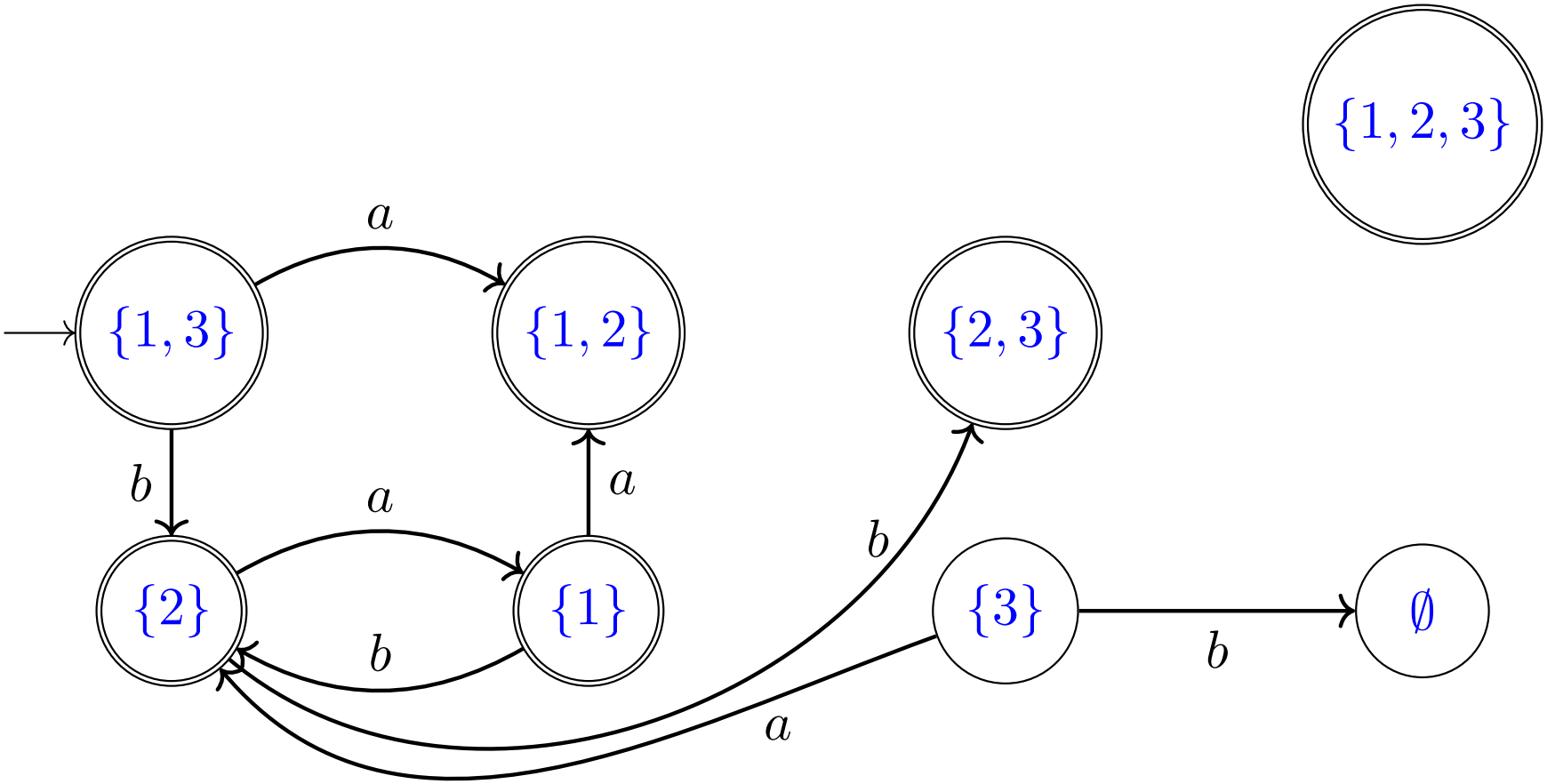
Les transitions



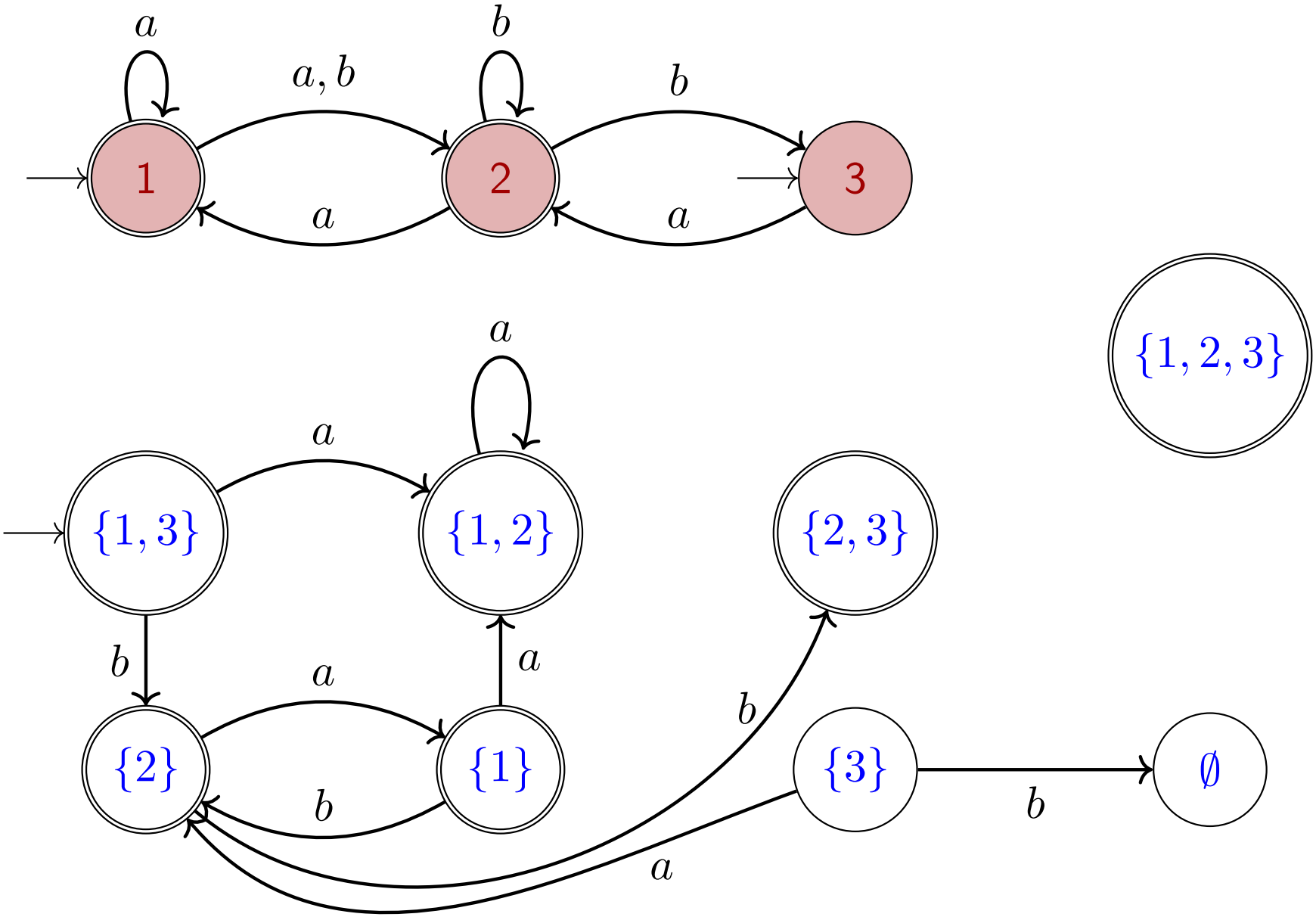


Les transitions

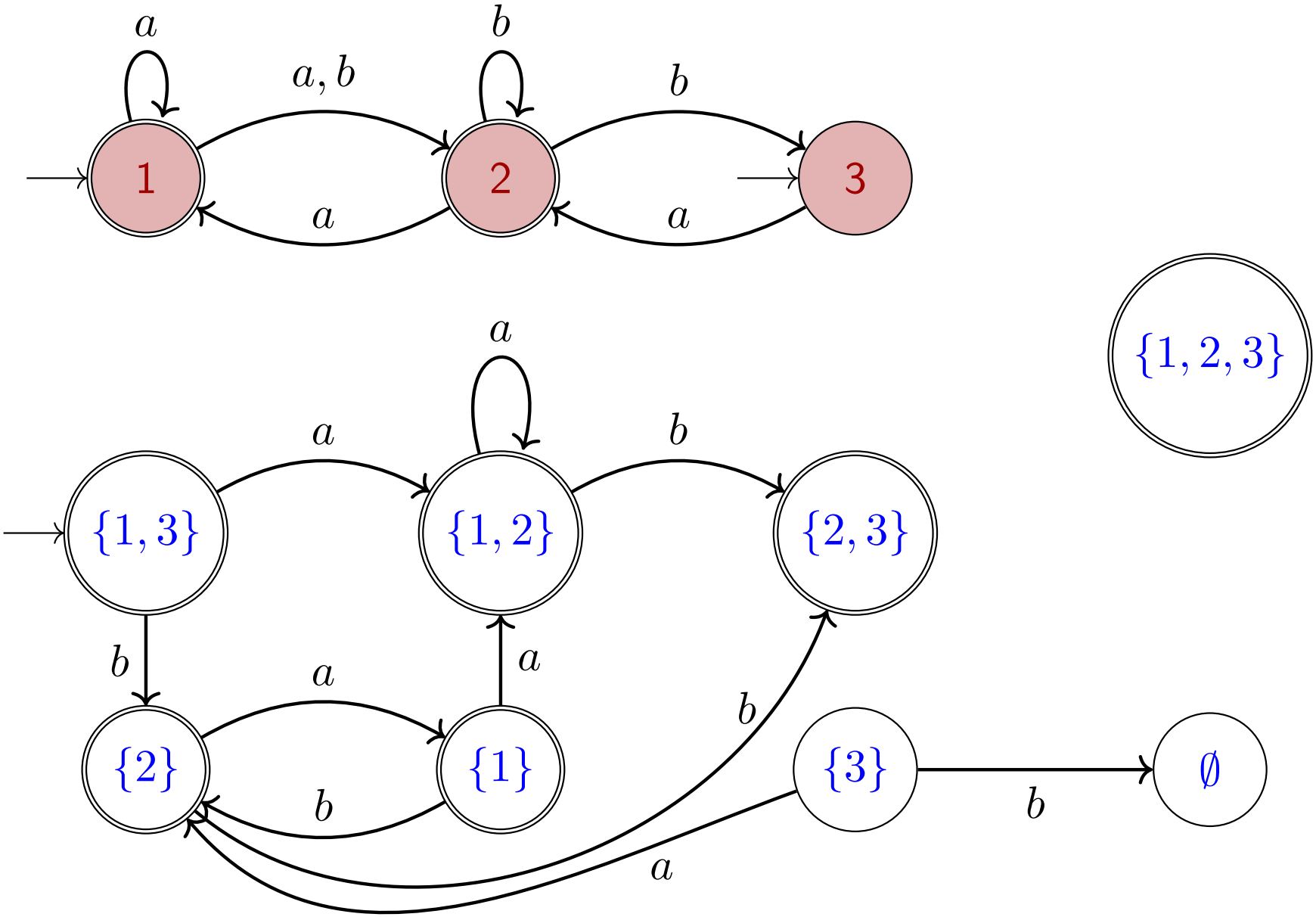




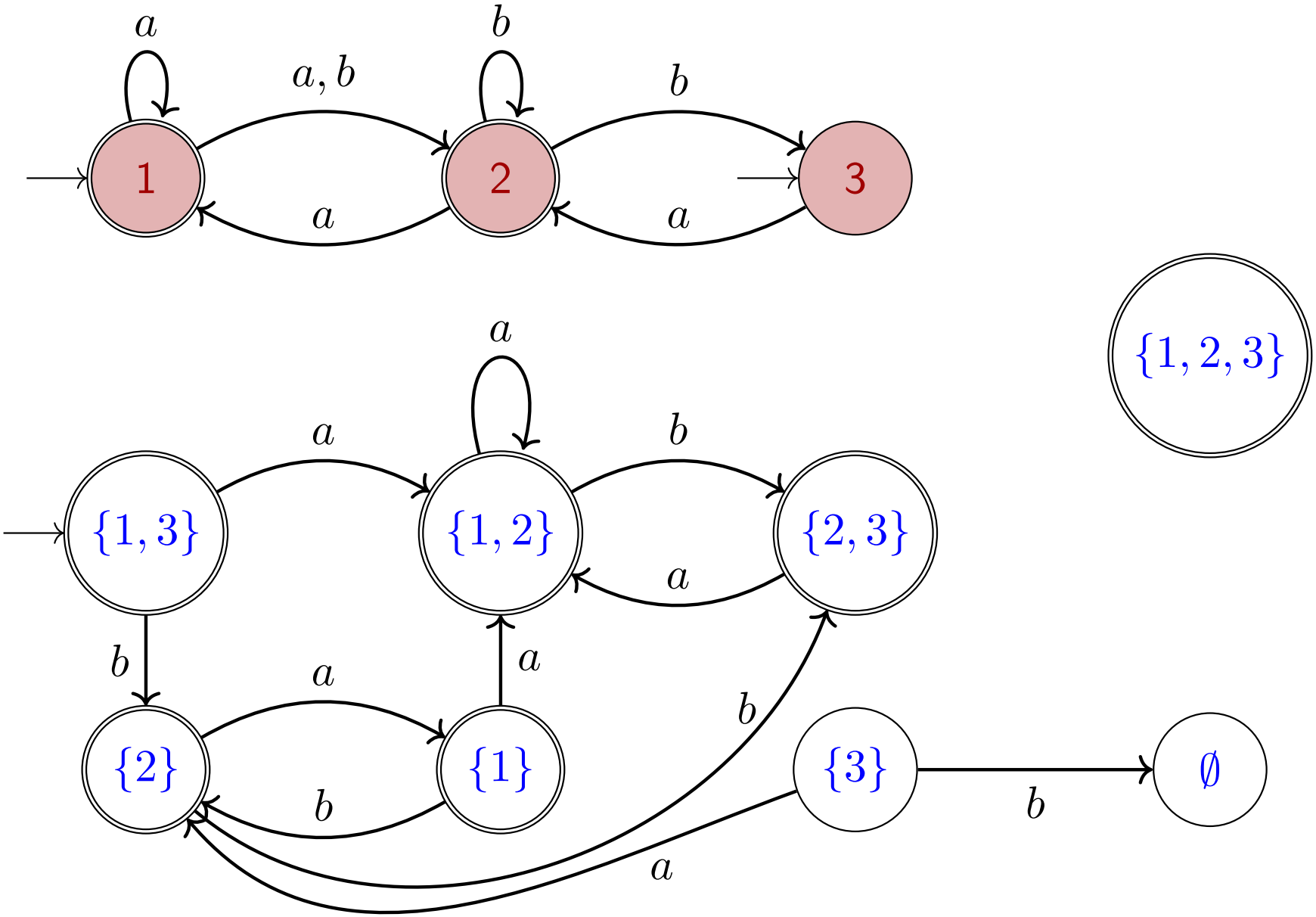
Les transitions



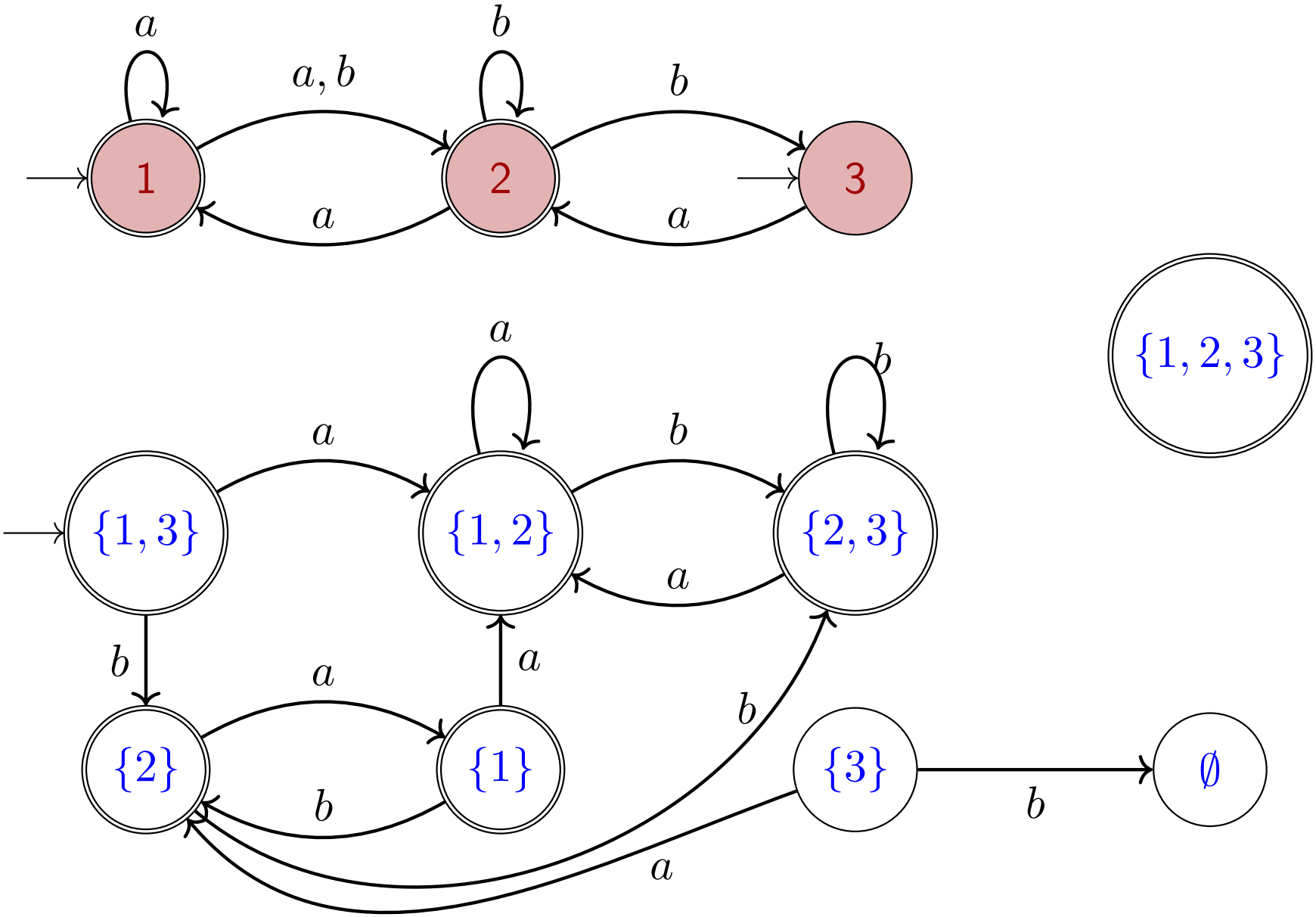
Les transitions



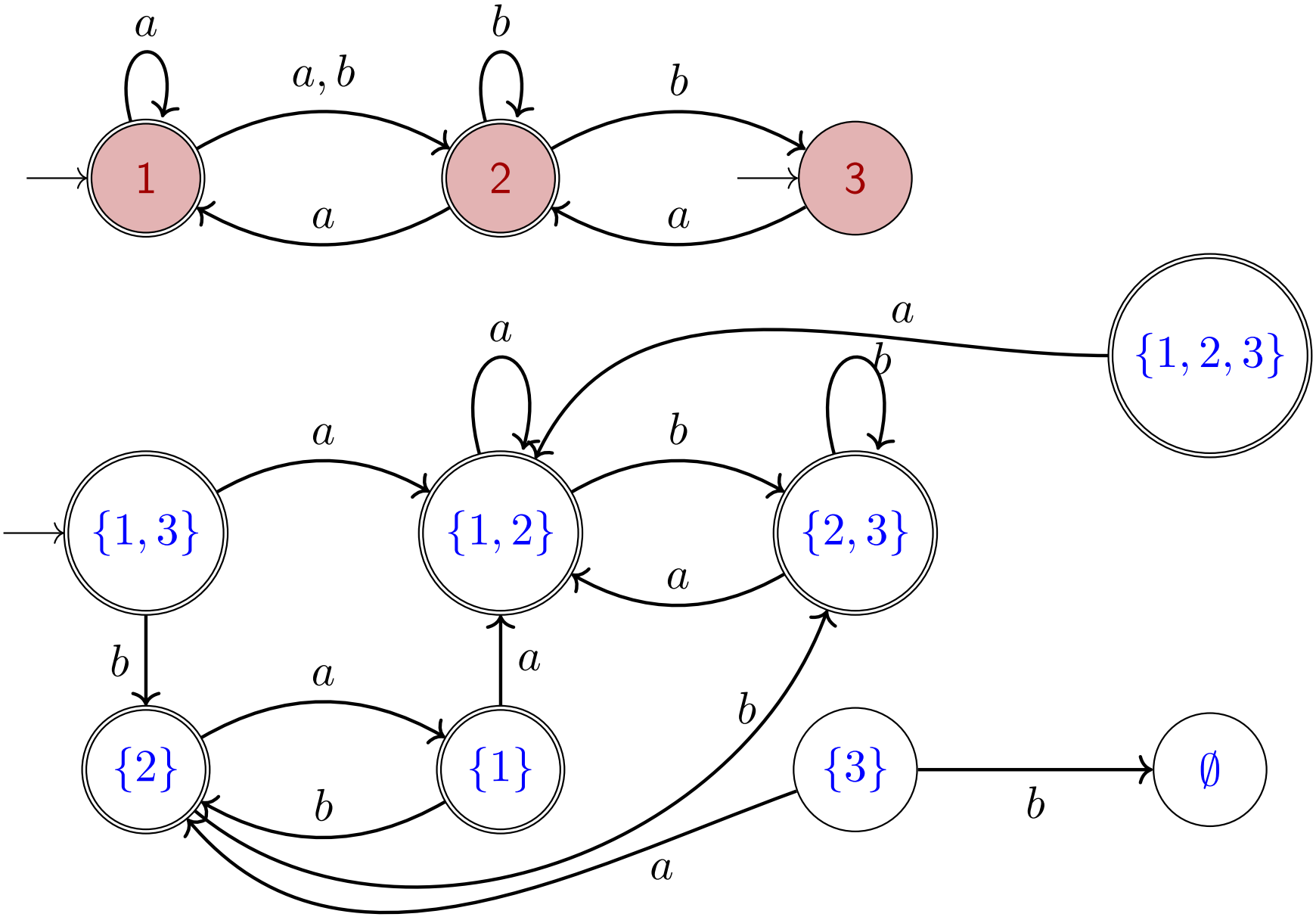
Les transitions



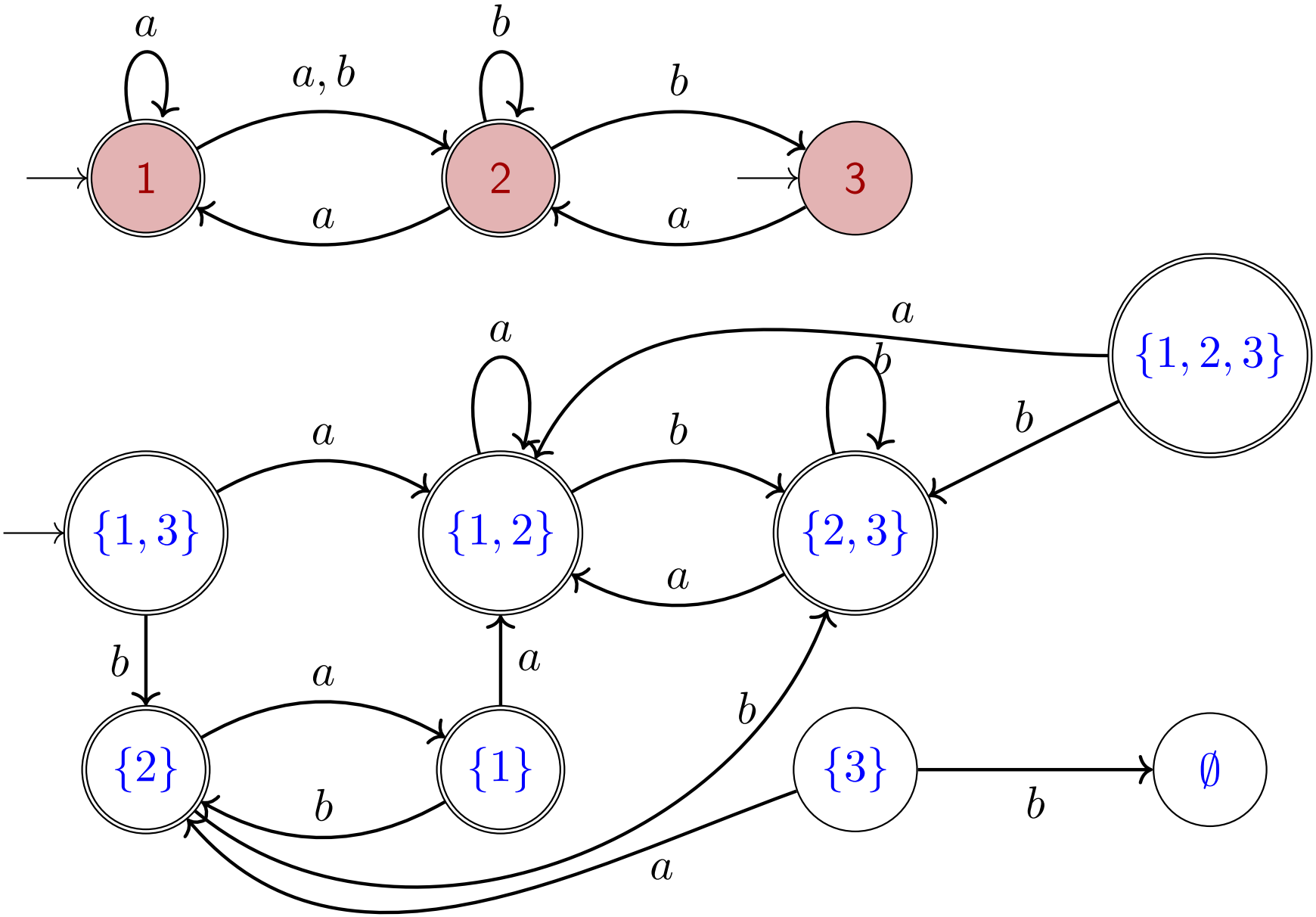
Les transitions



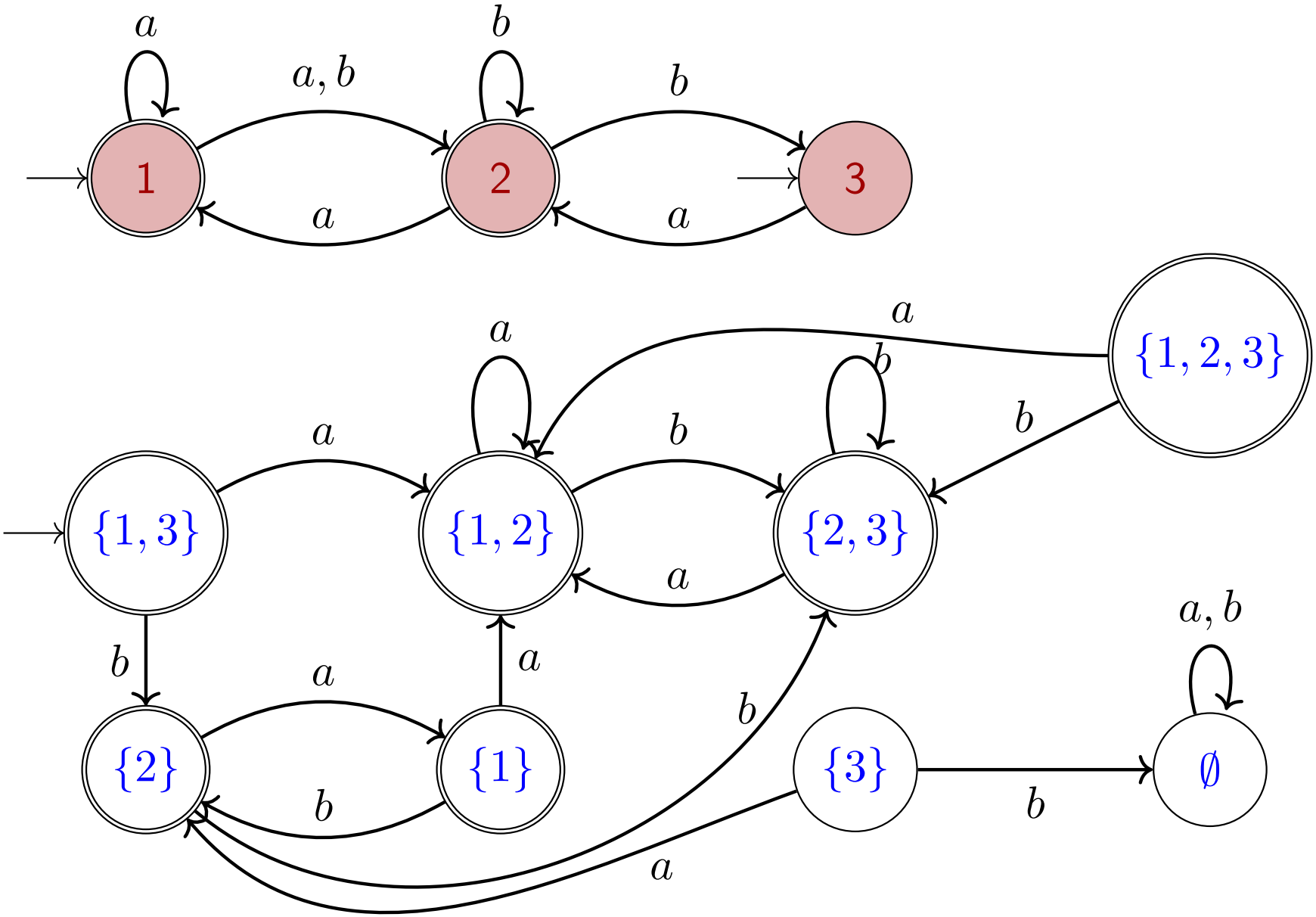
Les transitions



Les transitions



Les transitions



Exercices

Chercher l'exercice 7 la feuille de TD1.

Produit (direct) d'automates finis

Produit

Le produit (direct) d'automates finis est une opération : à deux automates sur un même alphabet, on associe un automate sur cet alphabet.

Produit (direct) d'automates finis

Produit

Le produit (direct) d'automates finis est une opération : à deux automates sur un même alphabet, on associe un automate sur cet alphabet.

- Pas de lien (évident) avec le produit de concaténation sur les mots ou les langages.
- Il existe d'autres sortes de produits, comme le produit synchronisé que nous ne montrons pas ici.

Soient $\mathcal{A} = (Q, \Sigma, E, I, F)$ et $\mathcal{B} = (Q', \Sigma, E', I', F')$ deux automates finis **sur le même alphabet** Σ. Le produit (direct) de $\mathcal{A}$ par $\mathcal{B}$, noté, $\mathcal{A} \times \mathcal{B}$ est l'automate dont :

Soient $\mathcal{A} = (Q, \Sigma, E, I, F)$ et $\mathcal{B} = (Q', \Sigma, E', I', F')$ deux automates finis **sur le même alphabet** Σ. Le produit (direct) de $\mathcal{A}$ par $\mathcal{B}$, noté, $\mathcal{A} \times \mathcal{B}$ est l'automate dont :

- L'ensemble des états est $Q \times Q'$,

Soient $\mathcal{A} = (Q, \Sigma, E, I, F)$ et $\mathcal{B} = (Q', \Sigma, E', I', F')$ deux automates finis **sur le même alphabet** Σ. Le produit (direct) de $\mathcal{A}$ par $\mathcal{B}$, noté, $\mathcal{A} \times \mathcal{B}$ est l'automate dont :

- L'ensemble des états est $Q \times Q'$,
- Les états initiaux sont tous les couples de la forme (p, p'), où $p \in I$ et $p' \in I'$,

Soient $\mathcal{A} = (Q, \Sigma, E, I, F)$ et $\mathcal{B} = (Q', \Sigma, E', I', F')$ deux automates finis **sur le même alphabet** Σ. Le produit (direct) de $\mathcal{A}$ par $\mathcal{B}$, noté, $\mathcal{A} \times \mathcal{B}$ est l'automate dont :

- L'ensemble des états est $Q \times Q'$,
- Les états initiaux sont tous les couples de la forme (p, p'), où $p \in I$ et $p' \in I'$,
- Les états initiaux sont tous les couples de la forme (p, p'), où $p \in F$ et $p' \in F'$,

Soient $\mathcal{A} = (Q, \Sigma, E, I, F)$ et $\mathcal{B} = (Q', \Sigma, E', I', F')$ deux automates finis **sur le même alphabet** Σ. Le produit (direct) de $\mathcal{A}$ par $\mathcal{B}$, noté, $\mathcal{A} \times \mathcal{B}$ est l'automate dont :

- L'ensemble des états est $Q \times Q'$,
- Les états initiaux sont tous les couples de la forme (p, p'), où $p \in I$ et $p' \in I'$,
- Les états initiaux sont tous les couples de la forme (p, p'), où $p \in F$ et $p' \in F'$,
- Les transitions sont tous les triplets $((p, p'), a, (q, q'))$ tels que $(p, a, q) \in E$ et $(p', a, q') \in E'$

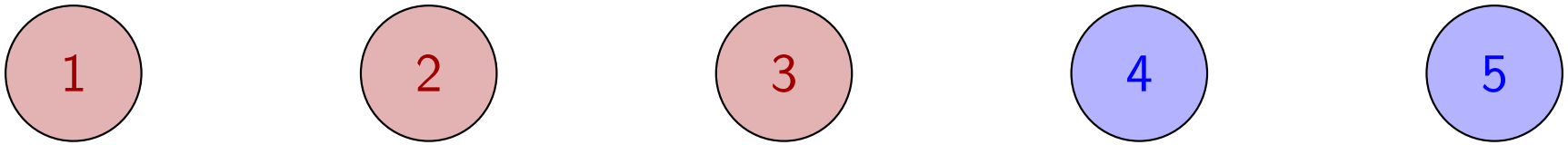
Soient $\mathcal{A} = (Q, \Sigma, E, I, F)$ et $\mathcal{B} = (Q', \Sigma, E', I', F')$ deux automates finis **sur le même alphabet** Σ. Le produit (direct) de $\mathcal{A}$ par $\mathcal{B}$, noté, $\mathcal{A} \times \mathcal{B}$ est l'automate dont :

- L'ensemble des états est $Q \times Q'$,
- Les états initiaux sont tous les couples de la forme (p, p'), où $p \in I$ et $p' \in I'$,
- Les états initiaux sont tous les couples de la forme (p, p'), où $p \in F$ et $p' \in F'$,
- Les transitions sont tous les triplets $((p, p'), a, (q, q'))$ tels que $(p, a, q) \in E$ et $(p', a, q') \in E'$

Propriété : On a $L(\mathcal{A} \times \mathcal{B}) = L(\mathcal{A}) \cap L(\mathcal{B})$.

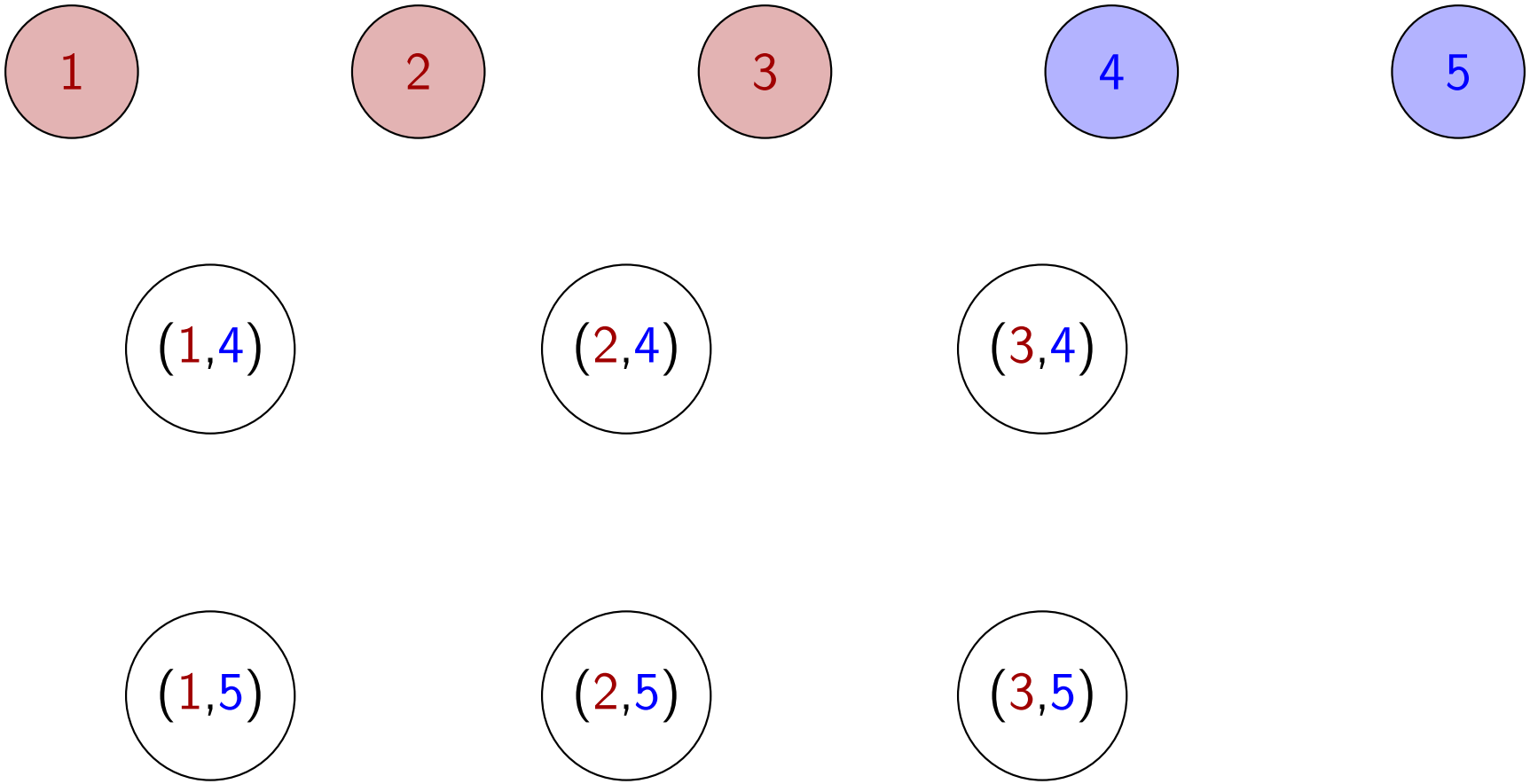
Les états

Supposons que $Q = \{1, 2, 3\}$ et $Q' = \{4, 5\}$.



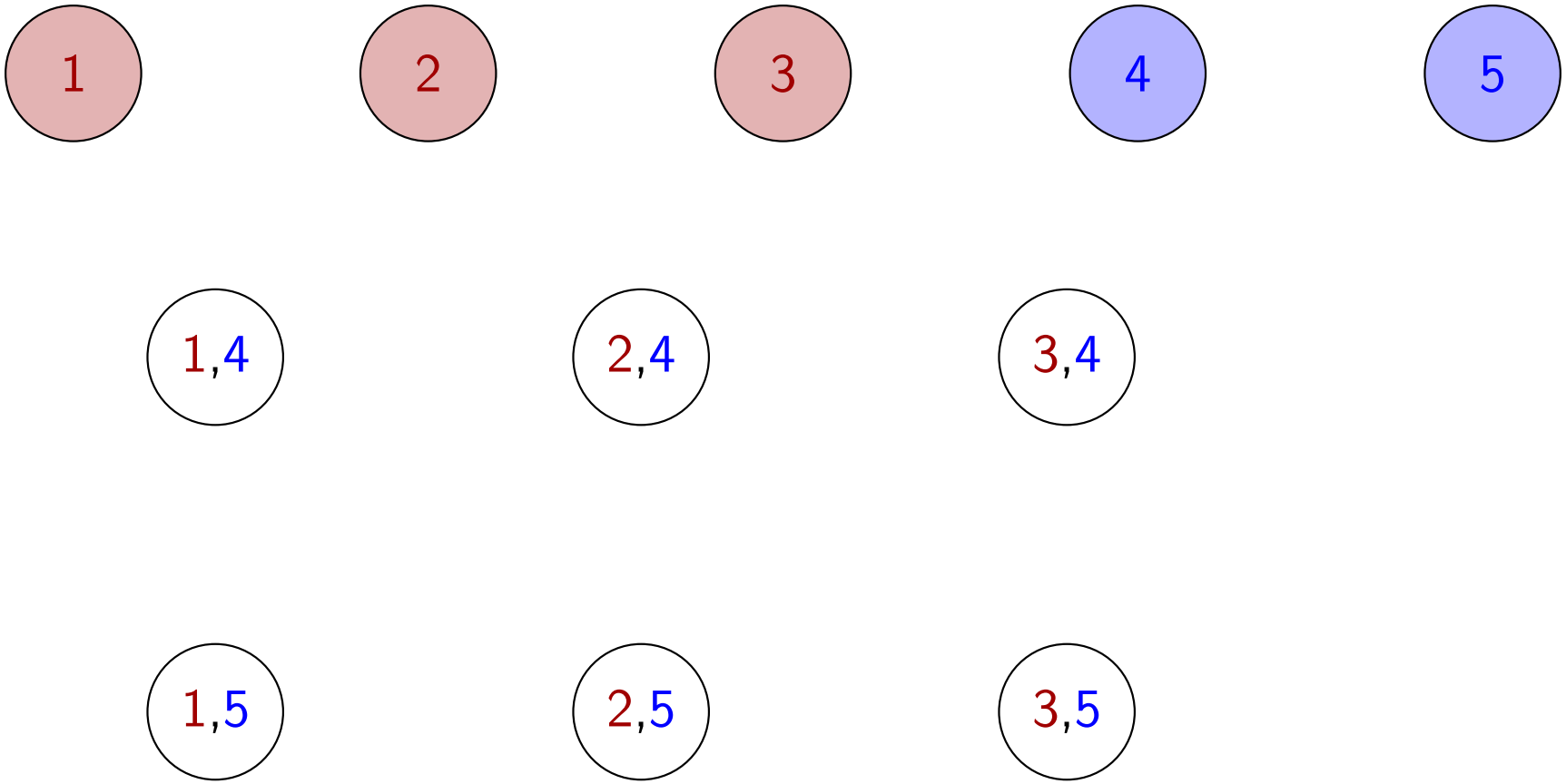
Les états

Supposons que $Q = \{1, 2, 3\}$ et $Q' = \{4, 5\}$.



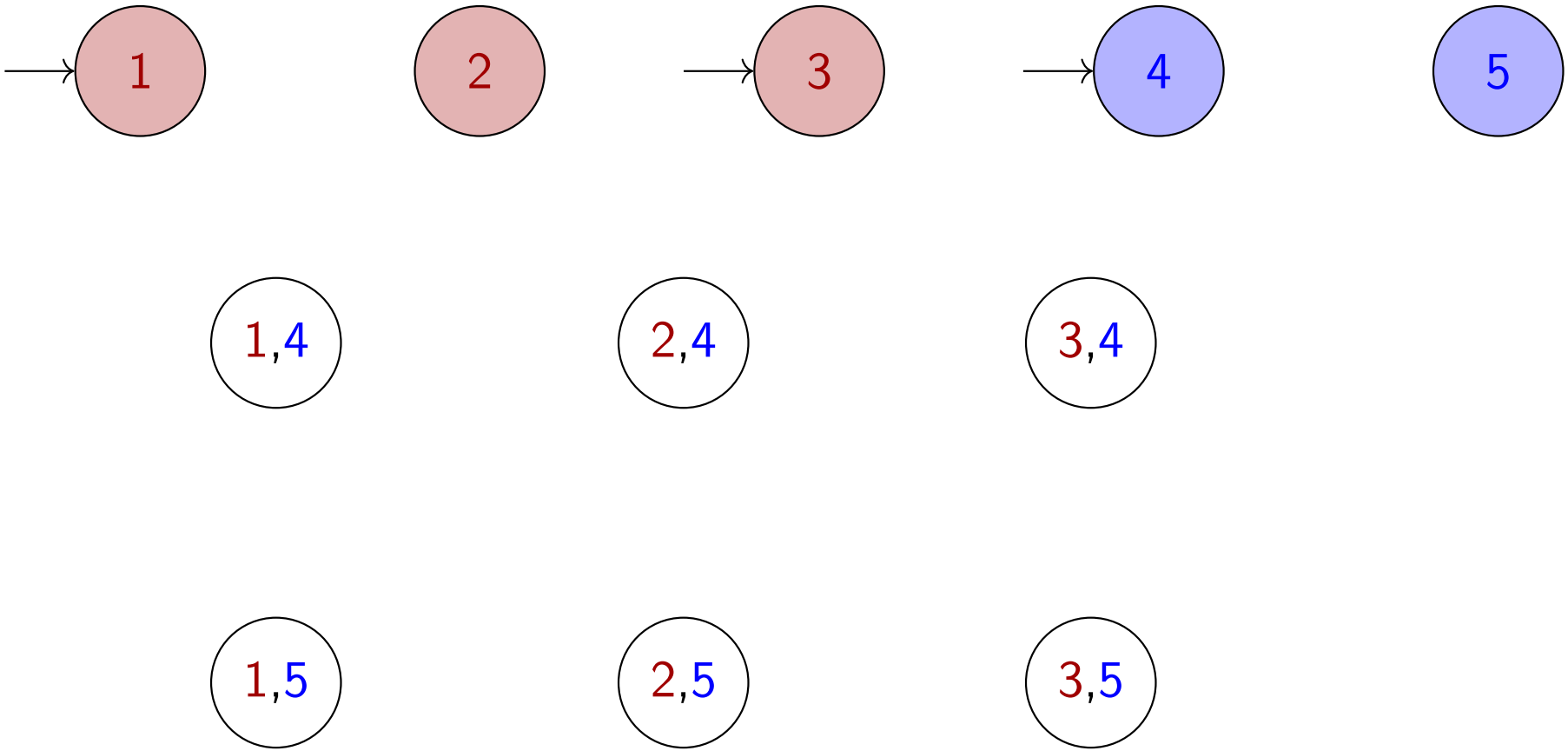
Les états

Supposons que $Q = \{1, 2, 3\}$ et $Q' = \{4, 5\}$.



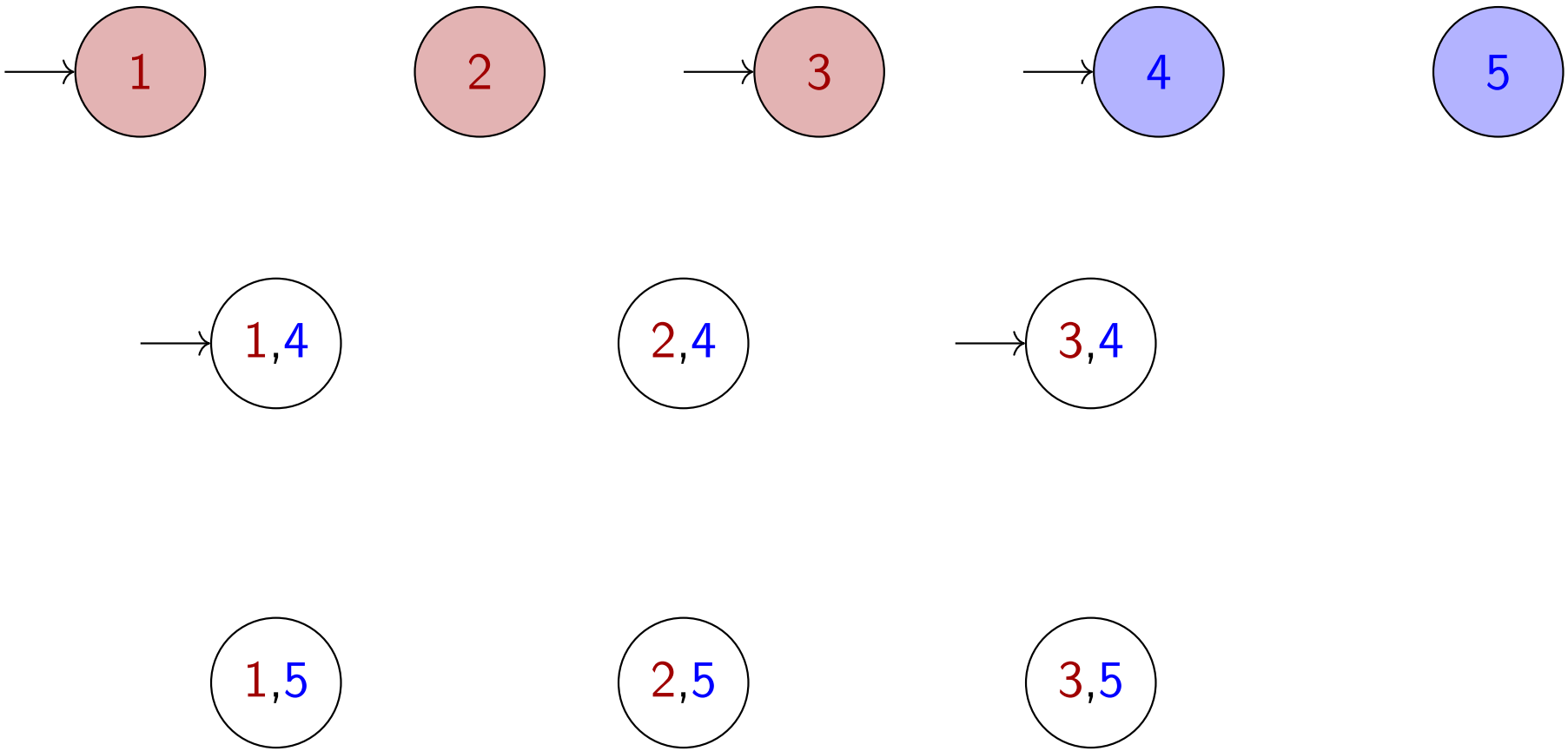
Les états initiaux

$Q = \{1, 2, 3\}$ et $Q' = \{4, 5\}$, et $I = \{1, 3\}$ et $I' = \{4\}$



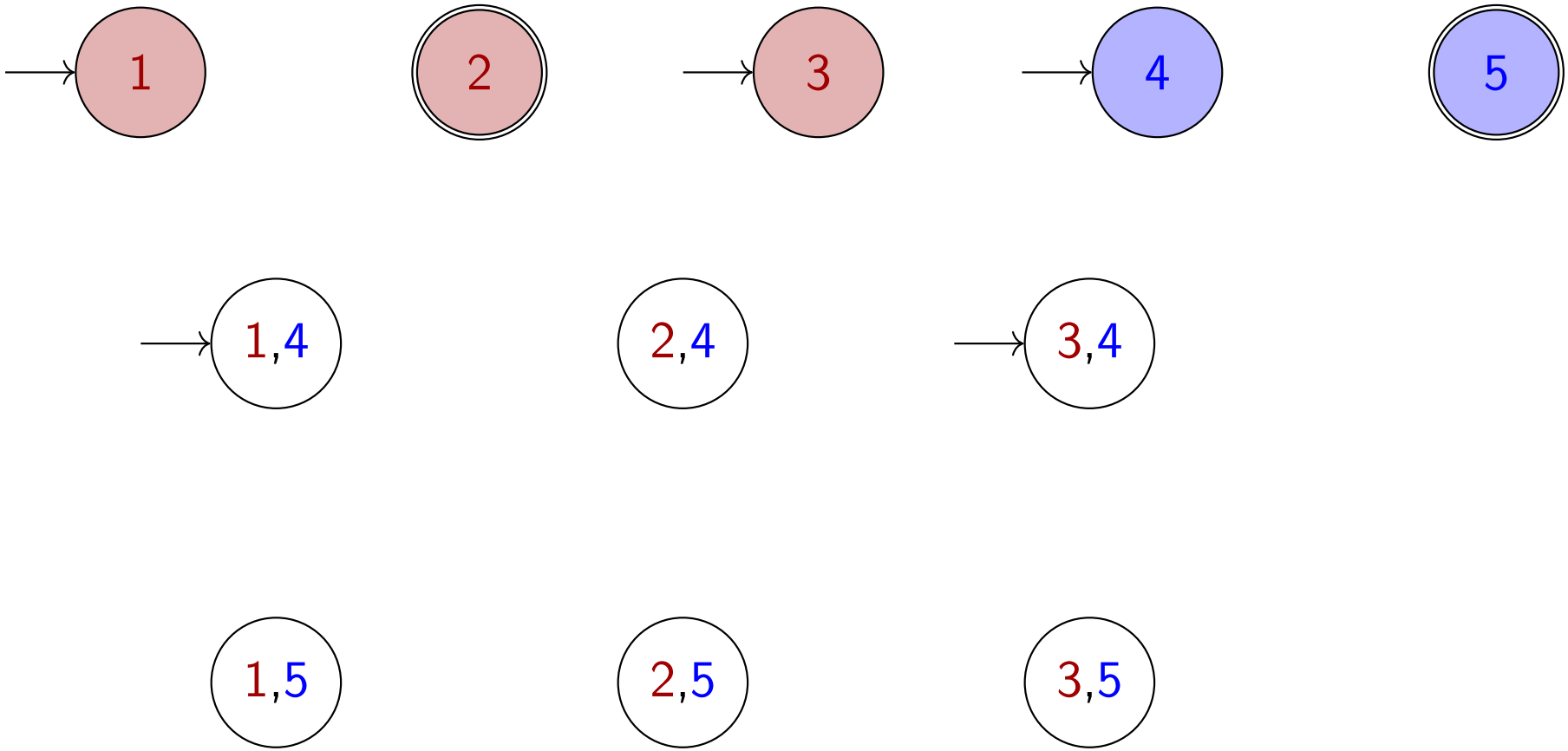
Les états initiaux

$Q = \{1, 2, 3\}$ et $Q' = \{4, 5\}$, et $I = \{1, 3\}$ et $I' = \{4\}$



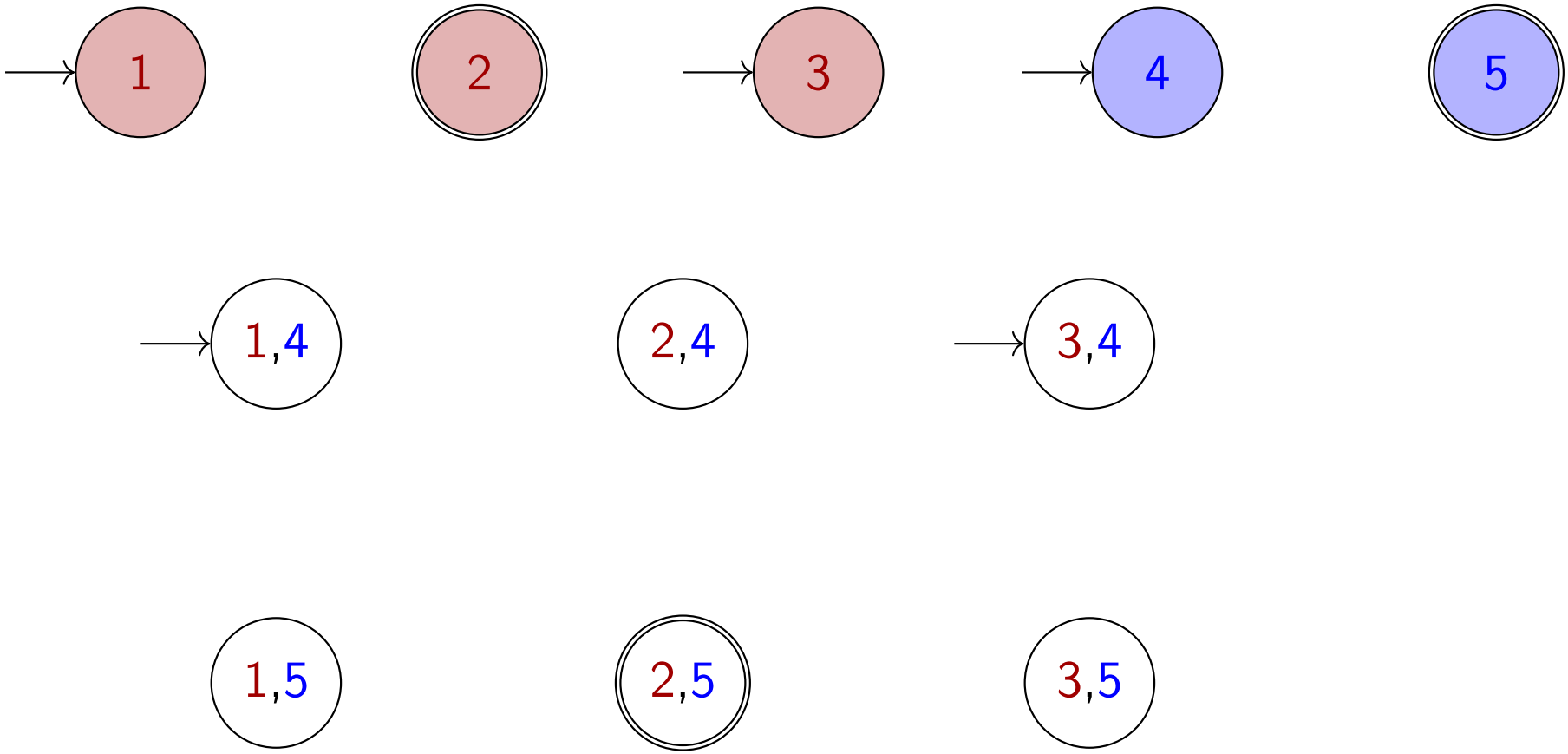
Les états initiaux – finaux

$Q = \{1, 2, 3\}$ et $Q' = \{4, 5\}$, et $I = \{1, 3\}$ et $I' = \{4\}$, et $F = \{2\}$ et $F' = \{5\}$.

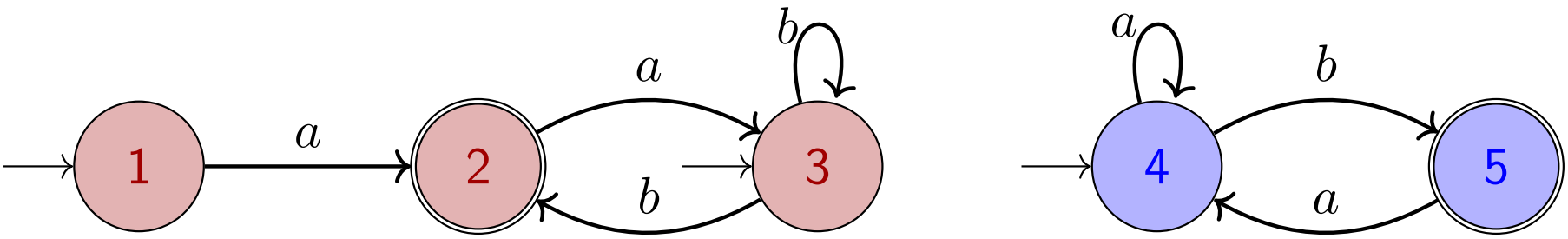


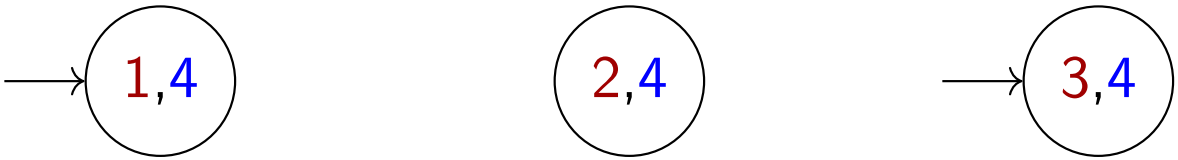
Les états initiaux – finaux

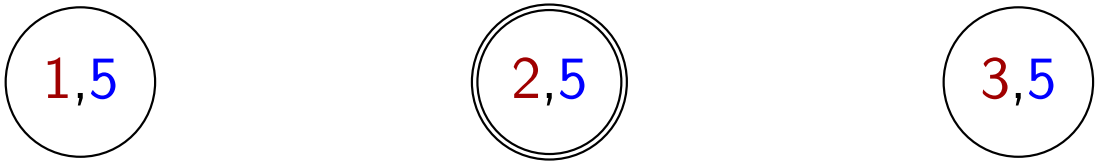
$Q = \{1, 2, 3\}$ et $Q' = \{4, 5\}$, et $I = \{1, 3\}$ et $I' = \{4\}$, et $F = \{2\}$ et $F' = \{5\}$.



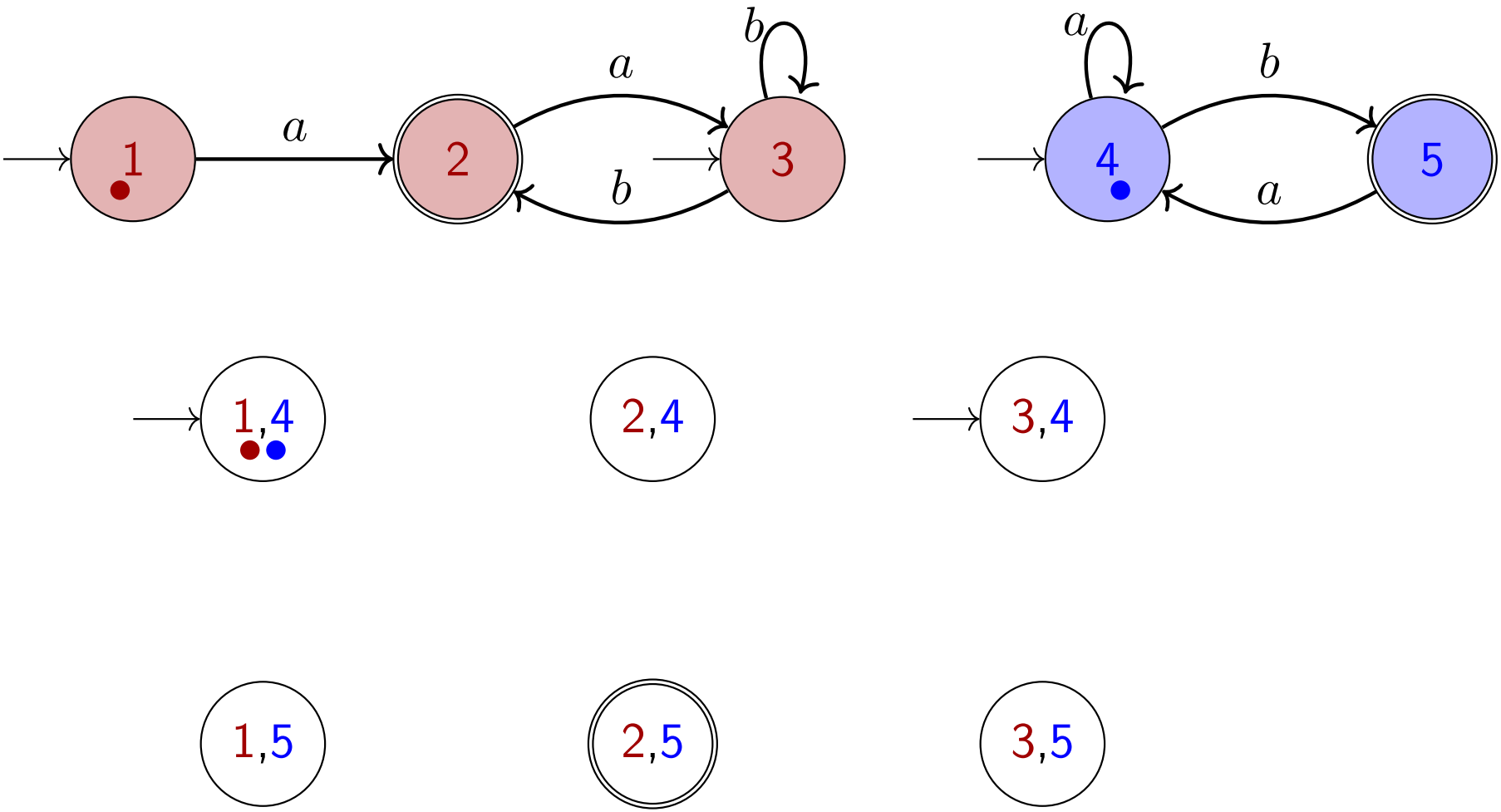
Les transitions



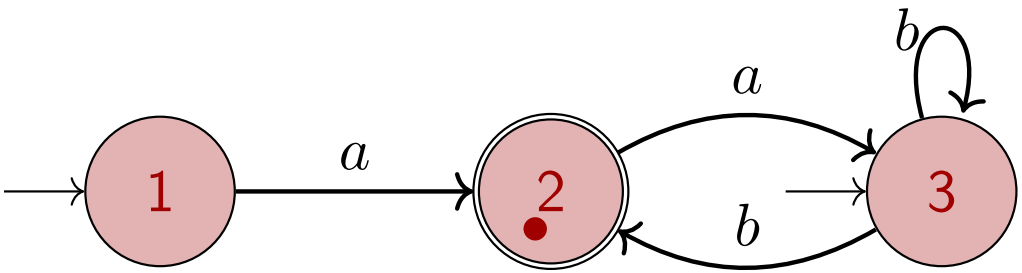


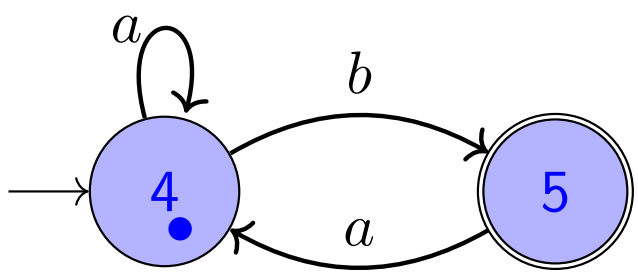


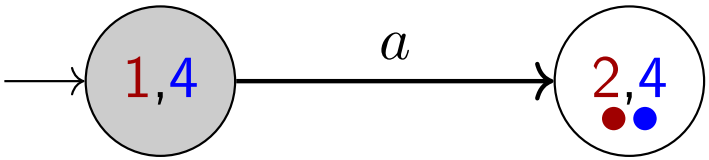
Les transitions

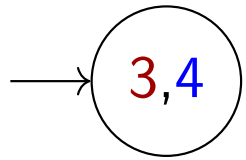


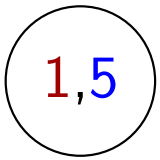
Les transitions

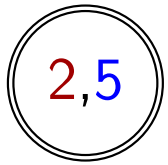


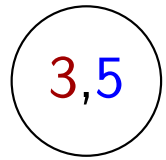




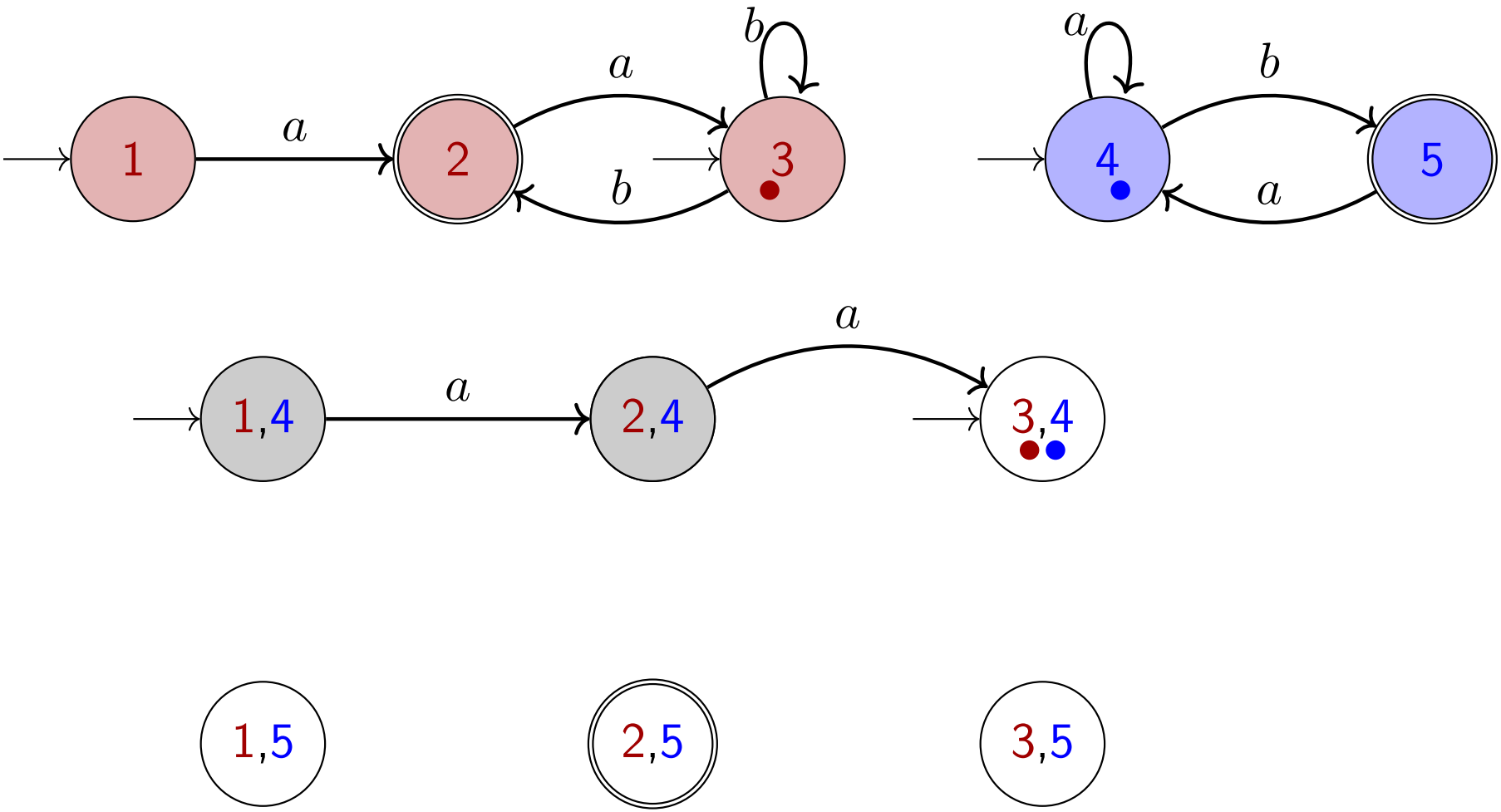




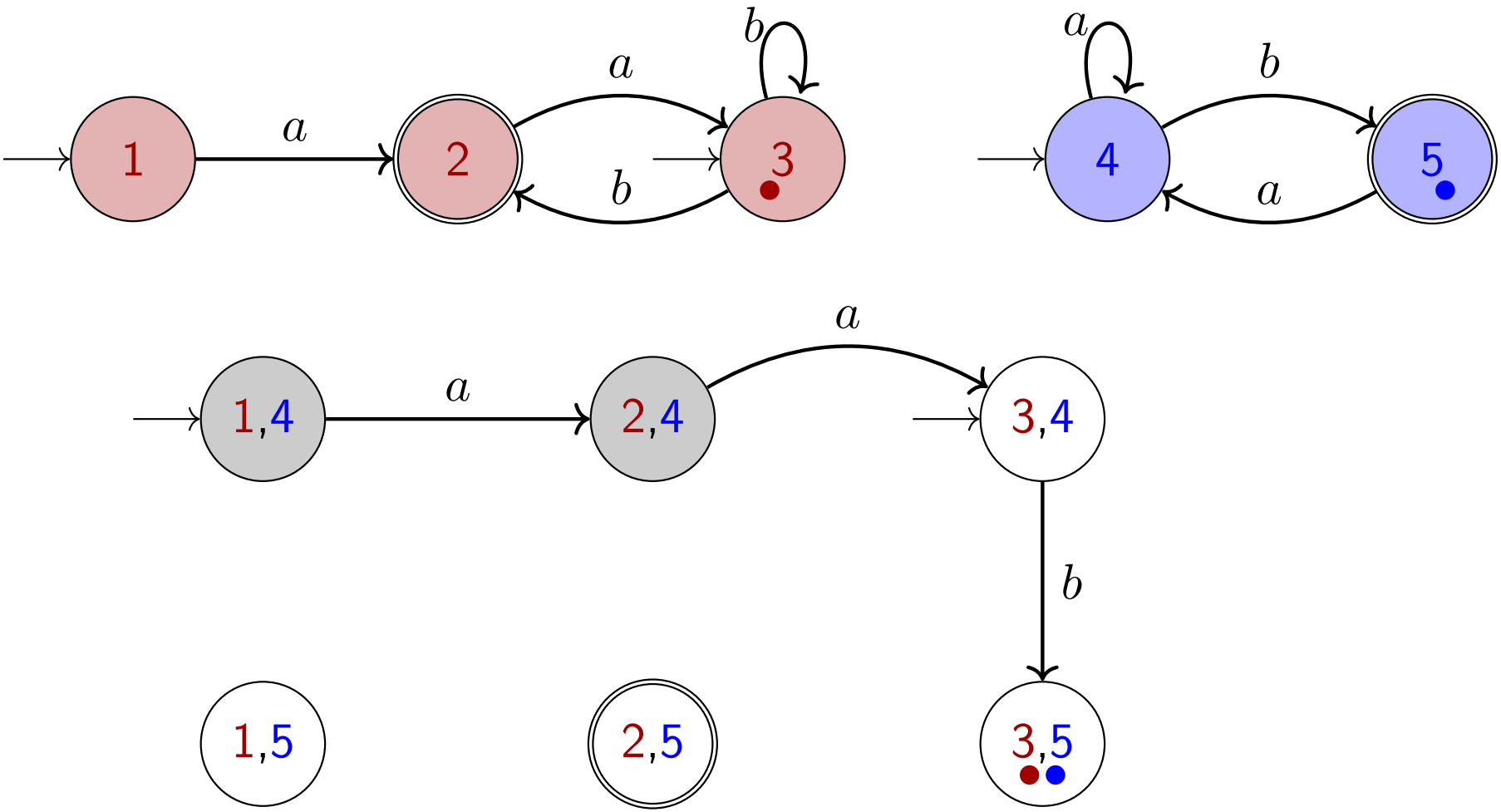




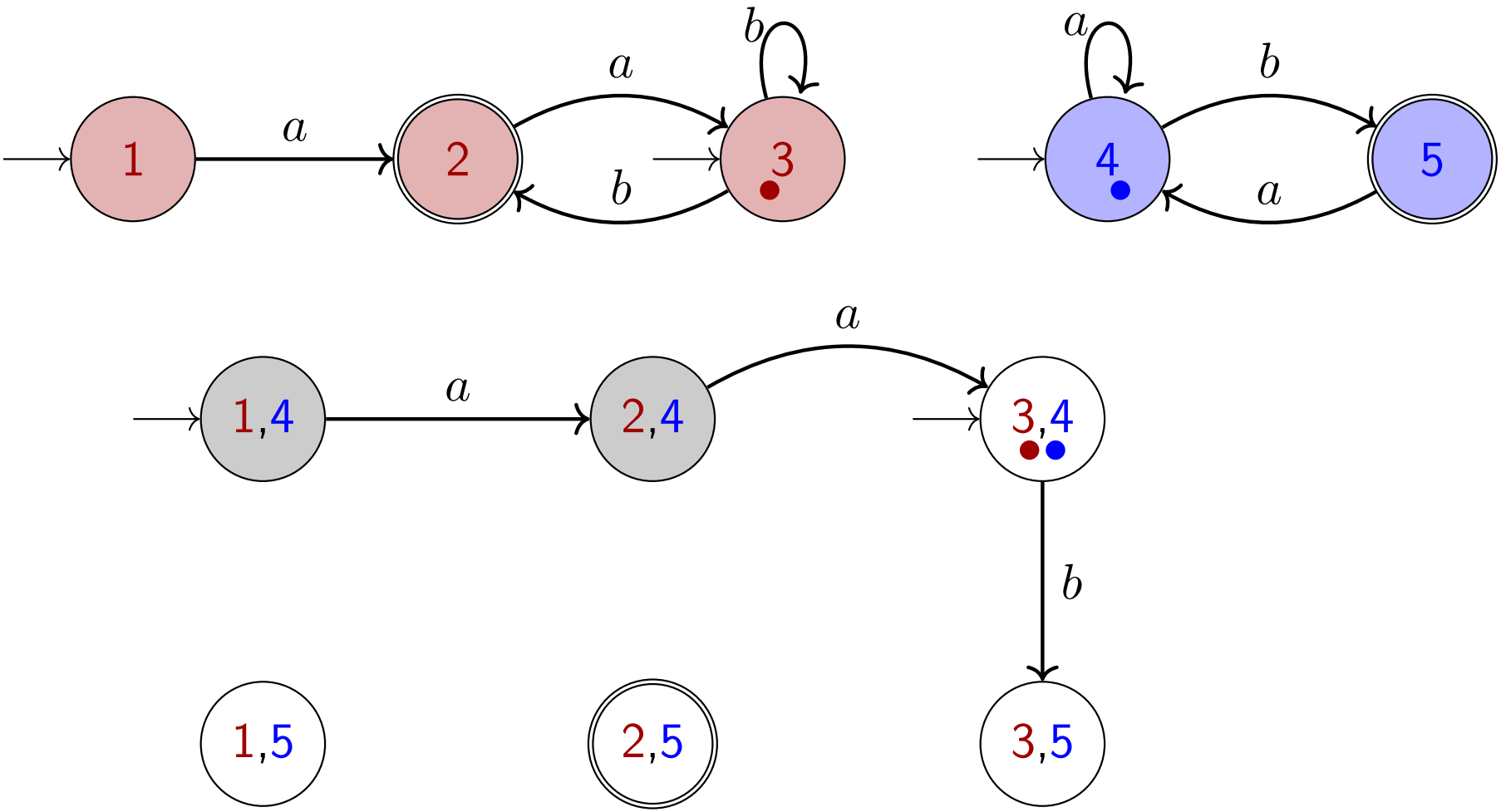
Les transitions



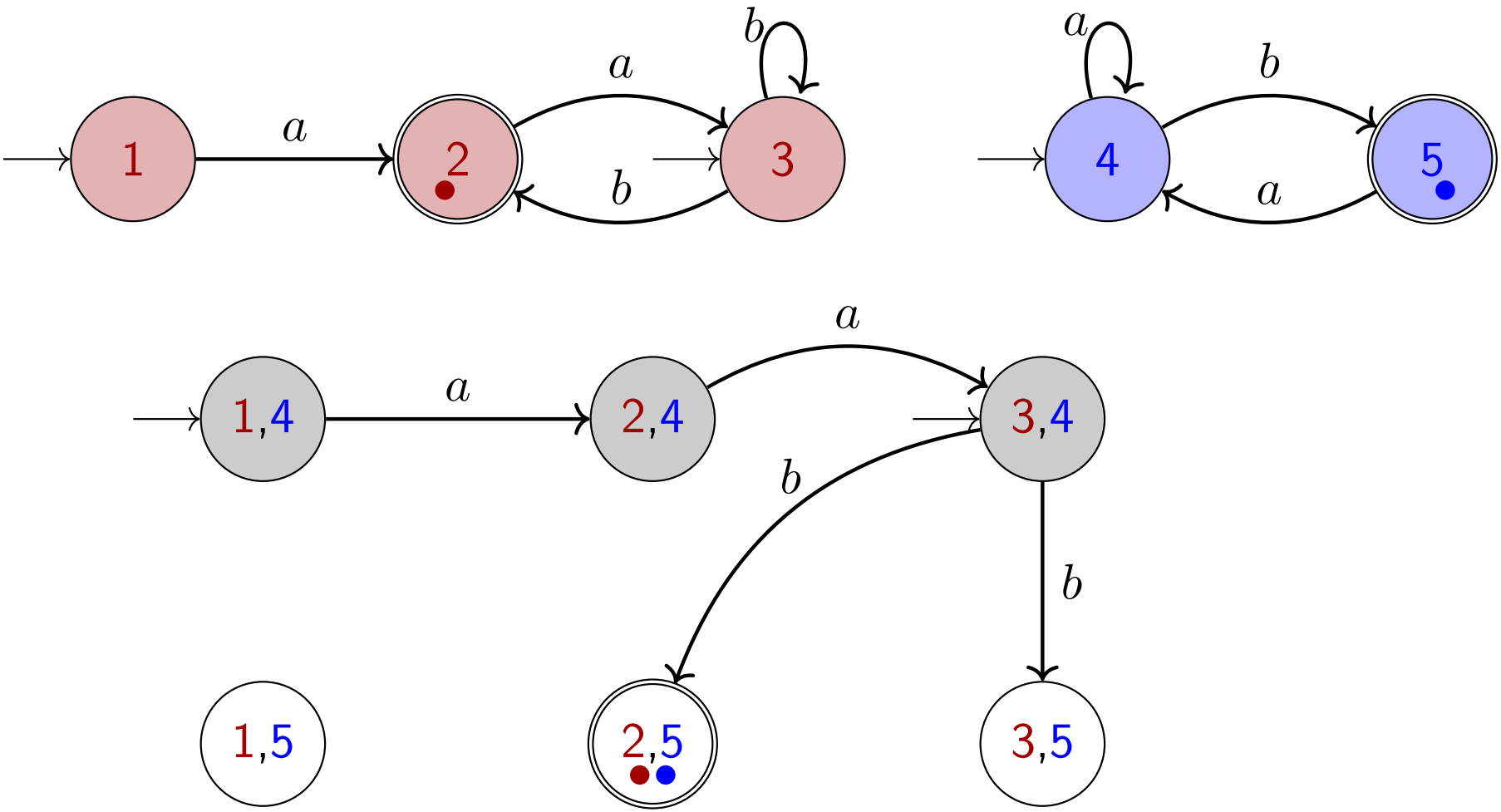
Les transitions



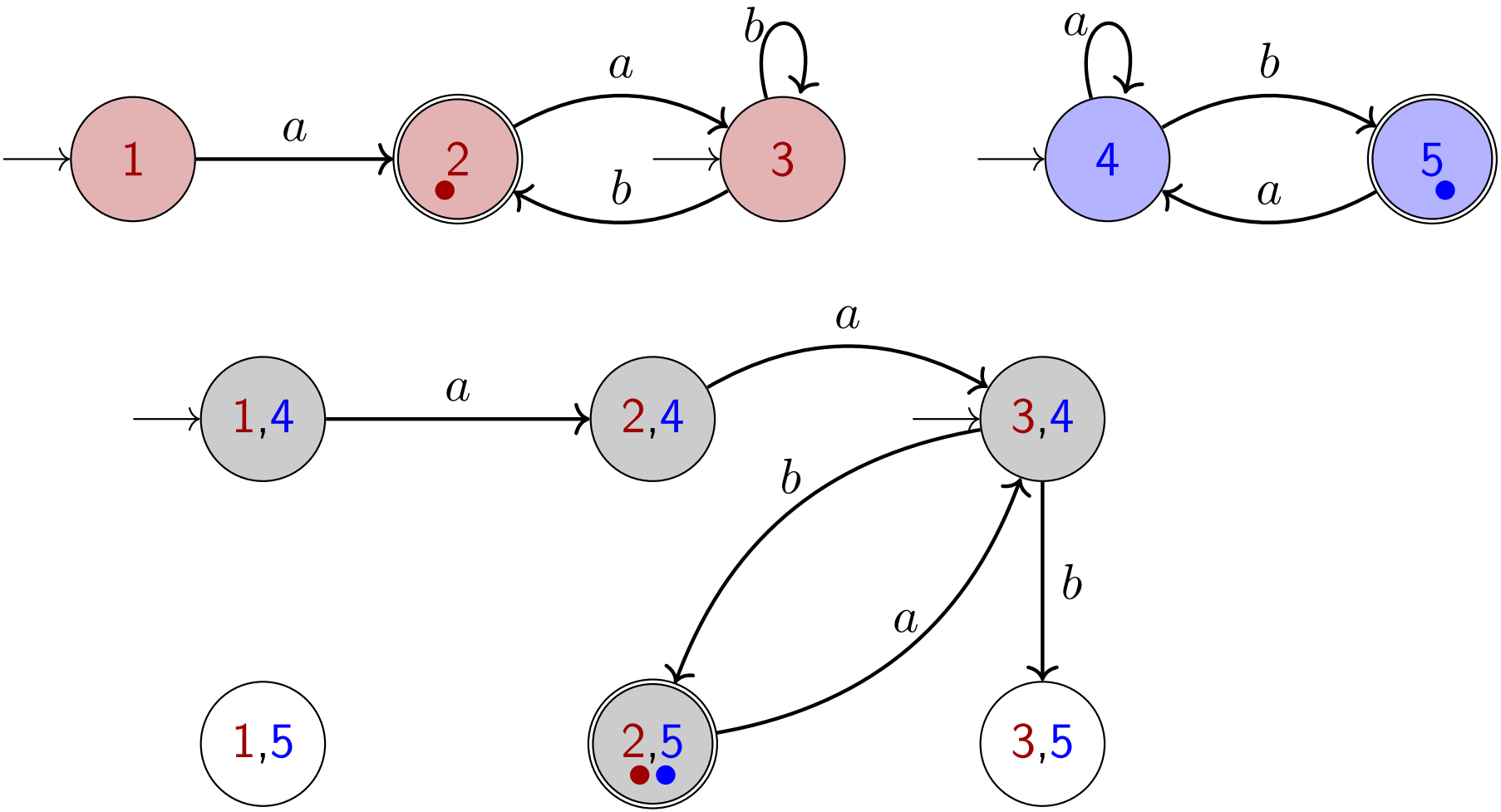
Les transitions



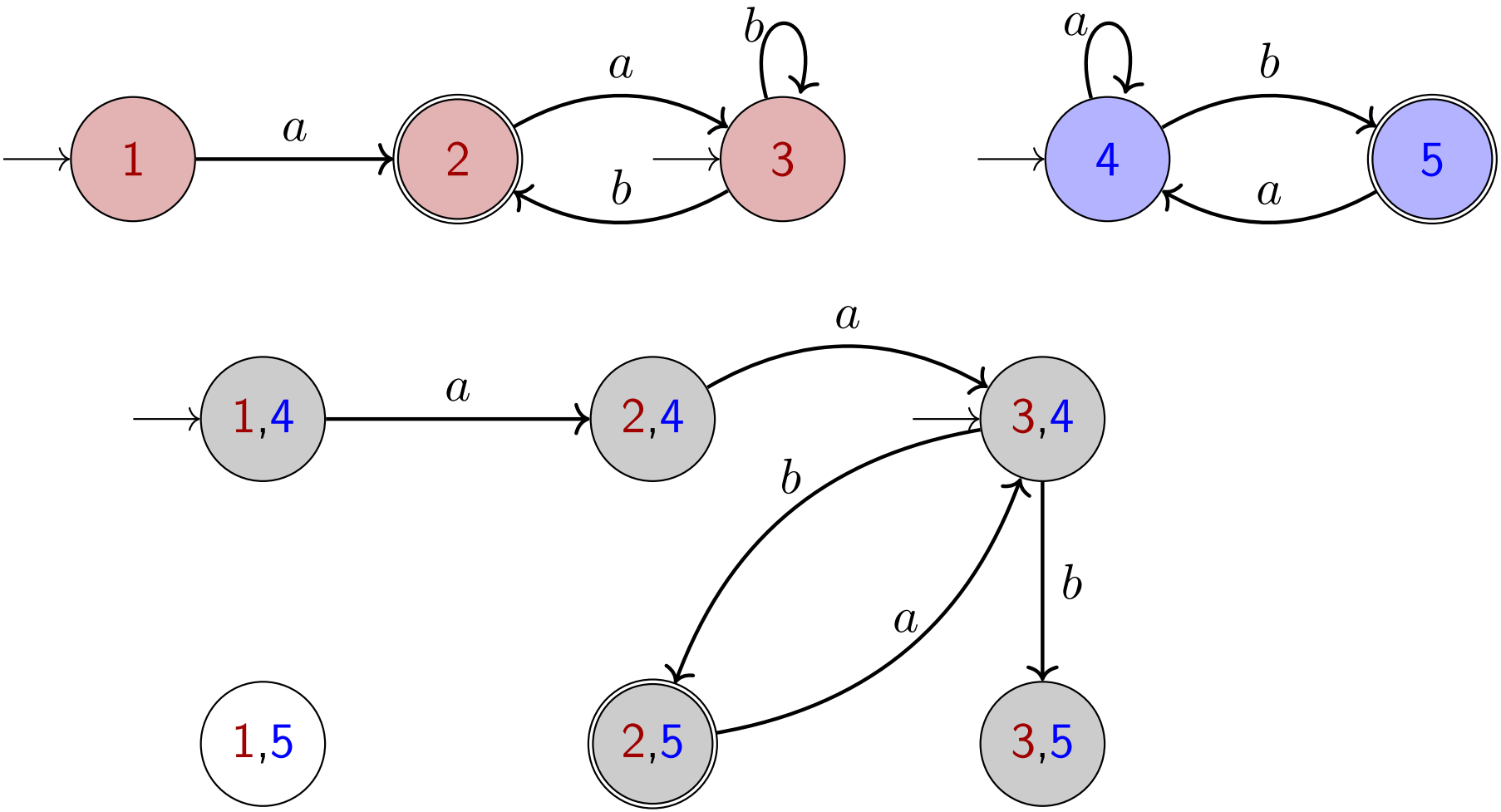
Les transitions



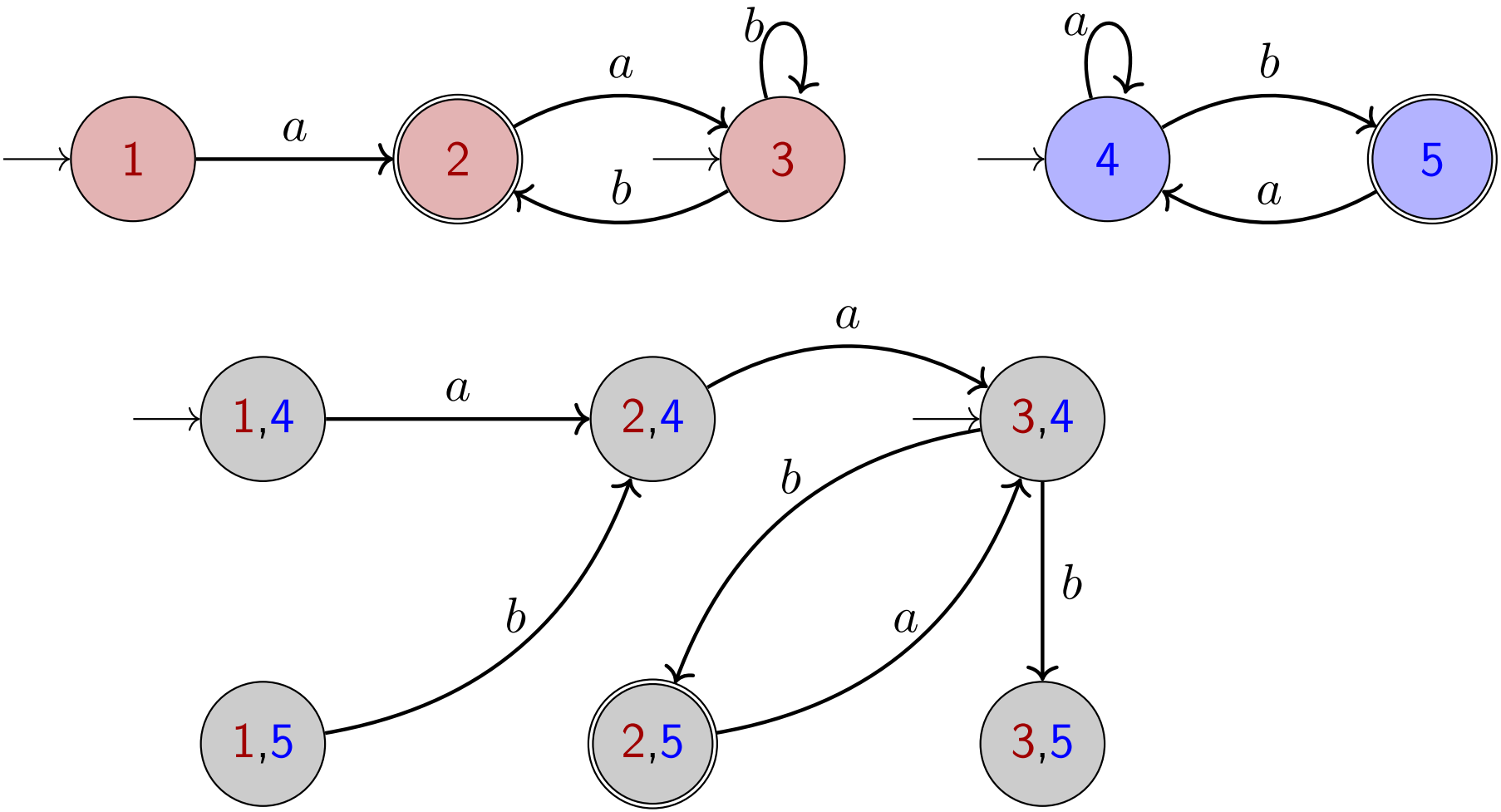
Les transitions



Les transitions



Les transitions



Exercices

Chercher l'exercice 8 la feuille de TD1.

Complément

Soit L un langage reconnu par un automate $\mathcal{A}$. Pour construire un automate qui reconnaît le complémentaire de L, c'est-à-dire $\Sigma^* \setminus L$, on procède en trois étapes.

- On déterminise $\mathcal{A}$ en un automate $\mathcal{B}$.
- On **complète** $\mathcal{B}$ en ajoutant éventuellement un état puits.
- On inverse les caractères initiaux et finaux des états.

Exercices

Chercher l'exercice 9 la feuille de TD1.

Plan du module

- Introduction, rappels automates.
- Model-Checking LTL.
- Modéliser la concurrence et les systèmes communicants.
- CTL.
- Utilisation des modèles pour le test.

Plan

Mots infinis

Soit Σ un alphabet fini. Un mot infini sur Σ est une suite infinie d'éléments de Σ. On note Σ^ω l'ensemble des mots infinis sur Σ.

Mots infinis

Soit Σ un alphabet fini. Un mot infini sur Σ est une suite infinie d'éléments de Σ. On note Σ^ω l'ensemble des mots infinis sur Σ.

Exemple : $u = ababbbbbbbb\ldots$

- $u(1) = a$,
- $u(2) = b$,
- $u(3) = a$,
- $u(4) = b$,
- $u(5) = b$,
- $u(i) = b$ pour $i \geq 4$.

Mots infinis

Soit Σ un alphabet fini. Un mot infini sur Σ est une suite infinie d'éléments de Σ. On note Σ^ω l'ensemble des mots infinis sur Σ.

Exemple : $u = ababbbbbbbb\ldots$

- $u(1) = a$,
- $u(2) = b$,
- $u(3) = a$,
- $u(4) = b$,
- $u(5) = b$,
- $u(i) = b$ pour $i \geq 4$.

Application : étude des flux de données.

Produit

On ne peut pas faire le produit de concaténation de deux mots infinis.

On peut concaténer un mot fini v avec un mot infini u.

Produit

On ne peut pas faire le produit de concaténation de deux mots infinis.

On peut concaténer un mot fini v avec un mot infini u.
$v = babba$
$u = ababbbbb\ldots$
$vu = babbaababbbbb\ldots$.

Produit

On ne peut pas faire le produit de concaténation de deux mots infinis.

On peut concaténer un mot fini v avec un mot infini u.
$v = babba$
$u = ababbbbb\ldots$
$vu = babbaababbbbb\ldots$.

Ce produit s'étend aux langages.

$$\{ab, ba, aa, bb\}^* \ \{bbbb\ldots\}$$

Notation ω

Soit L un langage de mots finis. On note L^ω l'ensemble des mots infinis obtenus par concaténation (infinie) de mot de L.

Exemples :

Notation ω

Soit L un langage de mots finis. On note L^ω l'ensemble des mots infinis obtenus par concaténation (infinie) de mot de L.

Exemples :

- $\{ab\}^\omega$ ne contient que le mot $ababababab\ldots$,

Notation ω

Soit L un langage de mots finis. On note L^ω l'ensemble des mots infinis obtenus par concaténation (infinie) de mot de L.

Exemples :

- $\{ab\}^\omega$ ne contient que le mot $ababababab\ldots$,
- $\{ab, b, c\}^\omega$: tous les mots sur $\{a, b, c\}$ où tout a est suivi d'un b.

Notation ω

Soit L un langage de mots finis. On note L^ω l'ensemble des mots infinis obtenus par concaténation (infinie) de mot de L.

Exemples :

- $\{ab\}^\omega$ ne contient que le mot $ababababab\ldots$,
- $\{ab, b, c\}^\omega$: tous les mots sur $\{a, b, c\}$ où tout a est suivi d'un b.
- $\{aa, b, c\}^\omega$: tous les mots sur $\{a, b, c\}$ où les blocs de a sont de longueur paire.

Notation ω

Soit L un langage de mots finis. On note L^ω l'ensemble des mots infinis obtenus par concaténation (infinie) de mot de L.

Exemples :

- $\{ab\}^\omega$ ne contient que le mot $ababababab\ldots$,
- $\{ab, b, c\}^\omega$: tous les mots sur $\{a, b, c\}$ où tout a est suivi d'un b.
- $\{aa, b, c\}^\omega$: tous les mots sur $\{a, b, c\}$ où les blocs de a sont de longueur paire.
- $(b^*a)^\omega$: tous les mots sur $\{a, b\}$, contenant un nombre infini de a.

Notation ω

Soit L un langage de mots finis. On note L^ω l'ensemble des mots infinis obtenus par concaténation (infinie) de mot de L.

Exemples :

- $\{ab\}^\omega$ ne contient que le mot $ababababab\ldots$,
- $\{ab, b, c\}^\omega$: tous les mots sur $\{a, b, c\}$ où tout a est suivi d'un b.
- $\{aa, b, c\}^\omega$: tous les mots sur $\{a, b, c\}$ où les blocs de a sont de longueur paire.
- $(b^*a)^\omega$: tous les mots sur $\{a, b\}$, contenant un nombre infini de a.

Si u est un mot **fini** non vide, alors u^ω désigne l'unique mot de $\{u\}^\omega$.

Exercices

Chercher les exercices 10 et 11 de la feuille de TD1.

LTL : syntaxe

On travaille sur un alphabet fini Σ. Une formule LTL est construite en utilisant (exemple : $\Sigma = \{a, b, c\}$) :

LTL : syntaxe

On travaille sur un alphabet fini Σ. Une formule LTL est construite en utilisant (exemple : $\Sigma = \{a, b, c\}$) :

- Les lettres de Σ (comme formules atomiques),
  exemple : a est une formule LTL

LTL : syntaxe

On travaille sur un alphabet fini Σ. Une formule LTL est construite en utilisant (exemple : $\Sigma = \{a, b, c\}$) :

- Les lettres de Σ (comme formules atomiques),
  exemple : a est une formule LTL
- Les opérateurs booléens classiques $\vee$, $\wedge$, $\neg$, $\Rightarrow$,$\Leftrightarrow$,
  exemple : $(a \vee b) \wedge \neg b$ est une formule LTL

LTL : syntaxe

On travaille sur un alphabet fini Σ. Une formule LTL est construite en utilisant (exemple : $\Sigma = \{a, b, c\}$) :

- Les lettres de Σ (comme formules atomiques),
  exemple : a est une formule LTL
- Les opérateurs booléens classiques $\vee$, $\wedge$, $\neg$, $\Rightarrow$,$\Leftrightarrow$,
  exemple : $(a \vee b) \wedge \neg b$ est une formule LTL
- L'opérateur temporel unaire $\circ$ (*next*).
  exemple : $(\circ a \vee b) \wedge \circ \neg \circ b$ est une formule LTL

LTL : syntaxe

On travaille sur un alphabet fini Σ. Une formule LTL est construite en utilisant (exemple : $\Sigma = \{a, b, c\}$) :

- Les lettres de Σ (comme formules atomiques),
  exemple : a est une formule LTL
- Les opérateurs booléens classiques $\vee$, $\wedge$, $\neg$, $\Rightarrow$,$\Leftrightarrow$,
  exemple : $(a \vee b) \wedge \neg b$ est une formule LTL
- L'opérateur temporel unaire $\circ$ (*next*).
  exemple : $(\circ a \vee b) \wedge \circ \neg \circ b$ est une formule LTL
- L'opérateur temporel binaire $\mathcal{U}$ (*until*).
  exemple : $(\circ a \vee b)\ \mathcal{U} b$ est une formule LTL.

LTL : syntaxe

On travaille sur un alphabet fini Σ. Une formule LTL est construite en utilisant (exemple : $\Sigma = \{a, b, c\}$) :

- Les lettres de Σ (comme formules atomiques),
  exemple : a est une formule LTL
- Les opérateurs booléens classiques $\vee$, $\wedge$, $\neg$, $\Rightarrow$,$\Leftrightarrow$,
  exemple : $(a \vee b) \wedge \neg b$ est une formule LTL
- L'opérateur temporel unaire $\circ$ (*next*).
  exemple : $(\circ a \vee b) \wedge \circ \neg \circ b$ est une formule LTL
- L'opérateur temporel binaire $\mathcal{U}$ (*until*).
  exemple : $(\circ a \vee b)\ \mathcal{U} b$ est une formule LTL.

$$\circ a\ \mathcal{U}(b \Rightarrow (a\ \mathcal{U} \neg c))$$

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

- $(u, i) \models a$, si $u(i) = a$, exemple : $(abacbbbb \ldots, 4) \models c$

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

- $(u, i) \models a$, si $u(i) = a$, exemple : $(abacbbbb\ldots, 4) \models c$
- $(u, i) \models \varphi_1 \wedge \varphi_2$, si $(u, i) \models \varphi_1$ et $(u, i) \models \varphi_2$ (idem pour les autres opérateurs booléens)

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

- $(u, i) \models a$, si $u(i) = a$, exemple : $(abacbbbb\ldots, 4) \models c$
- $(u, i) \models \varphi_1 \wedge \varphi_2$, si $(u, i) \models \varphi_1$ et $(u, i) \models \varphi_2$ (idem pour les autres opérateurs booléens)
- $(u, i) \models \circ\varphi$, si $(u, i + 1) \models \varphi$, exemple : $(abacbbbb\ldots, 3) \models \circ c$

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

- $(u, i) \models a$, si $u(i) = a$, exemple : $(abacbbbb\ldots, 4) \models c$
- $(u, i) \models \varphi_1 \wedge \varphi_2$, si $(u, i) \models \varphi_1$ et $(u, i) \models \varphi_2$ (idem pour les autres opérateurs booléens)
- $(u, i) \models \circ\varphi$, si $(u, i + 1) \models \varphi$, exemple : $(abacbbbb\ldots, 3) \models \circ c$
- $(u, i) \models \varphi_1 \mathcal{U} \varphi_2$, s'il existe $j \geq 0$ tel que

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

- $(u, i) \models a$, si $u(i) = a$, exemple : $(abacbbbb\ldots, 4) \models c$
- $(u, i) \models \varphi_1 \wedge \varphi_2$, si $(u, i) \models \varphi_1$ et $(u, i) \models \varphi_2$ (idem pour les autres opérateurs booléens)
- $(u, i) \models \circ\varphi$, si $(u, i + 1) \models \varphi$, exemple : $(abacbbbb\ldots, 3) \models \circ c$
- $(u, i) \models \varphi_1 \mathcal{U} \varphi_2$, s'il existe $j \geq 0$ tel que
  - $(u, i + j) \models \varphi_2$,
    (*maintenant* ($j = 0$) *ou plus loin dans le mot* ($j > 0$), φ_2 *sera satisfaite*),

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

- $(u, i) \models a$, si $u(i) = a$, exemple : $(abacbbbb\ldots, 4) \models c$
- $(u, i) \models \varphi_1 \wedge \varphi_2$, si $(u, i) \models \varphi_1$ et $(u, i) \models \varphi_2$ (idem pour les autres opérateurs booléens)
- $(u, i) \models \circ\varphi$, si $(u, i + 1) \models \varphi$, exemple : $(abacbbbb\ldots, 3) \models \circ c$
- $(u, i) \models \varphi_1 \mathcal{U} \varphi_2$, s'il existe $j \geq 0$ tel que
  - $(u, i + j) \models \varphi_2$,
    (*maintenant ($j = 0$) ou plus loin dans le mot ($j > 0$), φ_2 sera satisfaite*),
  - pour tout $0 \leq k < j$, $(u, i + k) \models \varphi_1$,
    (*tant qu'on n'y est pas, φ_1 est satisfaite*),

Sémantique

On considère une formule LTL φ sur Σ. On dit qu'un mot infini $u \in \Sigma^\omega$ satisfait φ à la position i ($i \in \mathbb{N}^*$), noté $(u, i) \models \varphi$ quand

- $(u, i) \models a$, si $u(i) = a$, exemple : $(abacbbbb\ldots, 4) \models c$
- $(u, i) \models \varphi_1 \wedge \varphi_2$, si $(u, i) \models \varphi_1$ et $(u, i) \models \varphi_2$ (idem pour les autres opérateurs booléens)
- $(u, i) \models \circ\varphi$, si $(u, i + 1) \models \varphi$, exemple : $(abacbbbb\ldots, 3) \models \circ c$
- $(u, i) \models \varphi_1 \mathcal{U} \varphi_2$, s'il existe $j \geq 0$ tel que
  - $(u, i + j) \models \varphi_2$,
    (*maintenant* ($j = 0$) *ou plus loin dans le mot* ($j > 0$), φ_2 *sera satisfaite*),
  - pour tout $0 \leq k < j$, $(u, i + k) \models \varphi_1$,
    (*tant qu'on n'y est pas,* φ_1 *est satisfaite*),

  $(aaacbbbb\ldots, 1) \models a\ \mathcal{U} c$

Illustration graphique

position	1	2	...	i	i+1	i+2	...	i+j-1	i+j	i+j+1	...
u=	a	a	...	a	b	b		b	c	b	...

$$(u, i) \models \varphi \ ?$$

Illustration graphique

position	1	2	...	i	i+1	i+2	...	i+j-1	i+j	i+j+1	...
u=	a	a	...	a	b	b		b	c	b	...
				φ							

$$(u, i) \models \varphi \ ?$$

Illustration graphique

position	1	2	...	i	i+1	i+2	...	i+j-1	i+j	i+j+1	...
u=	a	a	...	a	b	b		b	c	b	...
				φ		φ_1					

$$(u, i) \models \varphi \ ?$$

$$\varphi = \circ \circ \varphi_1$$

Illustration graphique

position	1	2	...	i	i+1	i+2	...	i+j-1	i+j	i+j+1	...
u=	a	a	...	a	b	b		b	c	b	...
				φ_1	φ_1	φ_1	...	φ_1	φ_2	??	...

$$(u, i) \models \varphi \ ?$$

$$\varphi = \varphi_1 \mathcal{U} \varphi_2$$

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

$(u, 1) \models a$

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

$(u, 1) \models a$

$(u, 4) \models c$

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

$(u, 1) \models a$

$(u, 4) \models c$

$(u, 2) \models \neg c$

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

$(u, 1) \models a$

$(u, 4) \models c$

$(u, 2) \models \neg c$

$(u, 2) \models a \wedge \circ\neg c$

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

$(u, 1) \models a$

$(u, 4) \models c$

$(u, 2) \models \neg c$

$(u, 2) \models a \wedge \circ\neg c$

$(u, 1) \models a \mathcal{U} b$ avec $j = 2$

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

$(u, 1) \models a$

$(u, 4) \models c$

$(u, 2) \models \neg c$

$(u, 2) \models a \wedge \circ\neg c$

$(u, 1) \models a \mathcal{U} b$ avec $j = 2$

$(u, 1) \not\models a \mathcal{U} c$

Exemples

	1	2	3	4	5	6	7	8	9	...
u=	a	a	b	c	a	b	b	a	b	...

$(u, 1) \models a$

$(u, 4) \models c$

$(u, 2) \models \neg c$

$(u, 2) \models a \wedge \circ\neg c$

$(u, 1) \models a \mathcal{U} b$ avec $j = 2$

$(u, 1) \not\models a \mathcal{U} c$

$(u, 3) \models a \mathcal{U} b$ avec $j = 0$

Sémantique (suite)

Définition

On dit qu'un mot u satisfait une formule φ, noté $u \models \varphi$ si $(u, 1) \models \varphi$.

Sémantique (suite)

Définition

On dit qu'un mot u satisfait une formule φ, noté $u \models \varphi$ si $(u, 1) \models \varphi$.

Notation

On note L_φ l'ensemble des mots infinis qui satisfont φ.

Sémantique (suite)

Définition

On dit qu'un mot u satisfait une formule φ, noté $u \models \varphi$ si $(u, 1) \models \varphi$.

Notation

On note L_φ l'ensemble des mots infinis qui satisfont φ.

Exemples

- L_a : mots commençant par a,

Sémantique (suite)

Définition

On dit qu'un mot u satisfait une formule φ, noté $u \models \varphi$ si $(u, 1) \models \varphi$.

Notation

On note L_φ l'ensemble des mots infinis qui satisfont φ.

Exemples

- L_a : mots commençant par a,
- $L_{\neg c}$: mots ne commençant pas par c,

Sémantique (suite)

Définition

On dit qu'un mot u satisfait une formule φ, noté $u \models \varphi$ si $(u, 1) \models \varphi$.

Notation

On note L_φ l'ensemble des mots infinis qui satisfont φ.

Exemples

- L_a : mots commençant par a,
- $L_{\neg c}$: mots ne commençant pas par c,
- $L_{\circ c}$: mots dont la seconde lettre est c,

Sémantique (suite)

Définition

On dit qu'un mot u satisfait une formule φ, noté $u \models \varphi$ si $(u, 1) \models \varphi$.

Notation

On note L_φ l'ensemble des mots infinis qui satisfont φ.

Exemples

- L_a : mots commençant par a,
- $L_{\neg c}$: mots ne commençant pas par c,
- $L_{\circ c}$: mots dont la seconde lettre est c,
- $L_{a\mathcal{U}c}$: mots ayant un suffixe dans a^*b.

Opérateurs

On travaille sur l'alphabet $\Sigma = \{a_1, \ldots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \ldots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top\mathcal{U}\varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

Opérateurs

On travaille sur l'alphabet $\Sigma = \{a_1, \ldots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \ldots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top\mathcal{U}\varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

- $L_{\Box a}$: mots ne contenant que des a,

Opérateurs

On travaille sur l'alphabet $\Sigma = \{a_1, \ldots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \ldots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top \mathcal{U} \varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

- $L_{\Box a}$: mots ne contenant que des a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,

Opérateurs

On travaille sur l'alphabet $\Sigma = \{a_1, \ldots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \ldots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top \mathcal{U} \varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

- $L_{\Box a}$: mots ne contenant que des a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,

Opérateurs

On travaille sur l'alphabet $\Sigma = \{a_1, \ldots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \ldots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top \mathcal{U} \varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

- $L_{\Box a}$: mots ne contenant que des a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\diamond a}$: mots contenant au moins un a,

Opérateurs

On travaille sur l'alphabet $\Sigma = \{a_1, \ldots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \ldots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top \mathcal{U} \varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

- $L_{\Box a}$: mots ne contenant que des a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\diamond a}$: mots contenant au moins un a,
- $L_{\diamond\neg a}$: mots contenant au moins une lettre qui n'est pas a,

Opérateurs

On travaille sur l'alphabet $\Sigma = \{a_1, \ldots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \ldots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top \mathcal{U} \varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

- $L_{\Box a}$: mots ne contenant que des a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\diamond a}$: mots contenant au moins un a,
- $L_{\diamond\neg a}$: mots contenant au moins une lettre qui n'est pas a,
- $L_{\diamond\Box a}$: mots finissant par a^ω,

Opérateurs

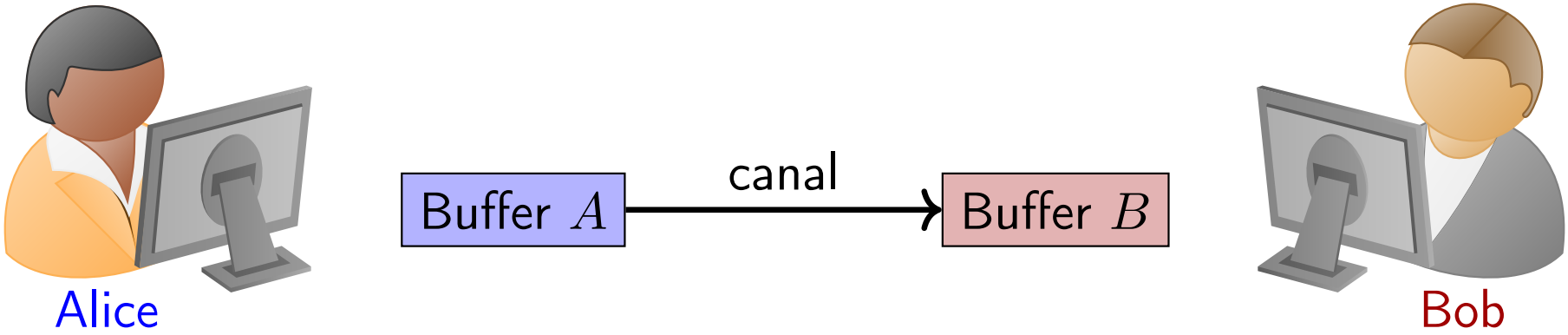
On travaille sur l'alphabet $\Sigma = \{a_1, \dots a_k\}$.

Définitions

- $\top = a_1 \vee a_2 \vee \dots \vee a_k$ (vrai), et $\bot = \neg\top$ (faux),
- $\diamond\varphi = \top \mathcal{U} \varphi$ (un jour dans le futur),
- $\Box\varphi = \neg \diamond \neg\varphi$ (toujours).

- $L_{\Box a}$: mots ne contenant que des a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\Box\neg a}$: mots ne contenant aucun a,
- $L_{\diamond a}$: mots contenant au moins un a,
- $L_{\diamond\neg a}$: mots contenant au moins une lettre qui n'est pas a,
- $L_{\diamond\Box a}$: mots finissant par a^ω,
- $L_{\Box\diamond a}$: mots contenant un nombre infini de a.

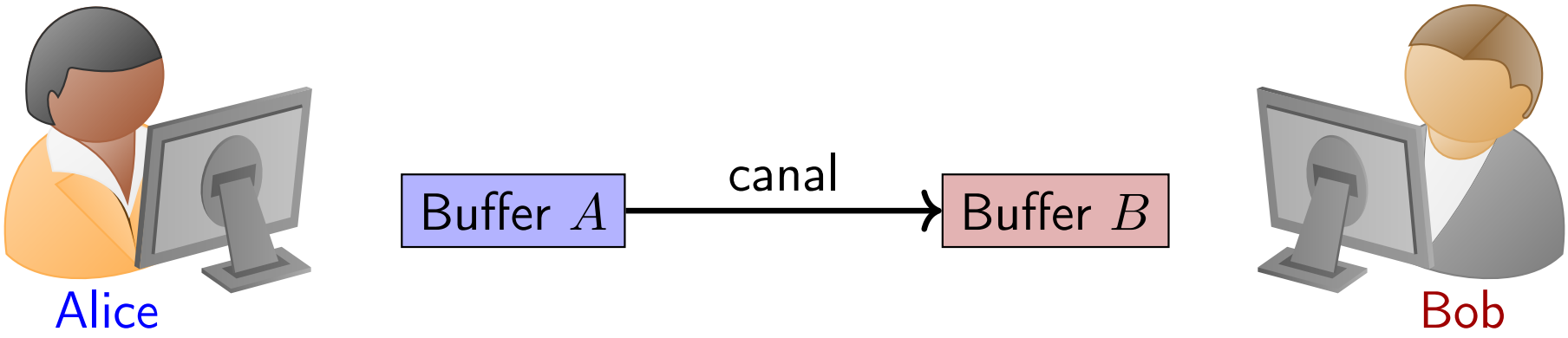
LTL : un exemple de canal de communication [2]



- Alice envoie des messages à Bob par un canal unidirectionnel.
- Leurs messages sont placés dans des buffers respectifs de capacités non bornées.
- A chaque étape, on sait quels messages sont dans chaque buffer, parmi un ensemble $\Sigma = \{m_1, \ldots, m_k\}$ de messages possibles.
- Une configuration du système est un couple de $2^{\Sigma} \times 2^{\Sigma}$.

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

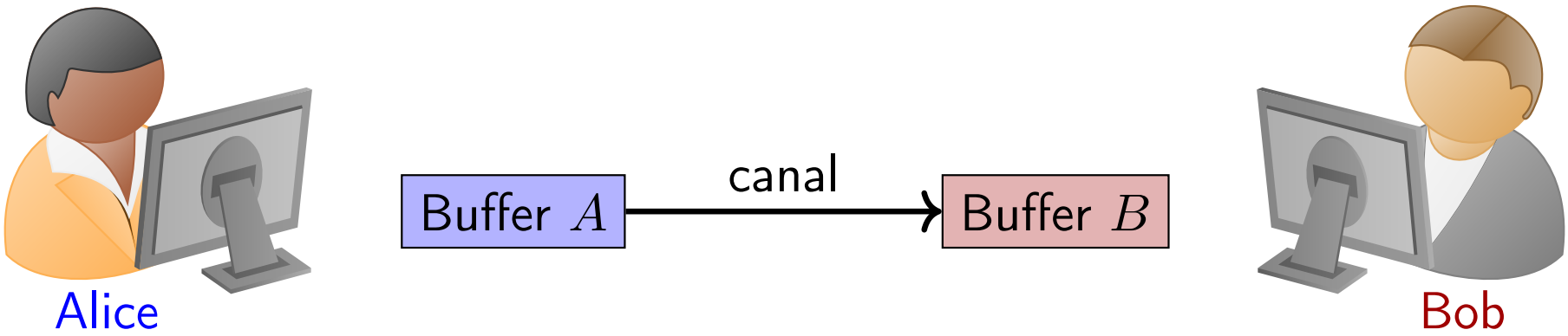
LTL : un exemple de canal de communication [2]



$(\emptyset, \emptyset)$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

LTL : un exemple de canal de communication [2]

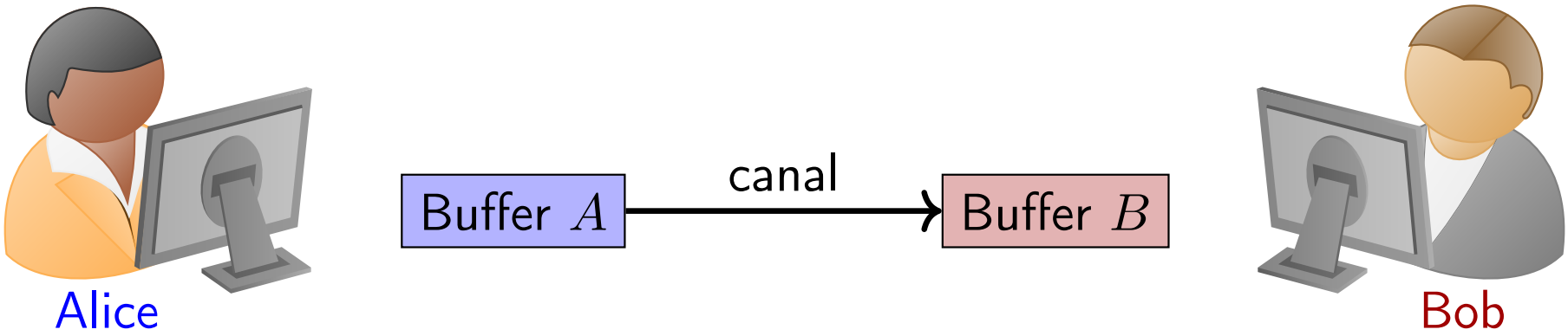


m_1

$(\emptyset, \emptyset) \rightarrow (\{m_1\}, \emptyset)$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

LTL : un exemple de canal de communication [2]

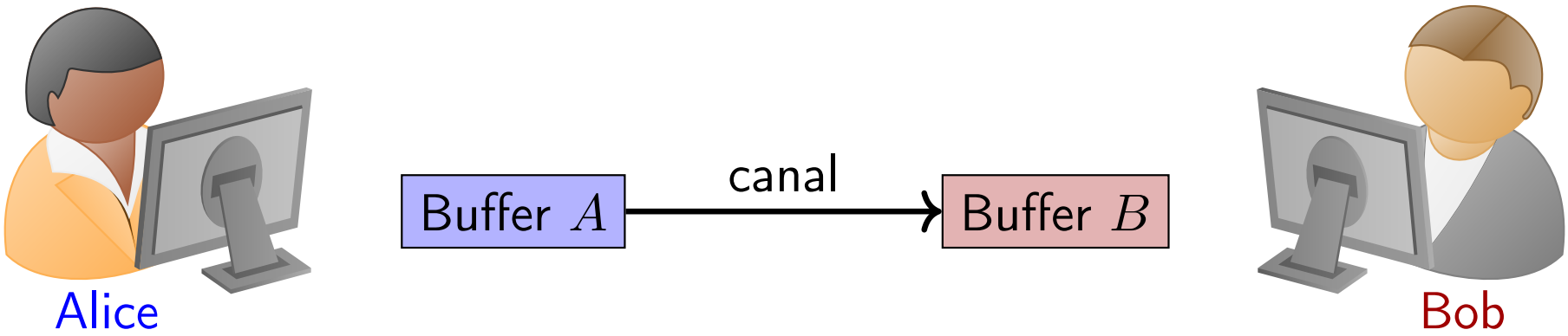


m_1, m_2

$(\emptyset, \emptyset) \rightarrow (\{m_1\}, \emptyset) \rightarrow (\{m_1, m_2\}, \emptyset)$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

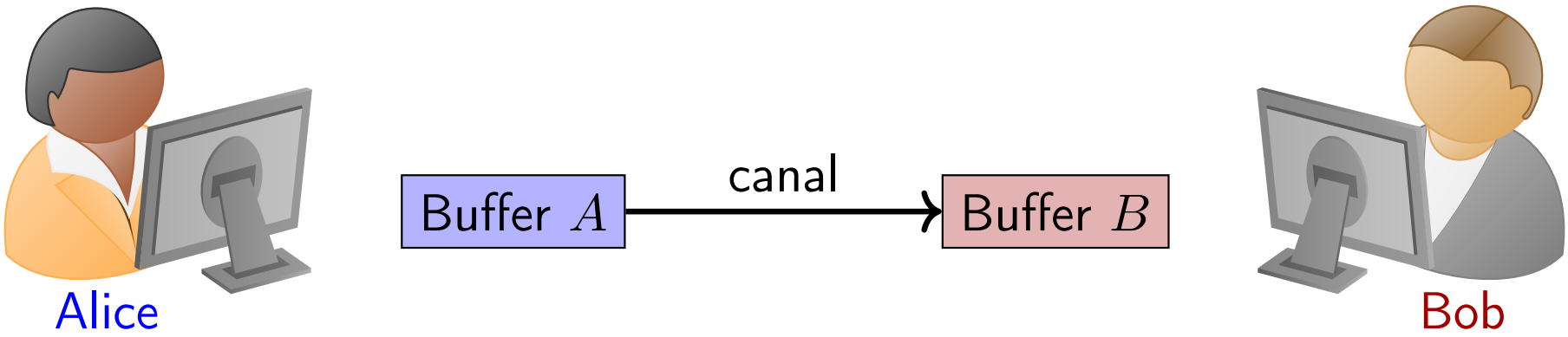
LTL : un exemple de canal de communication [2]



$$(\emptyset, \emptyset) \rightarrow (\{m_1\}, \emptyset) \rightarrow (\{m_1, m_2\}, \emptyset) \rightarrow (\{m_2\}, \{m_1\})$$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

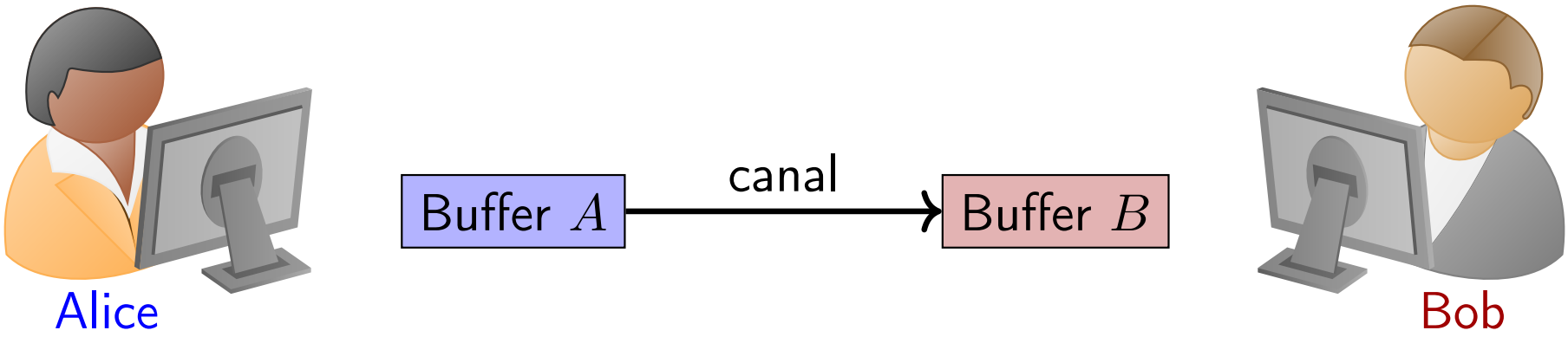
LTL : un exemple de canal de communication [2]



$$(\emptyset, \emptyset) \rightarrow (\{m_1\}, \emptyset) \rightarrow (\{m_1, m_2\}, \emptyset) \rightarrow (\{m_2\}, \{m_1\}) \rightarrow (\{m_2\}, \emptyset) \rightarrow \ldots$$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

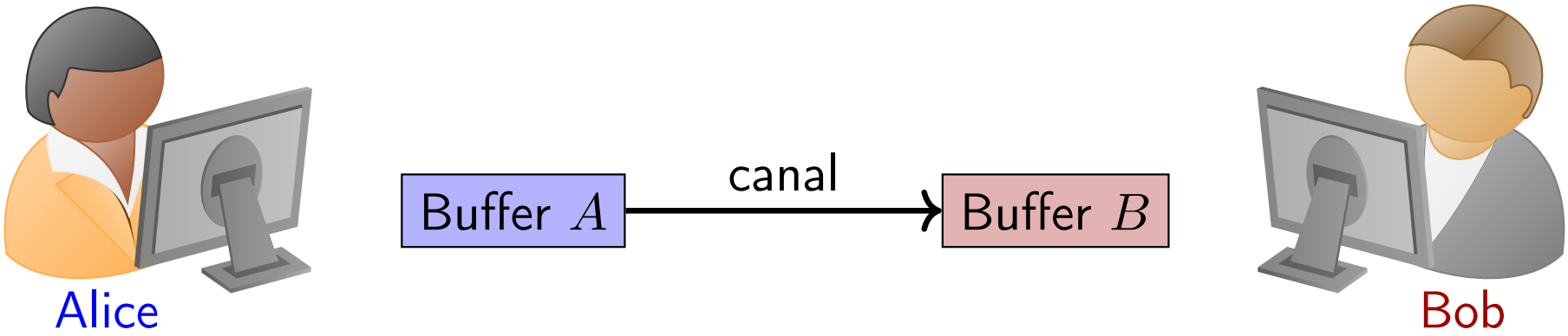
LTL : un exemple de canal de communication [2]



On note par $m \in A$ le fait que le message m soit dans la première composante et dualement $m \in B$ pour la seconde :
Si la configuration courante est $(\{m_2, m_3\}, \{m_1\})$, alors à cet instant $m_2 \in A$ est vraie, de même que $m_1 \in B$, mais pas $m_2 \in B$.

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

LTL : un exemple de canal de communication [2]

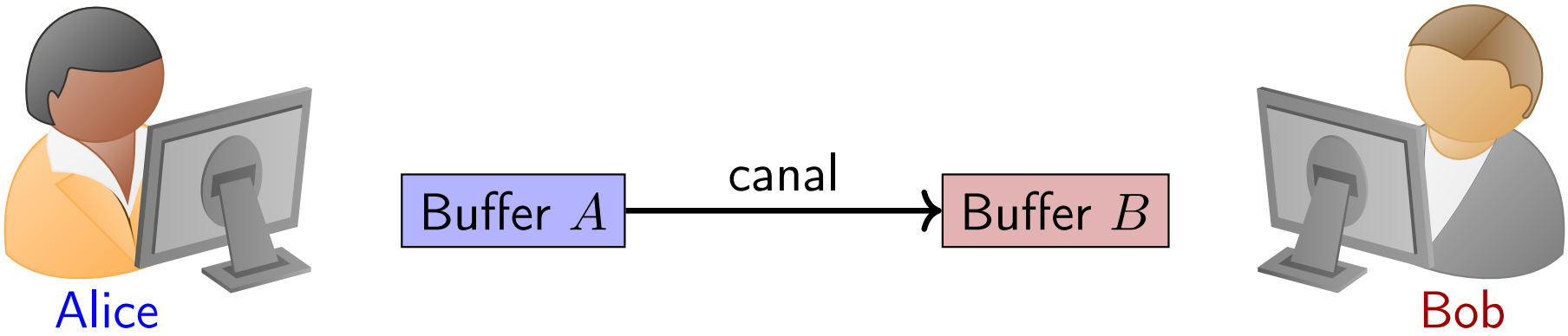


Le canal est sûr et tous les messages sont envoyés :

$$\Box(m \in A \Rightarrow \diamond\, m \in B)$$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

LTL : un exemple de canal de communication [2]

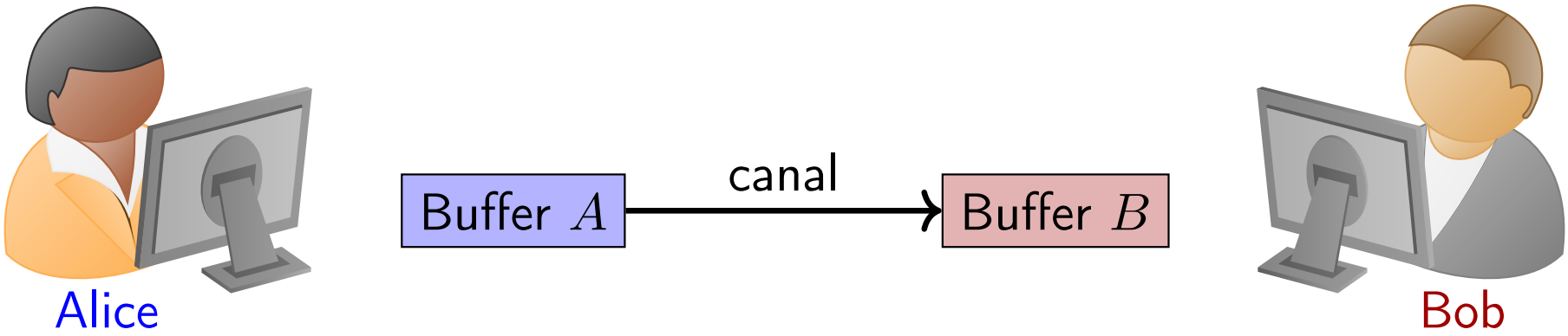


Un même message ne peut pas être dans les deux buffers à la fois :

$$\Box(\neg(m \in A \wedge m \in B))$$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

LTL : un exemple de canal de communication [2]

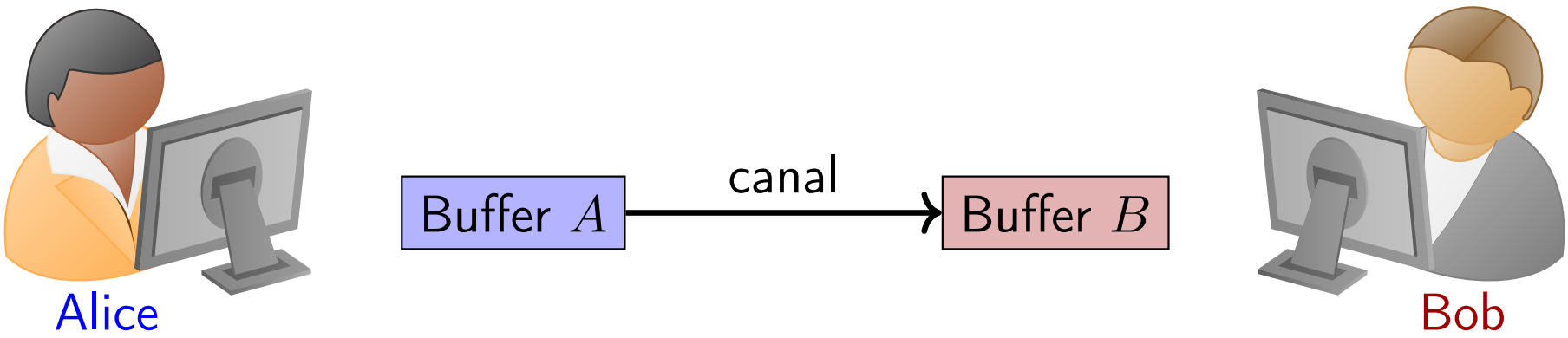


Aucun message ne disparaît :

$$\Box(m \in A \Rightarrow (m \in A \,\mathcal{U}\, m \in B))$$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

LTL : un exemple de canal de communication [2]



L'ordre des messages est respecté :

$$\Box\Big(m \in A \wedge \neg m' \in A \wedge \Diamond(m' \in A) \Rightarrow \Diamond\Big(m \in B \wedge \neg m' \in B \wedge \Diamond(m' \in B)\Big)\Big)$$

2. Ch. Baier and J.-P. Katoen, Principles of Model Checking, MIT Press 2008

Exercices

Chercher les exercices 12 à 14 de la feuille de TD1.

Plan

Un système (mots infinis) satisfait-il une propriété LTL ?

modèle du système

$\mathcal{A}$

propriété LTL

φ

Un système (mots infinis) satisfait-il une propriété LTL ?

modèle du système

$\mathcal{A}$

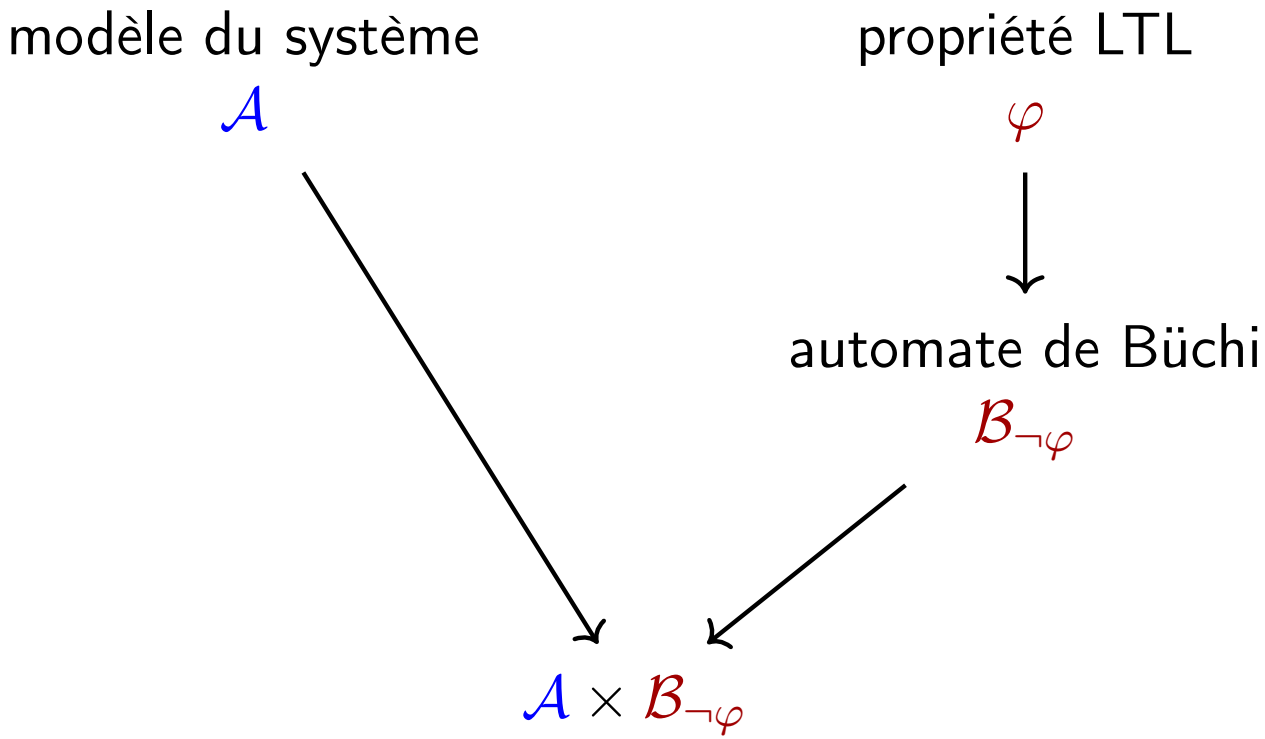
propriété LTL

φ

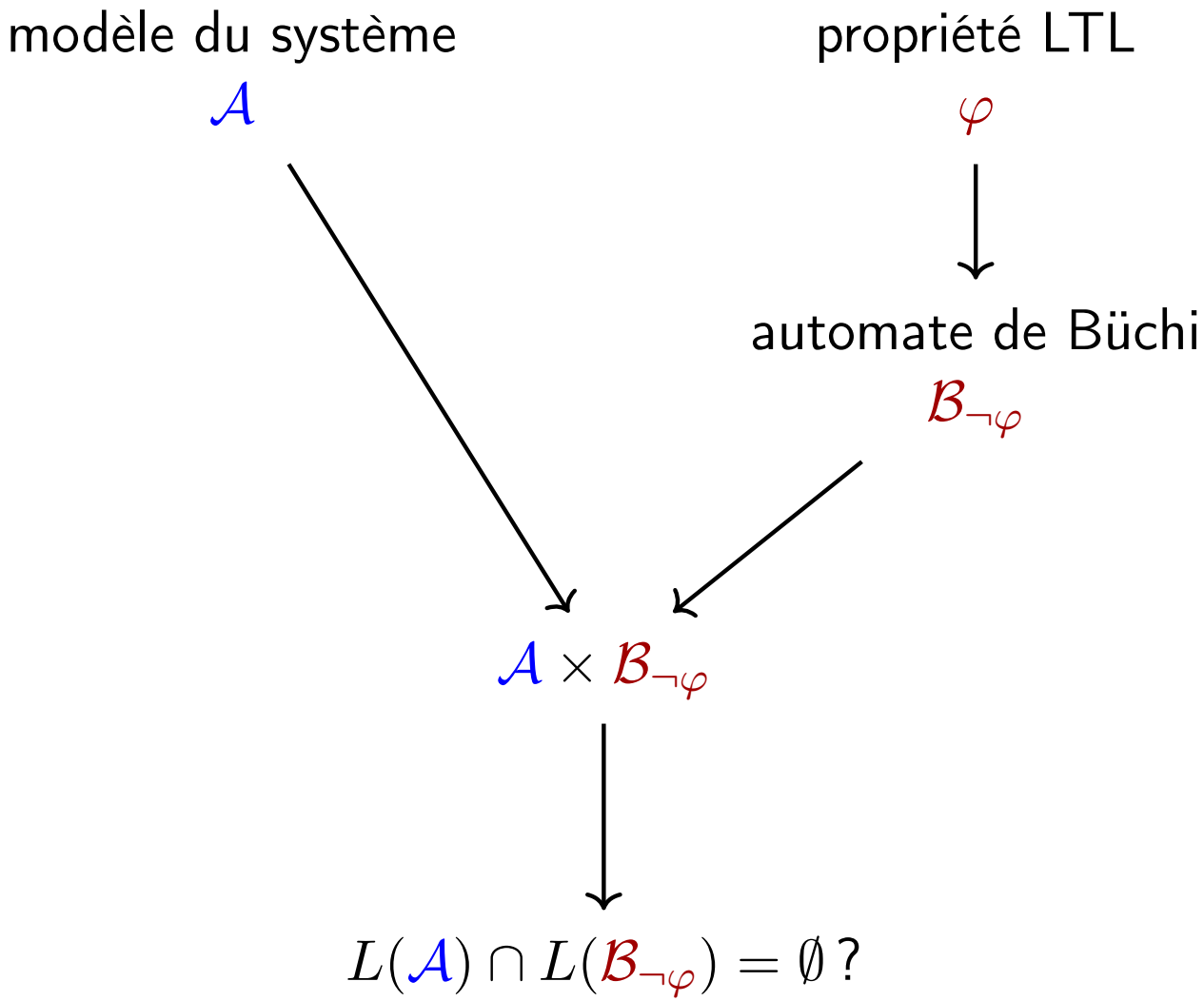
automate de Büchi

$\mathcal{B}_{\neg\varphi}$

Un système (mots infinis) satisfait-il une propriété LTL ?



Un système (mots infinis) satisfait-il une propriété LTL ?



Reconnaître des mots infinis : automates de Büchi

La définition d'un automate de Büchi est exactement la même que celle d'un automate fini classique.

Mots reconnus

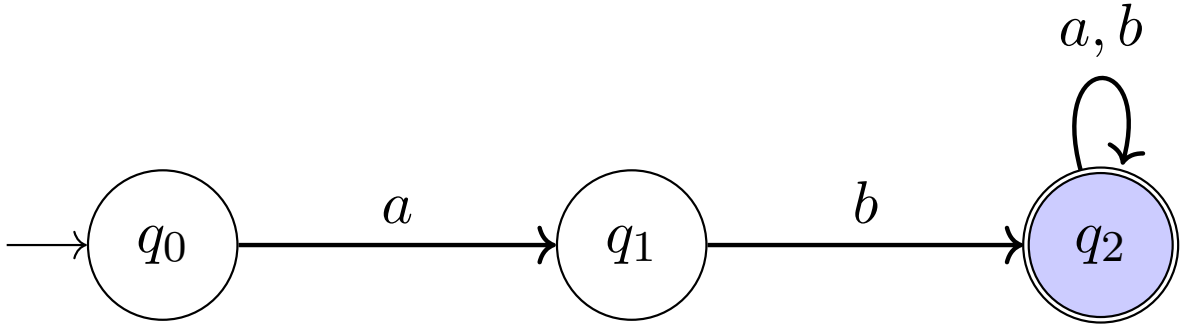
Un mots infini w est reconnu par un automate de Büchi $\mathcal{A}$ s'il existe un chemin inifni dans $\mathcal{A}$ partant d'un état initial, d'étiquette w et passant une infinité de fois par un état final.

Reconnaître des mots infinis : automates de Büchi

La définition d'un automate de Büchi est exactement la même que celle d'un automate fini classique.

Mots reconnus

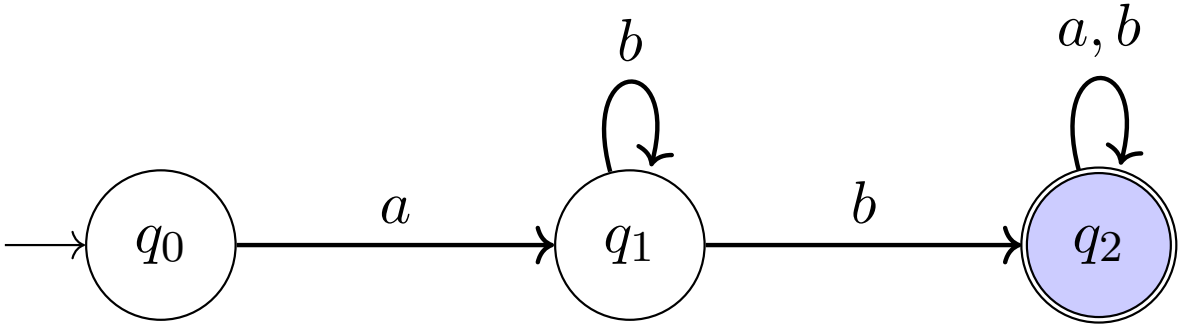
Un mots infini w est reconnu par un automate de Büchi $\mathcal{A}$ s'il existe un chemin inifni dans $\mathcal{A}$ partant d'un état initial, d'étiquette w et passant une infinité de fois par un état final.



Reconnaît l'ensemble des mots infinis commençant par ab.

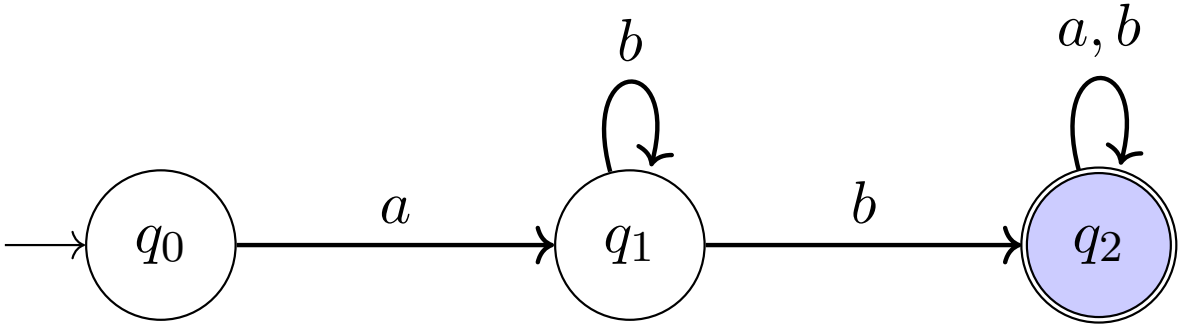
Exemple d'automates de Büchi

- Ensemble des mots infinis ayant un préfixe dans abb^* :

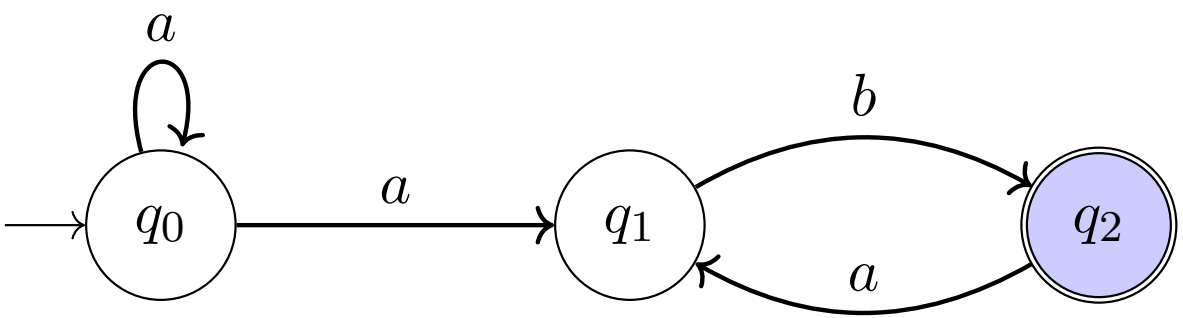


Exemple d'automates de Büchi

- Ensemble des mots infinis ayant un préfixe dans abb^* :

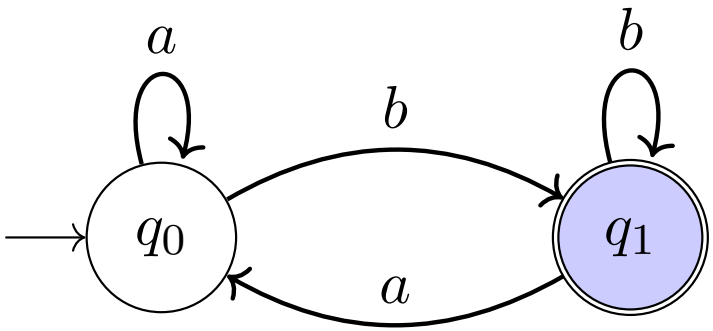


- $a^*a(ba)^\omega$:



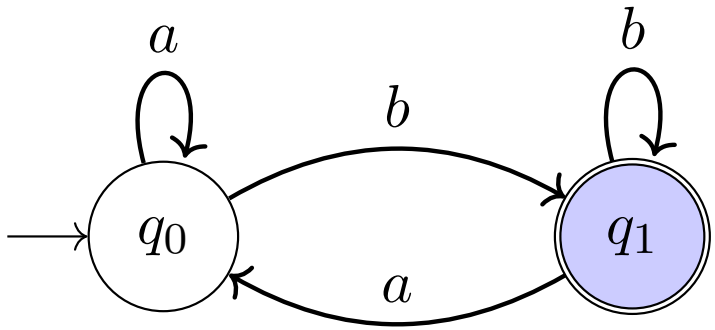
Exemple d'automates de Büchi (suite)

- Ensemble des mots infinis satsifaisant $\Box \diamond b$ (contenant une infinité de b) :

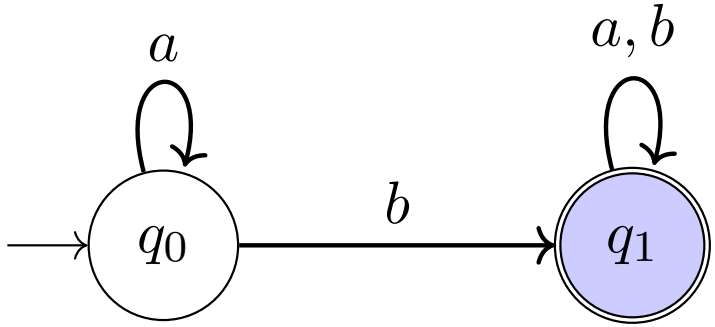


Exemple d'automates de Büchi (suite)

- Ensemble des mots infinis satsifaisant $\square \diamond b$ (contenant une infinité de b) :

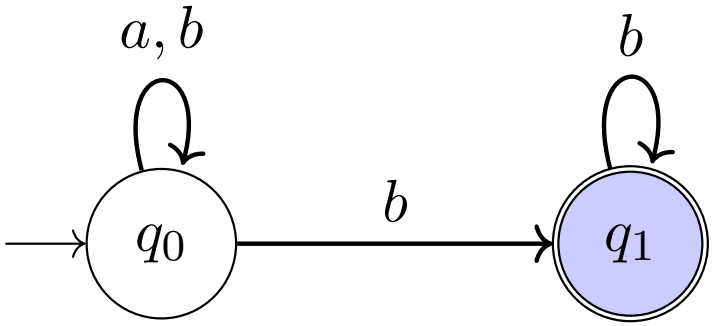


- $\diamond b$ (mot contenant au moins un b) :



Propriété des automates de Büchi

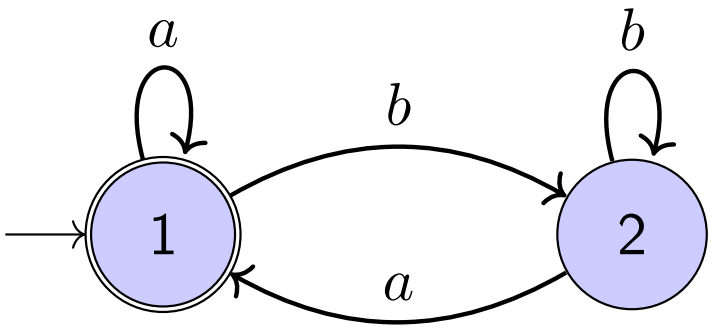
- La classe des langages reconnus par automates de Büchi est close par union, intersection et complément.
- Pour toute formule LTL, l'ensemble des mots qui la satisfait est reconnaissable par un automate de Büchi.
- Les automates de Büchi déterministes ont un pouvoir d'expression strictement plus faible que les automates de Büchi (les automates de Büchi ne sont pas déterminisiables). Exemple :

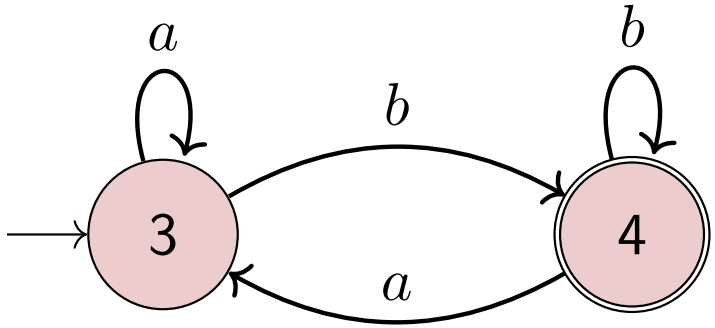


Produit et intersection de langages de mots infinis

Proposition

La classe des langages reconnaissables par automates de Büchi est close par intersection.

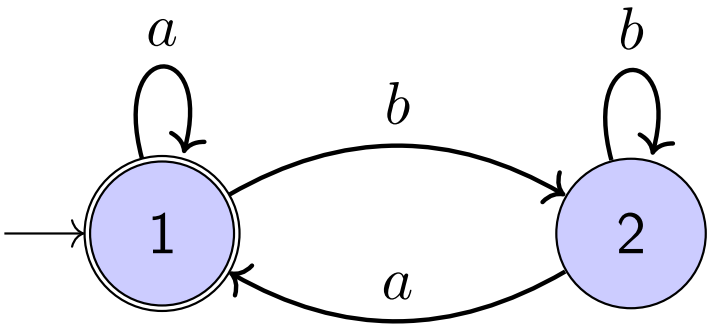


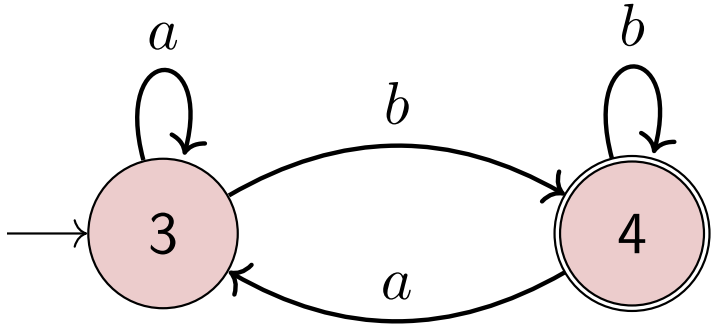


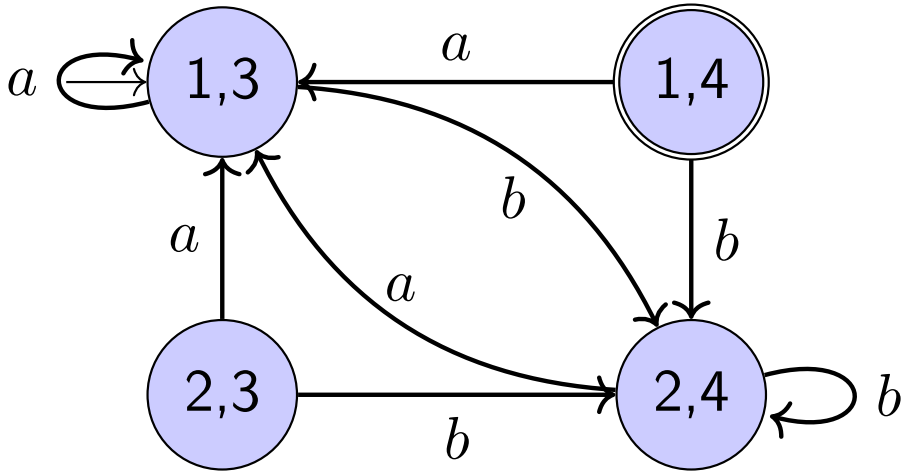
Produit et intersection de langages de mots infinis

Proposition

La classe des langages reconnaissables par automates de Büchi est close par intersection.



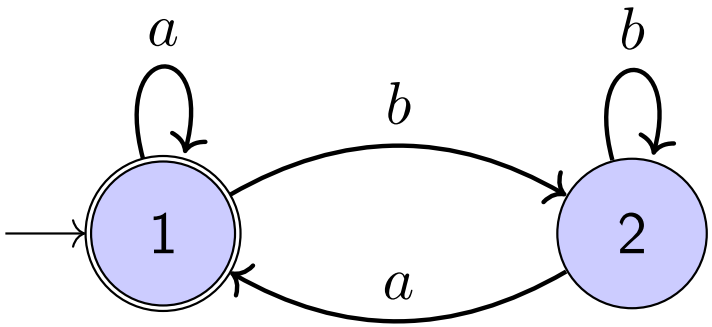


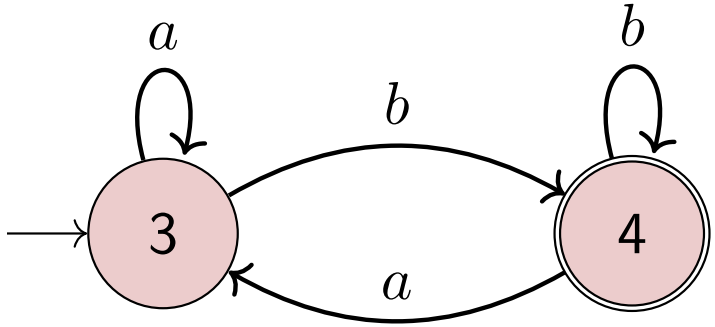


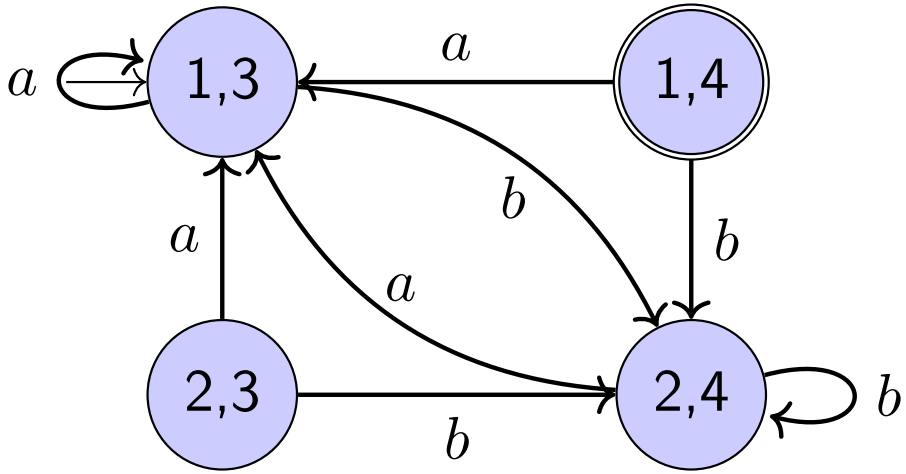
Produit et intersection de langages de mots infinis

Proposition

La classe des langages reconnaissables par automates de Büchi est close par intersection.





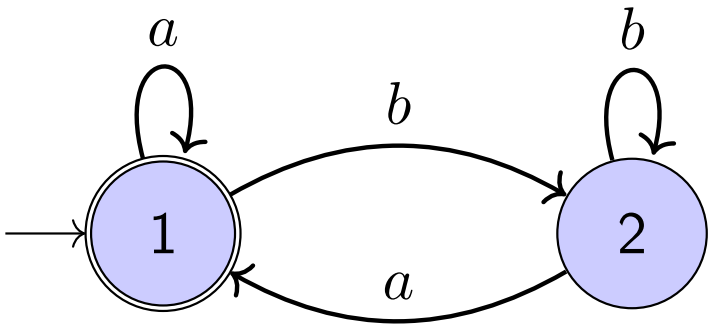


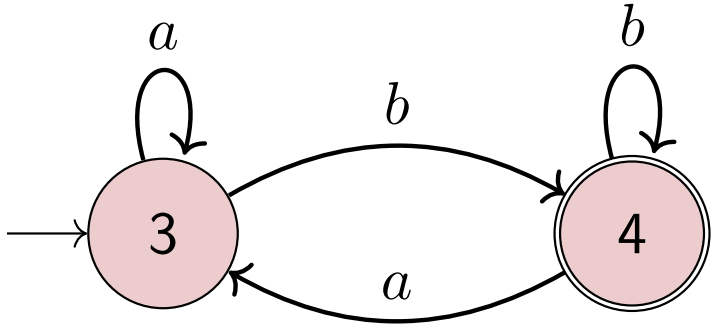
$bababababa\ldots$

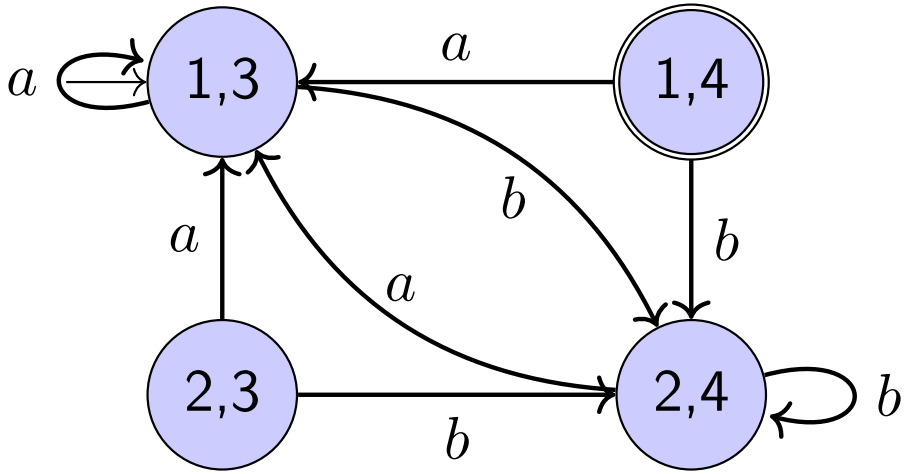
Produit et intersection de langages de mots infinis

Proposition

La classe des langages reconnaissables par automates de Büchi est close par intersection.







$bababababa...$

$(1,3)(2,4)(1,3)(2,4)(1,3)(2,4)...$

Construction

Idée de la construction

Les états sont de la forme $(p, p', \bot)$ ou $(p, p', \top)$. Le booléen code si l'on est passé par un état final du premier automate (p) depuis qu'on est passé la dernière fois par un état final du second (p').

- dont les états initiaux sont $I \times I' \times \{\top\}$,
- dont les états finaux sont $Q \times F' \times \{\top\}$,
- dont les transitions sont les triplets

$$
\begin{array}{l}
(p, p', \top), a, (q, q', \top) \text{ si } (p, a, q) \in E \text{ et } (p', a, q') \in E' \text{ et } p \in F \\
(p, p', \bot), a, (q, q', \top) \text{ si } (p, a, q) \in E \text{ et } (p', a, q') \in E' \text{ et } p \in F \\
(p, p', \top), a, (q, q', \bot) \text{ si } (p, a, q) \in E \text{ et } (p', a, q') \in E' \text{ et } p \notin F \text{ et } p' \in F' \\
(p, p', \bot), a, (q, q', \bot) \text{ si } (p, a, q) \in E \text{ et } (p', a, q') \in E' \text{ et } p \notin F \text{ et } p' \in F' \\
(p, p', \top), a, (q, q', \top) \text{ si } (p, a, q) \in E \text{ et } (p', a, q') \in E' \text{ et } p \notin F \text{ et } p' \notin F' \\
(p, p', \bot), a, (q, q', \bot) \text{ si } (p, a, q) \in E \text{ et } (p', a, q') \in E' \text{ et } p \notin F \text{ et } p' \notin F'
\end{array}
$$

Construction

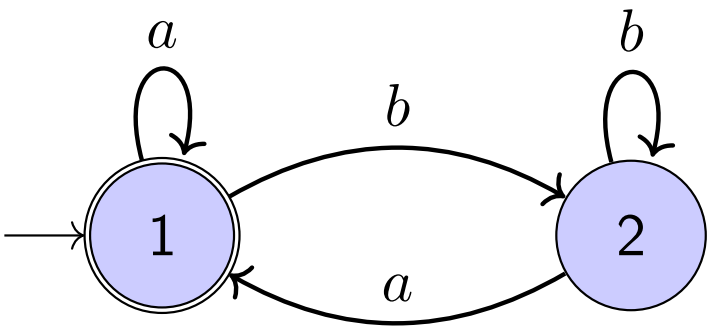
Idée de la construction

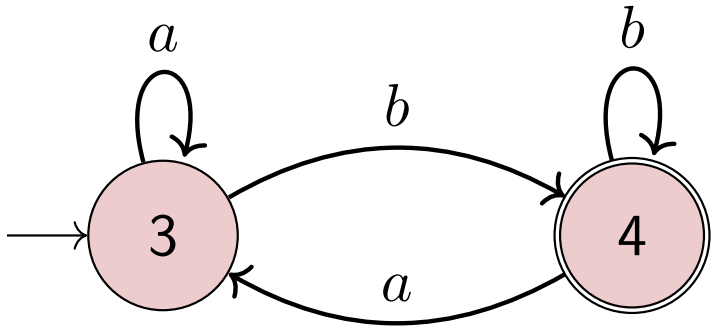
Les états sont de la forme $(p, p', \bot)$ ou $(p, p', \top)$. Le booléen code si l'on est passé par un état final du premier automate (p) depuis qu'on est passé la dernière fois par un état final du second (p').

- dont les états initiaux sont $I \times I' \times \{\top\}$,
- dont les états finaux sont $Q \times F' \times \{\top\}$,
- dont les transitions sont les triplets

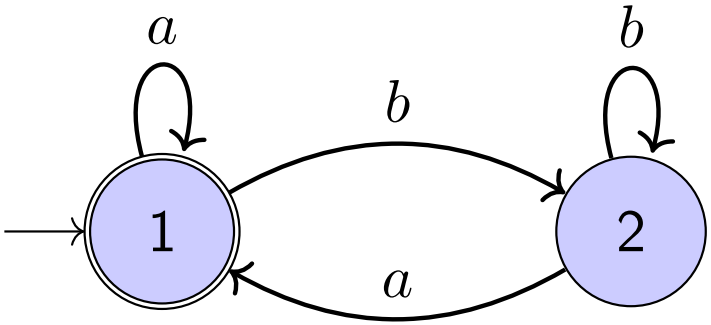
$$
\begin{array}{ll}
(p,p',\top),a,(q,q',\top) & \text{si } (p,a,q) \in E \text{ et } (p',a,q') \in E' \text{ et } p \in F \\
(p,p',\bot),a,(q,q',\top) & \text{si } (p,a,q) \in E \text{ et } (p',a,q') \in E' \text{ et } p \in F \\
(p,p',\top),a,(q,q',\bot) & \text{si } (p,a,q) \in E \text{ et } (p',a,q') \in E' \text{ et } p \notin F \text{ et } p' \in F' \\
(p,p',\bot),a,(q,q',\bot) & \text{si } (p,a,q) \in E \text{ et } (p',a,q') \in E' \text{ et } p \notin F \text{ et } p' \in F' \\
(p,p',\top),a,(q,q',\top) & \text{si } (p,a,q) \in E \text{ et } (p',a,q') \in E' \text{ et } p \notin F \text{ et } p' \notin F' \\
(p,p',\bot),a,(q,q',\bot) & \text{si } (p,a,q) \in E \text{ et } (p',a,q') \in E' \text{ et } p \notin F \text{ et } p' \notin F'
\end{array}
$$

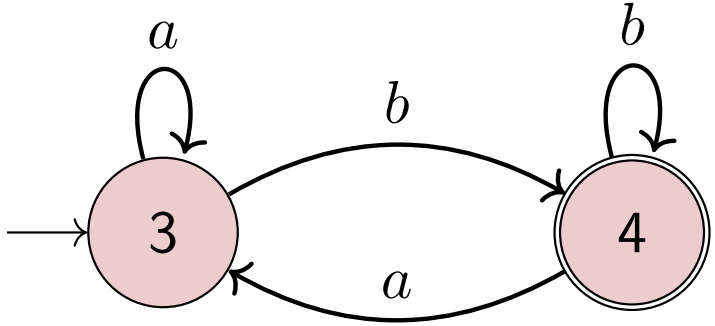
Exemple

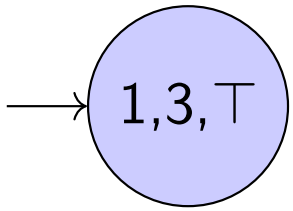




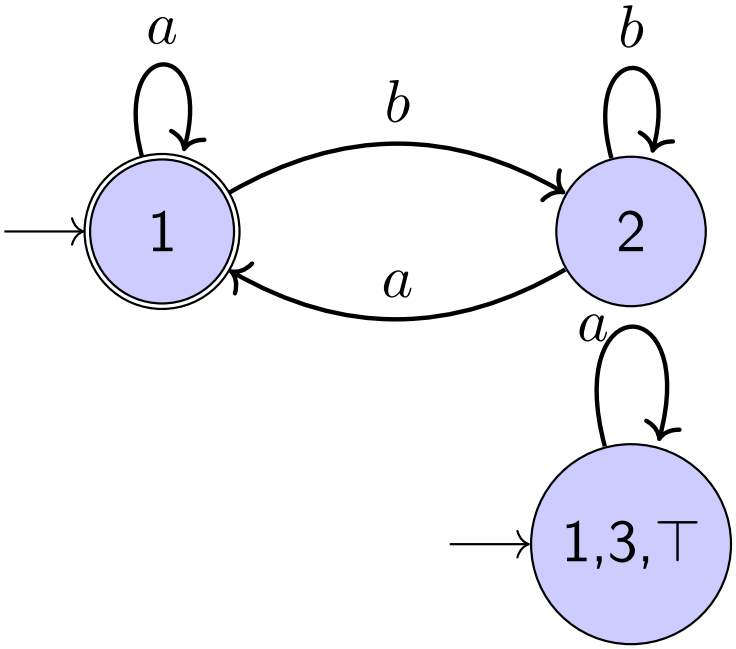
Exemple

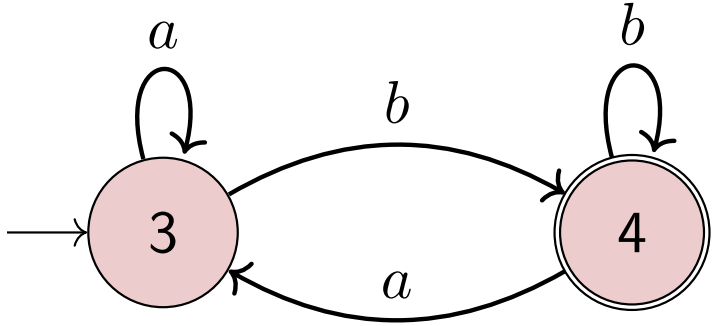




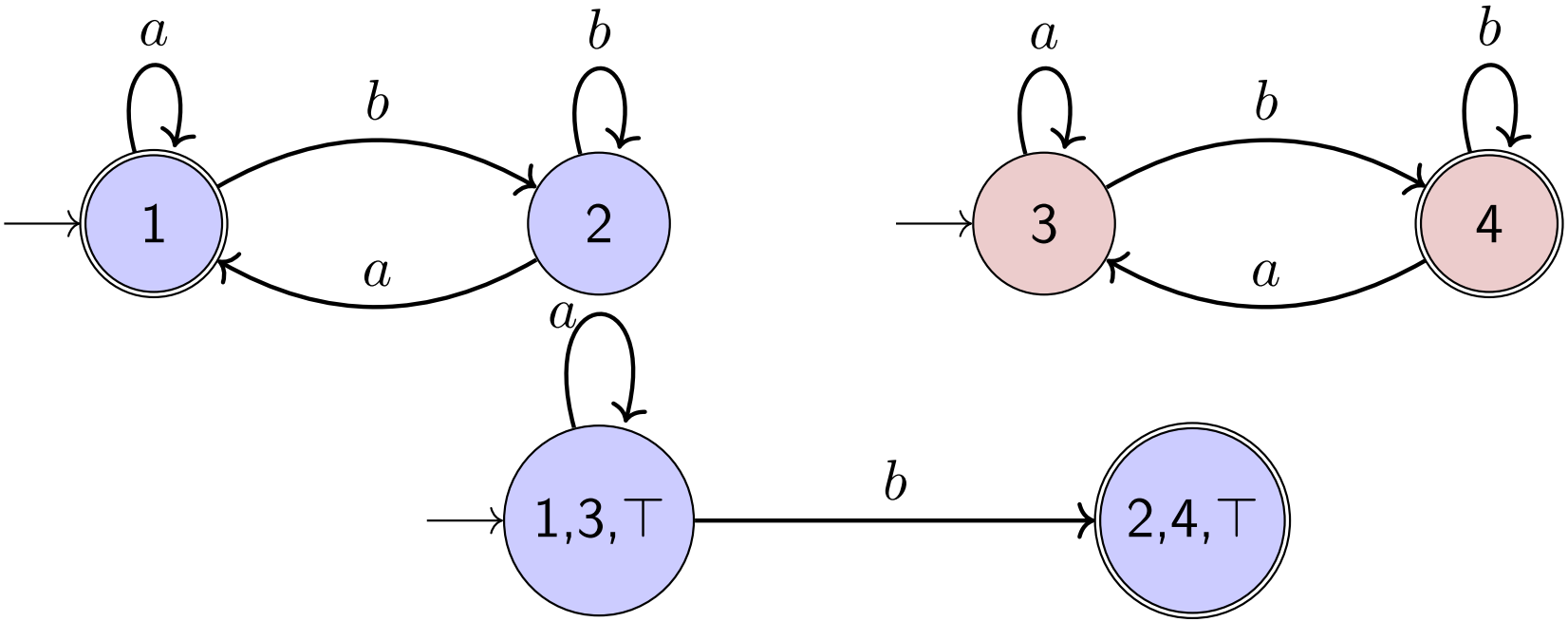


Exemple

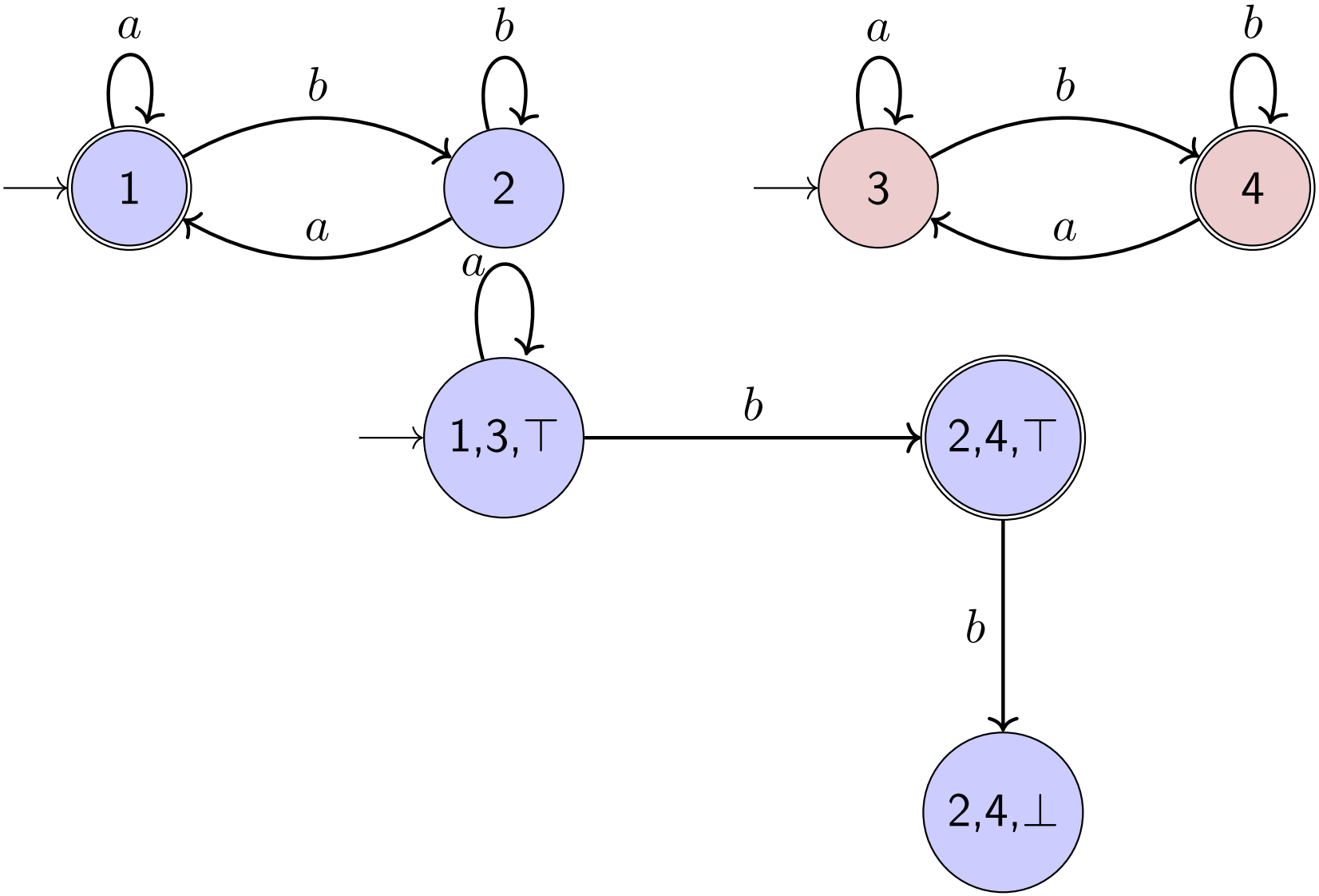




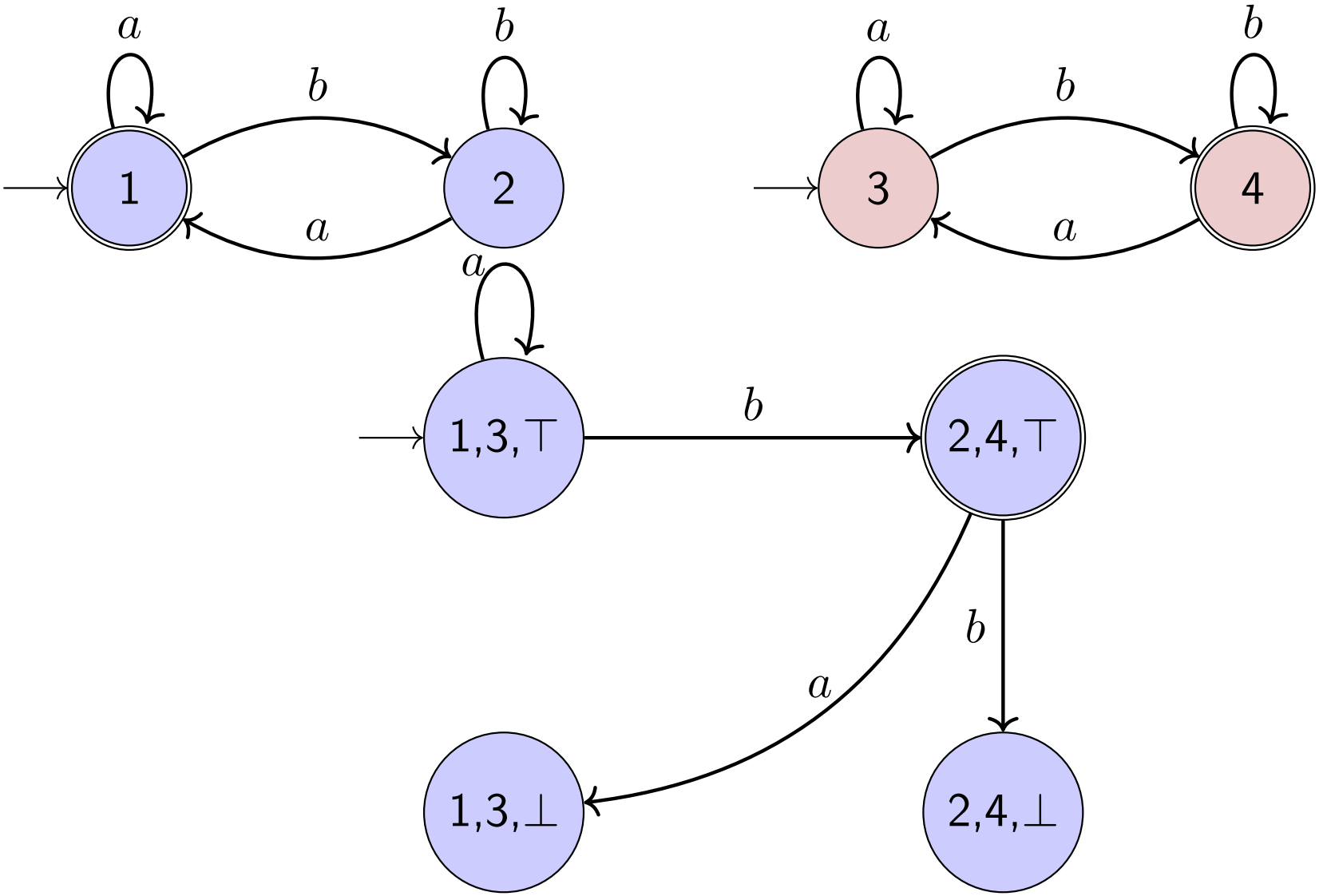
Exemple



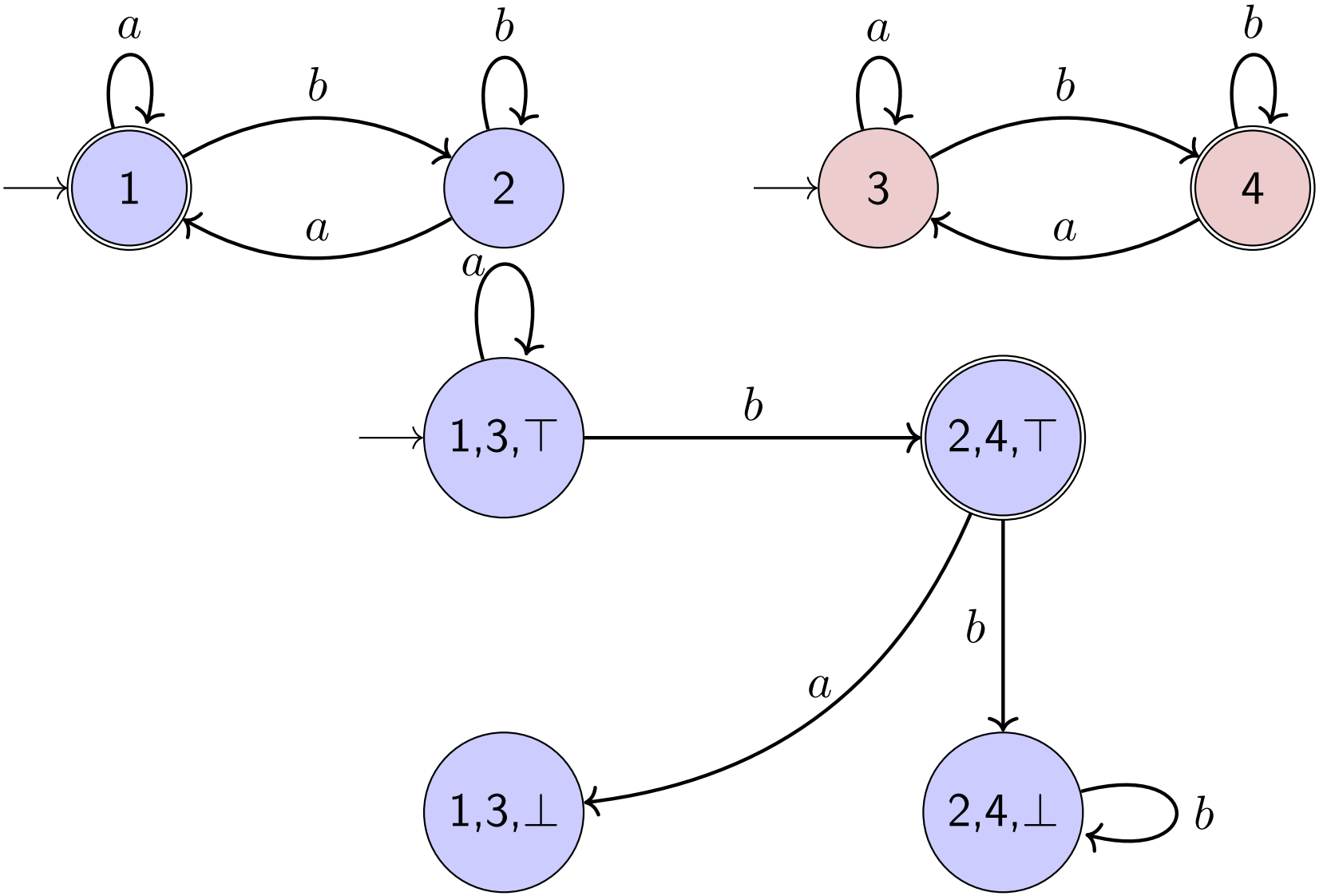
Exemple



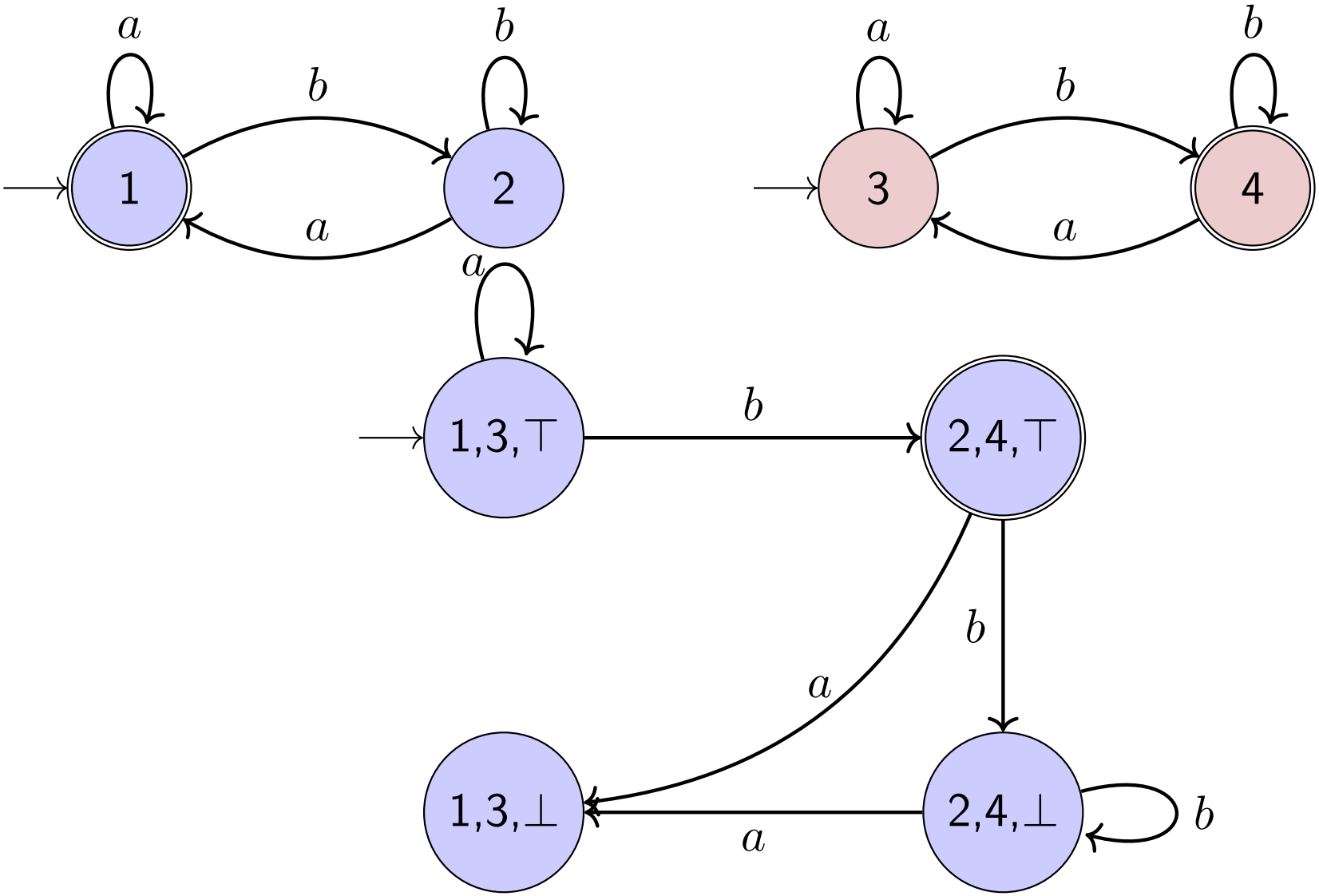
Exemple



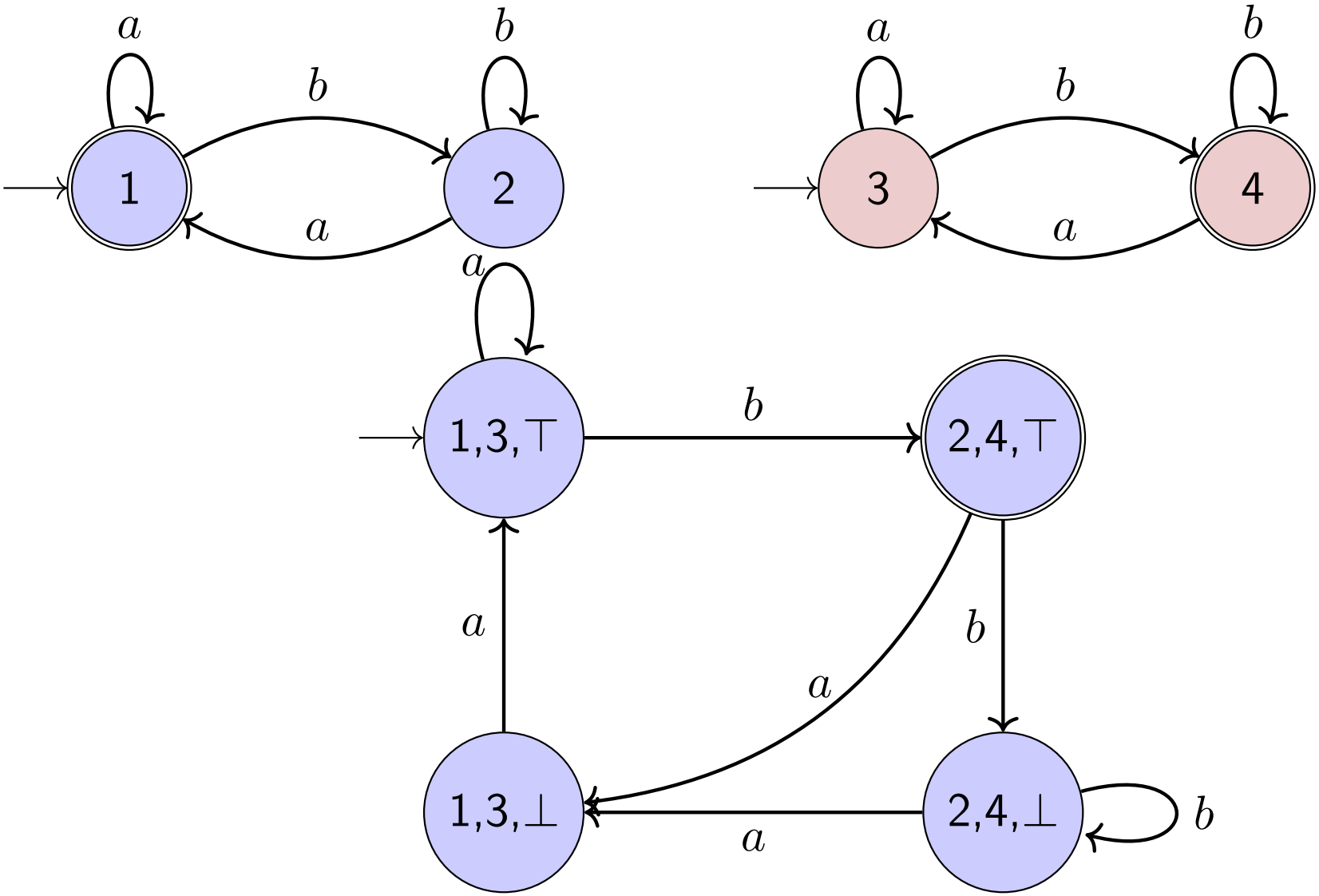
Exemple



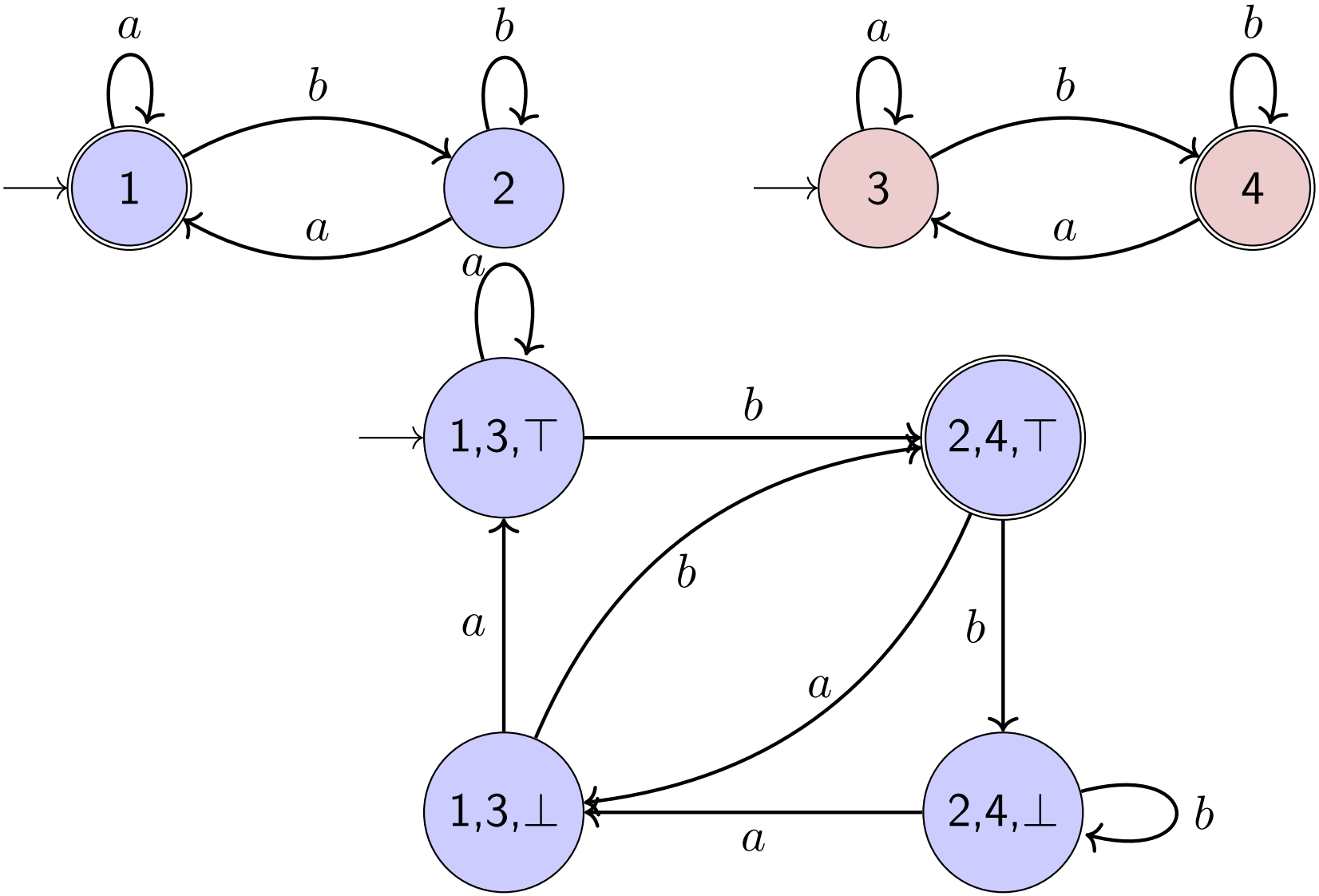
Exemple



Exemple



Exemple



Exercices

Chercher les exercices 15 à 18 de la feuille de TD1.