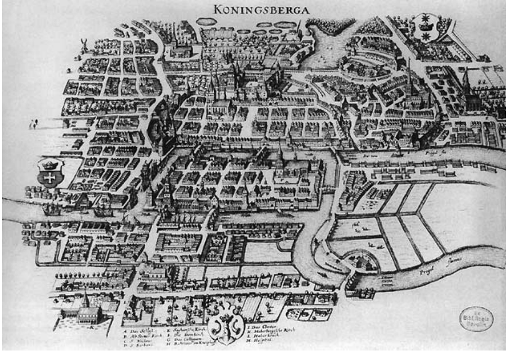
Graphes

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié



Graphes orientés

Graphe orienté

Un graphe orienté $\mathcal{G} = (E, \mathcal{A})$ est la donnée

Graphes orientés

Graphe orienté

Un graphe orienté $\mathcal{G} = (E, \mathcal{A})$ est la donnée

1. d'un ensemble E dont les éléments sont appelés sommets ou points

Graphes orientés

Graphe orienté

Un graphe orienté $\mathcal{G} = (E, \mathcal{A})$ est la donnée

1. d'un ensemble E dont les éléments sont appelés sommets ou points
2. et d'un ensemble $\mathcal{A}$ dont les éléments sont des couples de sommets appelés des arcs.

Graphes orientés

Graphe orienté

Un graphe orienté $\mathcal{G} = (E, \mathcal{A})$ est la donnée

1. d'un ensemble E dont les éléments sont appelés sommets ou points
2. et d'un ensemble $\mathcal{A}$ dont les éléments sont des couples de sommets appelés des arcs.

Si le nombre de sommets est n, le graphe est dit d'ordre n.

Graphes orientés

Graphe orienté

Un graphe orienté $\mathcal{G} = (E, \mathcal{A})$ est la donnée

1. d'un ensemble E dont les éléments sont appelés sommets ou points
2. et d'un ensemble $\mathcal{A}$ dont les éléments sont des couples de sommets appelés des arcs.

Si le nombre de sommets est n, le graphe est dit d'ordre n.
Si le nombre d'arcs est m, le graphe est dit de taille m.

Graphes orientés

Extrémité initiale, extrémité finale

Graphes orientés

Extrémité initiale, extrémité finale

On peut noter $i = 1, 2, ..., n$ les sommets.

Graphes orientés

Extrémité initiale, extrémité finale

On peut noter $i = 1, 2, ..., n$ les sommets.
Si $\alpha = (i, j)$ (on note encore $[i, j]$) est un arc de $\mathcal{A}$, i est l'extrémité initiale de α et j l'extrémité finale de α.

Graphes orientés

Extrémité initiale, extrémité finale

On peut noter $i = 1, 2, ..., n$ les sommets.
Si $\alpha = (i, j)$ (on note encore $[i, j]$) est un arc de $\mathcal{A}$, i est l'extrémité initiale de α et j l'extrémité finale de α.
Les sommets peuvent être représentés graphiquement par des points

Graphes orientés

Extrémité initiale, extrémité finale

On peut noter $i = 1, 2, ..., n$ les sommets.
Si $\alpha = (i, j)$ (on note encore $[i, j]$) est un arc de $\mathcal{A}$, i est l'extrémité initiale de α et j l'extrémité finale de α.
Les sommets peuvent être représentés graphiquement par des points et un arc $\alpha = (i, j)$ par une flèche joignant les points i et j.

Graphes orientés

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Extrémité initiale, extrémité finale

On peut noter $i = 1, 2, ..., n$ les sommets.
Si $\alpha = (i, j)$ (on note encore $[i, j]$) est un arc de $\mathcal{A}$, i est l'extrémité initiale de α et j l'extrémité finale de α.
Les sommets peuvent être représentés graphiquement par des points et un arc $\alpha = (i, j)$ par une flèche joignant les points i et j.
Un arc (i, i) est appelé une boucle.

Graphes orientés

p-graphe

Graphes orientés

p-graphe

Un p-graphe est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i, j) entre deux sommets quelconques i et j pris dans cet ordre.

Graphes orientés

p-graphe

Un *p-graphe* est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i, j) entre deux sommets quelconques i et j pris dans cet ordre.
Un 1-graphe est dit simple s'il ne comporte pas de boucle.

Graphes orientés

p-graphe

Un p-graphe est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i, j) entre deux sommets quelconques i et j pris dans cet ordre.
Un 1-graphe est dit simple s'il ne comporte pas de boucle.

Prédecesseur, successeur

Graphes orientés

p-graphe

Un p-graphe est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i, j) entre deux sommets quelconques i et j pris dans cet ordre.
Un 1-graphe est dit simple s'il ne comporte pas de boucle.

Prédecesseur, successeur

i est dit prédecesseur de j s'il existe un arc ayant i comme extrémité initiale et j comme extrémité finale.

Graphes orientés

p-graphe

Un p-graphe est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i, j) entre deux sommets quelconques i et j pris dans cet ordre.
Un 1-graphe est dit simple s'il ne comporte pas de boucle.

Prédecesseur, successeur

i est dit prédecesseur de j s'il existe un arc ayant i comme extrémité initiale et j comme extrémité finale.
j est dit successeur de i s'il existe un arc ayant i comme extrémité initiale et j comme extrémité finale.

Graphes orientés

p-graphe

Un p-graphe est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i, j) entre deux sommets quelconques i et j pris dans cet ordre.
Un 1-graphe est dit simple s'il ne comporte pas de boucle.

Prédecesseur, successeur

i est dit prédecesseur de j s'il existe un arc ayant i comme extrémité initiale et j comme extrémité finale.
j est dit successeur de i s'il existe un arc ayant i comme extrémité initiale et j comme extrémité finale.
L'ensemble des successeurs d'un sommet i est noté Γ_i.

Graphes orientés

p-graphe

Un p-graphe est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i, j) entre deux sommets quelconques i et j pris dans cet ordre.
Un 1-graphe est dit simple s'il ne comporte pas de boucle.

Prédecesseur, successeur

i est dit prédecesseur de j s'il existe un arc ayant i comme extrémité initiale et j comme extrémité finale.
j est dit successeur de i s'il existe un arc ayant i comme extrémité initiale et j comme extrémité finale.
L'ensemble des successeurs d'un sommet i est noté Γ_i.
L'ensemble des prédécesseurs d'un sommet j est noté Γ_j^{-1}.

Graphes orientés

Exemple 1

Graphes orientés

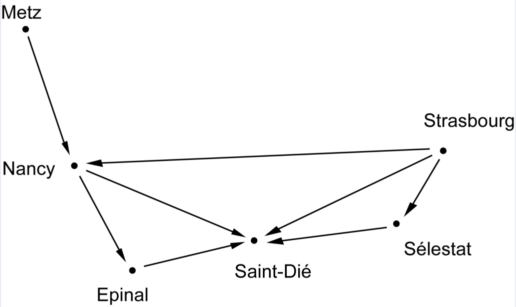
Exemple 1

Accès possibles à Saint-Dié à partir des grandes villes périphériques.

Graphes orientés

Exemple 1

Accès possibles à Saint-Dié à partir des grandes villes périphériques.



Graphes orientés

Exemple 1

Graphes orientés

Exemple 1

$E = \{\text{Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg}\}$

Graphes orientés

Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$
$\mathcal{A} = \{$(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)$\}$.

Graphes orientés

Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$
$\mathcal{A} = \{$(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)$\}$.
Il s'agit d'un 1-graphe simple.

Graphes orientés

Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$
$\mathcal{A} = \{$(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)$\}$.
Il s'agit d'un 1-graphe simple.
Ordre : 6

Graphes orientés

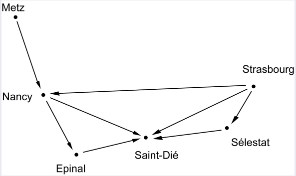
Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$
$\mathcal{A} = \{$(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)$\}$.
Il s'agit d'un 1-graphe simple.
Ordre : 6
Taille : 8

Graphes orientés

Exemple 1

E = {Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg}
$\mathcal{A}$ = {(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)}.
Il s'agit d'un 1-graphe simple.
Ordre : 6
Taille : 8



Graphes orientés

Exemple 1

Graphes orientés

Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$

Graphes orientés

Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$
$\mathcal{A} = \{$(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)$\}$.

Graphes orientés

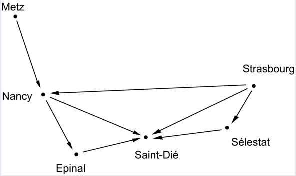
Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$
$\mathcal{A} = \{$(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)$\}$.
$\Gamma_{Strasbourg} = \{$Sélestat, Nancy, Saint-Dié$\}$

Graphes orientés

Exemple 1

$E = \{$Epinal, Metz, Nancy, Saint-Dié, Sélestat, Strasbourg$\}$
$\mathcal{A} = \{$(Strasbourg, Saint-Dié), (Strasbourg, Nancy), (Strasbourg, Sélestat), (Sélestat, Saint-Dié), (Metz, Nancy), (Nancy, Epinal), (Nancy, Saint-Dié), (Epinal, Saint-Dié)$\}$.
$\Gamma_{Strasbourg} = \{$Sélestat, Nancy, Saint-Dié$\}$
$\Gamma^{-1}_{Saint-Dié} = \{$Epinal, Sélestat, Strasbourg,Nancy$\}$.



Graphes orientés

Exemple 2

Graphes orientés

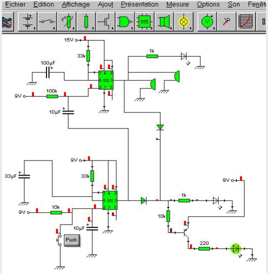
Exemple 2

Réseau électrique :

Graphes orientés

Exemple 2

Réseau électrique :

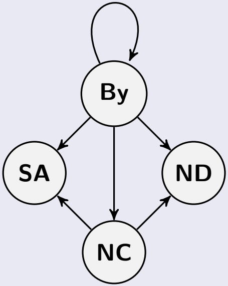


Graphes orientés

Exemple 3 : Creative commons

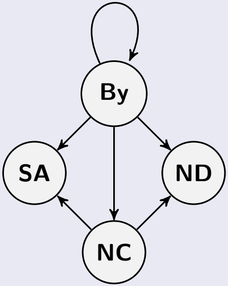
Graphes orientés

Exemple 3 : Creative commons



Graphes orientés

Exemple 3 : Creative commons



By : Attribution (paternité)
SA : Share Adapt (partage dans les mêmes conditions)
ND : No Derivatives (pas de modification)
NC : Non Commercial (pas d'utilisation commerciale)

Graphes non orientés

Définitions

Graphes non orientés

Définitions

On peut s'intéresser uniquement à l'existence d'un arc entre deux sommets sans se préoccuper de l'orientation des arcs.

Graphes non orientés

Définitions

On peut s'intéresser uniquement à l'existence d'un arc entre deux sommets sans se préoccuper de l'orientation des arcs.
On parle de graphe non orienté $\mathcal{G} = (E, \mathcal{A})$ et d'arêtes d'extrémités i et j notées $\{i, j\}$ (paires de sommets).

Graphes non orientés

Définitions

On peut s'intéresser uniquement à l'existence d'un arc entre deux sommets sans se préoccuper de l'orientation des arcs.
On parle de graphe non orienté $\mathcal{G} = (E, \mathcal{A})$ et d'arêtes d'extrémités i et j notées $\{i, j\}$ (paires de sommets).
Graphiquement, une arête $\{i, j\}$ est représentée par un segment sans flèche.

Graphes non orientés

Définitions

On peut s'intéresser uniquement à l'existence d'un arc entre deux sommets sans se préoccuper de l'orientation des arcs.
On parle de graphe non orienté $\mathcal{G} = (E, \mathcal{A})$ et d'arêtes d'extrémités i et j notées $\{i, j\}$ (paires de sommets).
Graphiquement, une arête $\{i, j\}$ est représentée par un segment sans flèche.
Une boucle est notée $\{i\}$.

Graphes non orientés

Définitions

On peut s'intéresser uniquement à l'existence d'un arc entre deux sommets sans se préoccuper de l'orientation des arcs.
On parle de graphe non orienté $\mathcal{G} = (E, \mathcal{A})$ et d'arêtes d'extrémités i et j notées $\{i, j\}$ (paires de sommets).
Graphiquement, une arête $\{i, j\}$ est représentée par un segment sans flèche.
Une boucle est notée $\{i\}$.
L'ordre du graphe est le cardinal de E et la taille celui de $\mathcal{A}$.

Graphes non orientés

Exemple

Graphes non orientés

Exemple

Accès possibles à Saint-Dié à partir des grandes villes périphériques.

Graphes non orientés

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

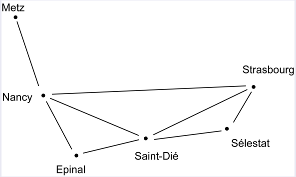
Graphes
- Généralités
- Arbre
- Cycle eulérien, hamiltonien
- Coloration
- Le problème du plus court chemin
- Arbre recouvrant de poids minimal

Langages et automates
- Langages
- Automates

Annexes

Exemple

Accès possibles à Saint-Dié à partir des grandes villes périphériques.



Graphes non orientés

Définitions

Graphes non orientés

Définitions

Un multigraphe est un graphe non orienté pouvant comporter plusieurs arêtes entre deux sommets.

Graphes non orientés

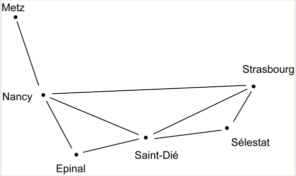
Définitions

Un multigraphe est un graphe non orienté pouvant comporter plusieurs arêtes entre deux sommets.
Un graphe non orienté est dit simple s'il est sans boucle et s'il y a au plus une arête entre deux sommets quelconques.

Graphes non orientés

Définitions

Un multigraphe est un graphe non orienté pouvant comporter plusieurs arêtes entre deux sommets.
Un graphe non orienté est dit simple s'il est sans boucle et s'il y a au plus une arête entre deux sommets quelconques.



Graphes non orientés

Exemple 1

Graphes non orientés

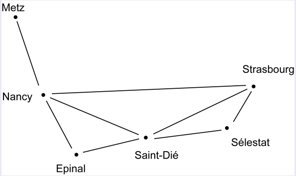
Exemple 1

Notation : $\mathcal{A}' = \{\{\text{Strasbourg, Saint-Dié}\}, \{\text{Strasbourg, Sélestat}\}, \{\text{Strasbourg, Nancy}\}, \{\text{Nancy, Epinal}\}, \{\text{Metz, Nancy}\}, \{\text{Nancy, Saint-Dié}\}, \{\text{Sélestat, Saint-Dié}\}, \{\text{Epinal, Saint-Dié}\}\}$.

Graphes non orientés

Exemple 1

Notation : $\mathcal{A}' = \{\{$Strasbourg, Saint-Dié$\}$, $\{$Strasbourg, Sélestat$\}$, $\{$ Strasbourg, Nancy$\}$, $\{$Nancy, Epinal$\}$, $\{$Metz, Nancy$\}$, $\{$Nancy, Saint-Dié$\}$, $\{$Sélestat, Saint-Dié$\}$, $\{$Epinal, Saint-Dié$\}\}$.



Graphes non orientés

Exemple 2

Graphes non orientés

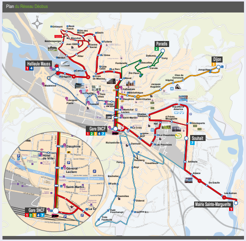
Exemple 2

Un réseau de transport (routier, ferroviaire, métro, etc) peut être représenté par un graphe dont les sommets sont des lieux (intersections de rues, gares, stations de métro, etc) et les arêtes indiquent la possibilité d'aller d'un lieu à un autre (par un tronçon de route, une ligne de train ou de métro, etc.) :

Graphes non orientés

Exemple 2

Un réseau de transport (routier, ferroviaire, métro, etc) peut être représenté par un graphe dont les sommets sont des lieux (intersections de rues, gares, stations de métro, etc) et les arêtes indiquent la possibilité d'aller d'un lieu à un autre (par un tronçon de route, une ligne de train ou de métro, etc.) :



Graphes non orientés

Exemple 3

Graphes non orientés

Exemple 3

Le plan du métro de Paris

Graphes non orientés

Exemple 3

Le plan du métro de Paris

Graphes non orientés

Exemple 4

Graphes non orientés

Exemple 4

Un réseau social peut être représenté par un graphe dont les sommets sont les membres, et les arêtes les relations entre membres :

Graphes non orientés

Exemple 4

Un réseau social peut être représenté par un graphe dont les sommets sont les membres, et les arêtes les relations entre membres :

Graphes non orientés

Exemple 4

Graphes non orientés

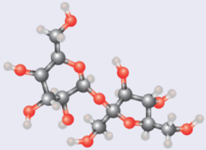
Exemple 4

Une molécule peut être représentée par un graphe dont les sommets sont les atomes, et les arêtes les liaisons entre atomes :

Graphes non orientés

Exemple 4

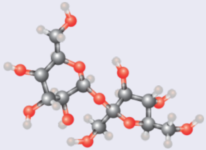
Une molécule peut être représentée par un graphe dont les sommets sont les atomes, et les arêtes les liaisons entre atomes :



Graphes non orientés

Exemple 4

Une molécule peut être représentée par un graphe dont les sommets sont les atomes, et les arêtes les liaisons entre atomes :

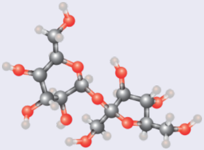


Remarque

Graphes non orientés

Exemple 4

Une molécule peut être représentée par un graphe dont les sommets sont les atomes, et les arêtes les liaisons entre atomes :



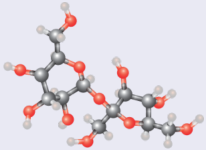
Remarque

Dans la suite, on précisera si le graphe est non orienté.

Graphes non orientés

Exemple 4

Une molécule peut être représentée par un graphe dont les sommets sont les atomes, et les arêtes les liaisons entre atomes :



Remarque

Dans la suite, on précisera si le graphe est non orienté.
Dans le cas contraire, il s'agira d'un graphe orienté.

Graphes : définitions

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Graphe complet

Graphes : définitions

Graphe complet

Un graphe simple est complet si $\forall i \in E\ \forall j \in E\ (i,j) \in \mathcal{A}$ ou $(j,i) \in \mathcal{A}$ (respectivement $\{i,j\} \in \mathcal{A}$).

Graphes : définitions

Graphe complet

Un graphe simple est complet si $\forall i \in E\ \forall j \in E\ (i,j) \in \mathcal{A}$ ou $(j,i) \in \mathcal{A}$ (respectivement $\{i,j\} \in \mathcal{A}$).
Le graphe non orienté complet d'ordre n est noté K_n (graphe de Kuratowski).

Graphes : définitions

Graphe complet

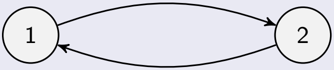
Un graphe simple est complet si $\forall i \in E\ \forall j \in E\ (i,j) \in \mathcal{A}$ ou $(j,i) \in \mathcal{A}$ (respectivement $\{i,j\} \in \mathcal{A}$).
Le graphe non orienté complet d'ordre n est noté K_n (graphe de Kuratowski).
Kazimierz Kuratowski (1896-1980) est un mathématicien polonais.

Graphes : définitions

Graphe complet

Un graphe simple est complet si $\forall i \in E\ \forall j \in E\ (i,j) \in \mathcal{A}$ ou $(j,i) \in \mathcal{A}$ (respectivement $\{i,j\} \in \mathcal{A}$).
Le graphe non orienté complet d'ordre n est noté K_n (graphe de Kuratowski).
Kazimierz Kuratowski (1896-1980) est un mathématicien polonais.

Un graphe orienté complet d'ordre 2

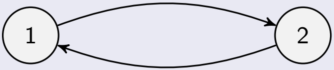


Graphes : définitions

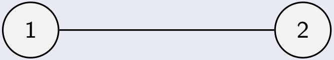
Graphe complet

Un graphe simple est complet si $\forall i \in E \ \forall j \in E \ (i,j) \in \mathcal{A}$ ou $(j,i) \in \mathcal{A}$ (respectivement $\{i,j\} \in \mathcal{A}$).
Le graphe non orienté complet d'ordre n est noté K_n (graphe de Kuratowski).
Kazimierz Kuratowski (1896-1980) est un mathématicien polonais.

Un graphe orienté complet d'ordre 2



K_2



Graphes : définitions

Graphe biparti

Graphe biparti

Un graphe non orienté est dit biparti s'il existe une partition $\{E_1, E_2\}$ de l'ensemble des sommets E telle que $\forall\{i,j\} \in \mathcal{A}\ i \in E_1 \Rightarrow j \in E_2$ et $i \in E_2 \Rightarrow j \in E_1$.

Graphes : définitions

Graphe biparti

Un graphe non orienté est dit biparti s'il existe une partition $\{E_1, E_2\}$ de l'ensemble des sommets E telle que $\forall\{i,j\} \in \mathcal{A}$ $i \in E_1 \Rightarrow j \in E_2$ et $i \in E_2 \Rightarrow j \in E_1$.

Exemple

Graphes : définitions

Graphe biparti

Un graphe non orienté est dit biparti s'il existe une partition $\{E_1, E_2\}$ de l'ensemble des sommets E telle que $\forall \{i, j\} \in \mathcal{A}$ $i \in E_1 \Rightarrow j \in E_2$ et $i \in E_2 \Rightarrow j \in E_1$.

Exemple

$E = \{1, 2, 3, 4, 5\}$, $E_1 = \{1, 2, 3\}$, $E_2 = \{4, 5\}$

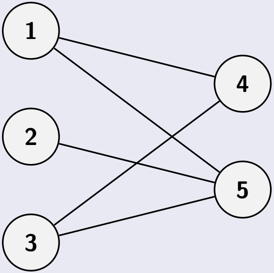
Graphes : définitions

Graphe biparti

Un graphe non orienté est dit biparti s'il existe une partition $\{E_1, E_2\}$ de l'ensemble des sommets E telle que $\forall\{i,j\} \in \mathcal{A}\ i \in E_1 \Rightarrow j \in E_2$ et $i \in E_2 \Rightarrow j \in E_1$.

Exemple

$E = \{1,2,3,4,5\}$, $E_1 = \{1,2,3\}$, $E_2 = \{4,5\}$



Graphes : définitions

Arcs adjacents, arêtes adjacentes

Graphes : définitions

Arcs adjacents, arêtes adjacentes

Deux arcs (respectivement arêtes) sont dits adjacents (respectivement adjacentes) s'ils (elles) ont au moins une extrémité commune.

Graphes : définitions

Arcs adjacents, arêtes adjacentes

Deux arcs (respectivement arêtes) sont dits adjacents (respectivement adjacentes) s'ils (elles) ont au moins une extrémité commune.

Exemple

Graphes : définitions

Arcs adjacents, arêtes adjacentes

Deux arcs (respectivement arêtes) sont dits adjacents (respectivement adjacentes) s'ils (elles) ont au moins une extrémité commune.

Exemple

1. (Strasbourg, Sélestat) et (Sélestat, Saint-Dié) sont des arcs adjacents.

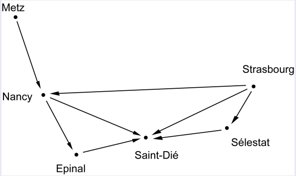
Graphes : définitions

Arcs adjacents, arêtes adjacentes

Deux arcs (respectivement arêtes) sont dits adjacents (respectivement adjacentes) s'ils (elles) ont au moins une extrémité commune.

Exemple

1. (Strasbourg, Sélestat) et (Sélestat, Saint-Dié) sont des arcs adjacents.
2. {Nancy, Epinal}, {Metz, Nancy} sont des arêtes adjacentes.



Graphes : définitions

Sommet incident à un arc ou une arête, sommets adjacents

Sommet incident à un arc ou une arête, sommets adjacents

Un sommet est dit incident à un arc ou une arête s'il s'agit d'une extrémité de l'arc ou de l'arête.

Graphes : définitions

Sommet incident à un arc ou une arête, sommets adjacents

Un sommet est dit incident à un arc ou une arête s'il s'agit d'une extrémité de l'arc ou de l'arête.
Des sommets incidents à une même arête sont dits adjacents.

Graphes : définitions

Sommet incident à un arc ou une arête, sommets adjacents

Un sommet est dit incident à un arc ou une arête s'il s'agit d'une extrémité de l'arc ou de l'arête.
Des sommets incidents à une même arête sont dits adjacents.

Exemple

Graphes : définitions

Sommet incident à un arc ou une arête, sommets adjacents

Un sommet est dit incident à un arc ou une arête s'il s'agit d'une extrémité de l'arc ou de l'arête.
Des sommets incidents à une même arête sont dits adjacents.

Exemple

1. Le sommet Strasbourg est incident à l'arc (Strasbourg, Sélestat).

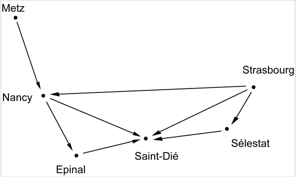
Graphes : définitions

Sommet incident à un arc ou une arête, sommets adjacents

Un sommet est dit incident à un arc ou une arête s'il s'agit d'une extrémité de l'arc ou de l'arête.
Des sommets incidents à une même arête sont dits adjacents.

Exemple

1. Le sommet Strasbourg est incident à l'arc (Strasbourg, Sélestat).
2. Les sommets Nancy et Epinal sont adjacents : $\{$Nancy,Epinal$\}$ est une arête du graphe.



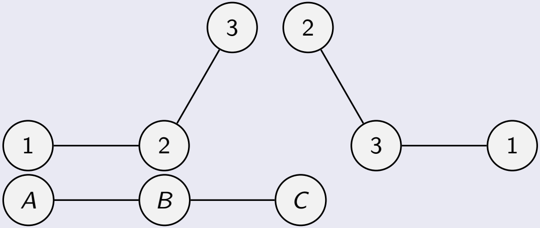
Graphes : définitions

Isomorphisme de graphes

Graphes : définitions

Isomorphisme de graphes

Deux graphes $\mathcal{G} = (E, \mathcal{A})$ et $\mathcal{G}' = (E', \mathcal{A}')$ sont isomorphes s'il existe une bijection f de E sur E' telle que pour tout sommet x et y de $\mathcal{G}$:
x adjacent à y dans $\mathcal{G} \Leftrightarrow f(x)$ adjacent à $f(y)$ dans $\mathcal{G}'$.
Exemple de graphes isomorphes :



Graphes : définitions

Demi-degré, degré

Graphes : définitions

Demi-degré, degré

Le demi-degré extérieur du sommet i, noté d_i^+ ou $d^+(i)$ est le nombre d'arcs ayant i comme extrémité initiale.

Graphes : définitions

Demi-degré, degré

Le demi-degré extérieur du sommet i, noté d_i^+ ou $d^+(i)$ est le nombre d'arcs ayant i comme extrémité initiale.
$d_i^+ = |\Gamma_i|$ dans le cas d'un 1-graphe.

Graphes : définitions

Demi-degré, degré

Le demi-degré extérieur du sommet i, noté d_i^+ ou $d^+(i)$ est le nombre d'arcs ayant i comme extrémité initiale.
$d_i^+ = |\Gamma_i|$ dans le cas d'un 1-graphe.
Le demi-degré intérieur du sommet j, noté d_j^- ou $d^-(j)$ est le nombre d'arcs ayant j comme extrémité finale.

Graphes : définitions

Demi-degré, degré

Le demi-degré extérieur du sommet i, noté d_i^+ ou $d^+(i)$ est le nombre d'arcs ayant i comme extrémité initiale.
$d_i^+ = |\Gamma_i|$ dans le cas d'un 1-graphe.
Le demi-degré intérieur du sommet j, noté d_j^- ou $d^-(j)$ est le nombre d'arcs ayant j comme extrémité finale.
$d_j^- = \left|\Gamma_j^{-1}\right|$ dans le cas d'un 1-graphe.

Graphes : définitions

Demi-degré, degré

Le demi-degré extérieur du sommet i, noté d_i^+ ou $d^+(i)$ est le nombre d'arcs ayant i comme extrémité initiale.
$d_i^+ = |\Gamma_i|$ dans le cas d'un 1-graphe.
Le demi-degré intérieur du sommet j, noté d_j^- ou $d^-(j)$ est le nombre d'arcs ayant j comme extrémité finale.
$d_j^- = \left|\Gamma_j^{-1}\right|$ dans le cas d'un 1-graphe.
Le degré du sommet i, noté d_i ou $d(i)$, est le nombre d'arcs (respectivement arêtes) ayant i comme extrémité,

Graphes : définitions

Demi-degré, degré

Le demi-degré extérieur du sommet i, noté d_i^+ ou $d^+(i)$ est le nombre d'arcs ayant i comme extrémité initiale.
$d_i^+ = |\Gamma_i|$ dans le cas d'un 1-graphe.
Le demi-degré intérieur du sommet j, noté d_j^- ou $d^-(j)$ est le nombre d'arcs ayant j comme extrémité finale.
$d_j^- = \left|\Gamma_j^{-1}\right|$ dans le cas d'un 1-graphe.
Le degré du sommet i, noté d_i ou $d(i)$, est le nombre d'arcs (respectivement arêtes) ayant i comme extrémité,
et on a

Graphes : définitions

Demi-degré, degré

Le demi-degré extérieur du sommet i, noté d_i^+ ou $d^+(i)$ est le nombre d'arcs ayant i comme extrémité initiale.
$d_i^+ = |\Gamma_i|$ dans le cas d'un 1-graphe.
Le demi-degré intérieur du sommet j, noté d_j^- ou $d^-(j)$ est le nombre d'arcs ayant j comme extrémité finale.
$d_j^- = \left|\Gamma_j^{-1}\right|$ dans le cas d'un 1-graphe.
Le degré du sommet i, noté d_i ou $d(i)$, est le nombre d'arcs (respectivement arêtes) ayant i comme extrémité,
et on a : $d_i = d_i^+ + d_i^-$.

Graphes : définitions

Somme des demi-degrés, des degrés

Graphes : définitions

Somme des demi-degrés, des degrés

$$\sum_{i=1}^{n} d^{+}(i) = \sum_{i=1}^{n} d^{-}(i) = m$$

Graphes : définitions

Somme des demi-degrés, des degrés

$$\sum_{i=1}^{n} d^+(i) = \sum_{i=1}^{n} d^-(i) = m$$

$$\sum_{i=1}^{n} d(i) = 2m$$

Graphes : définitions

Somme des demi-degrés, des degrés

$$\sum_{i=1}^{n} d^{+}(i) = \sum_{i=1}^{n} d^{-}(i) = m$$

$$\sum_{i=1}^{n} d(i) = 2m$$

Remarque : dans un graphe non orienté le degré d'une boucle est égal à 2.

Graphes : définitions

Exemple

Graphes : définitions

Exemple

$d^+(\text{Nancy}) = 2,$

Graphes : définitions

Exemple

$d^+(\text{Nancy}) = 2, \ d^+(\text{Saint-Dié}) = 0$

Graphes : définitions

Exemple

$d^+(\text{Nancy}) = 2,\ d^+(\text{Saint-Dié}) = 0$
$d^-(\text{Nancy}) = 2,$

Graphes : définitions

Exemple

$d^+(\text{Nancy}) = 2,\ d^+(\text{Saint-Dié}) = 0$

$d^-(\text{Nancy}) = 2,\ d^-(\text{Saint-Dié}) = 4$

Graphes : définitions

Exemple

$d^+(\text{Nancy}) = 2,\ d^+(\text{Saint-Dié}) = 0$
$d^-(\text{Nancy}) = 2,\ d^-(\text{Saint-Dié}) = 4$
$d(\text{Nancy}) = d^+(\text{Nancy}) + d^-(\text{Nancy}) = 2 + 2 = 4$

Graphes : définitions

Exemple

$d^+(\text{Nancy}) = 2,\ d^+(\text{Saint-Dié}) = 0$
$d^-(\text{Nancy}) = 2,\ d^-(\text{Saint-Dié}) = 4$
$d(\text{Nancy}) = d^+(\text{Nancy}) + d^-(\text{Nancy}) = 2 + 2 = 4$

Graphes : définitions

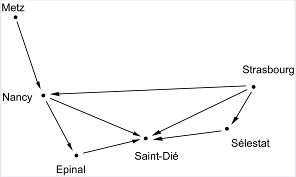
Exemple

$d^+(\text{Nancy}) = 2,\ d^+(\text{Saint-Dié}) = 0$
$d^-(\text{Nancy}) = 2,\ d^-(\text{Saint-Dié}) = 4$
$d(\text{Nancy}) = d^+(\text{Nancy}) + d^-(\text{Nancy}) = 2 + 2 = 4$
$d(\text{Saint-Dié}) = d^+(\text{Saint-Dié}) + d^-(\text{Saint-Dié}) = 0 + 4 = 4$

Graphes : définitions

Exemple

$d^+(\text{Nancy}) = 2,\ d^+(\text{Saint-Dié}) = 0$
$d^-(\text{Nancy}) = 2,\ d^-(\text{Saint-Dié}) = 4$
$d(\text{Nancy}) = d^+(\text{Nancy}) + d^-(\text{Nancy}) = 2 + 2 = 4$
$d(\text{Saint-Dié}) = d^+(\text{Saint-Dié}) + d^-(\text{Saint-Dié}) = 0 + 4 = 4$

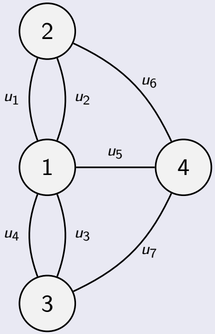


Graphes : définitions

Exemple

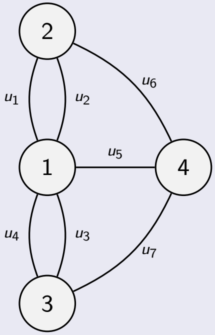
Graphes : définitions

Exemple



Graphes : définitions

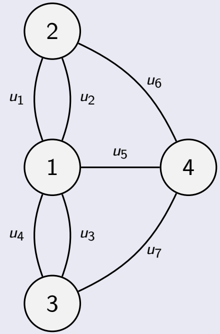
Exemple



$d(1) = 5$ et $d(2) = d(3) = d(4) = 3$

Graphes : définitions

Exemple



$d(1) = 5$ et $d(2) = d(3) = d(4) = 3$ et
$d(1) + d(2) + d(3) + d(4) = 14 = 2 \times 7$

Graphes : définitions

Arcs incidents (arêtes incidentes)

Graphes : définitions

Arcs incidents (arêtes incidentes)

Soit $A \subset E$.

Graphes : définitions

Arcs incidents (arêtes incidentes)

Soit $A \subset E$.

- $\omega^+(A)$ est l'ensemble des arcs ayant leur extrémité initiale dans A et leur extrémité finale dans $E - A$.

Graphes : définitions

Arcs incidents (arêtes incidentes)

Soit $A \subset E$.

- $\omega^+(A)$ est l'ensemble des arcs ayant leur extrémité initiale dans A et leur extrémité finale dans $E - A$.
- $\omega^-(A)$ est l'ensemble des arcs ayant leur extrémité finale dans A et leur extrémité initiale dans $E - A$.

Graphes : définitions

Arcs incidents (arêtes incidentes)

Soit $A \subset E$.

- $\omega^+(A)$ est l'ensemble des arcs ayant leur extrémité initiale dans A et leur extrémité finale dans $E - A$.
- $\omega^-(A)$ est l'ensemble des arcs ayant leur extrémité finale dans A et leur extrémité initiale dans $E - A$.
- $\omega(A) = \omega^-(A) \cup \omega^+(A)$ (ensemble des arcs incidents ou arêtes incidentes à A) est appelé cocycle du graphe.

Graphes : définitions

Arcs incidents (arêtes incidentes)

Soit $A \subset E$.

- $\omega^+(A)$ est l'ensemble des arcs ayant leur extrémité initiale dans A et leur extrémité finale dans $E - A$.
- $\omega^-(A)$ est l'ensemble des arcs ayant leur extrémité finale dans A et leur extrémité initiale dans $E - A$.
- $\omega(A) = \omega^-(A) \cup \omega^+(A)$ (ensemble des arcs incidents ou arêtes incidentes à A) est appelé cocycle du graphe.

Remarque : pour un graphe non orienté, $\omega(A)$ est l'ensemble des arêtes ayant une extrémité dans A et l'autre dans $\overline{A}$.

Graphes : définitions

Exemple

Graphes : définitions

Exemple

$\omega^+(\text{Nancy}) = \{(\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$

Graphes : définitions

Exemple

$$\omega^+(\text{Nancy}) = \{(\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$$
$$\omega^-(\text{Nancy}) = \{(\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy})\}$$

Graphes : définitions

Exemple

$\omega^+(\text{Nancy}) = \{(\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$
$\omega^-(\text{Nancy}) = \{(\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy})\}$
$\omega(\text{Nancy}) =$
$\{(\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy}), (\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$

Graphes : définitions

Exemple

$\omega^+(\mathsf{Nancy}) = \{(\mathsf{Nancy}, \mathsf{Epinal}), (\mathsf{Nancy}, \mathsf{Saint\text{-}Dié})\}$
$\omega^-(\mathsf{Nancy}) = \{(\mathsf{Metz}, \mathsf{Nancy}), (\mathsf{Strasbourg}, \mathsf{Nancy})\}$
$\omega(\mathsf{Nancy}) =$
$\{(\mathsf{Metz}, \mathsf{Nancy}), (\mathsf{Strasbourg}, \mathsf{Nancy}), (\mathsf{Nancy}, \mathsf{Epinal}), (\mathsf{Nancy}, \mathsf{Saint\text{-}Dié})\}$
$\omega(\mathsf{Nancy}) =$
$\{\{\mathsf{Metz}, \mathsf{Nancy}\}, \{\mathsf{Strasbourg}, \mathsf{Nancy}\}, \{\mathsf{Epinal}, \mathsf{Nancy}\}, \{\mathsf{Saint\text{-}Dié}, \mathsf{Nancy}\}\}$

Graphes : définitions

Exemple

$\omega^+(\text{Nancy}) = \{(\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$

$\omega^-(\text{Nancy}) = \{(\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy})\}$

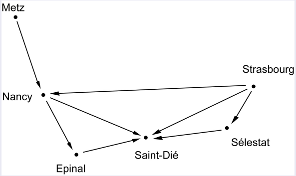
$\omega(\text{Nancy}) =$
$\{(\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy}), (\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$

$\omega(\text{Nancy}) =$
$\{\{\text{Metz}, \text{Nancy}\}, \{\text{Strasbourg}, \text{Nancy}\}, \{\text{Epinal}, \text{Nancy}\}, \{\text{Saint-Dié}, \text{Nancy}\}\}$

Graphes : définitions

Exemple

$\omega^+(\text{Nancy}) = \{(\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$
$\omega^-(\text{Nancy}) = \{(\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy})\}$
$\omega(\text{Nancy}) =$
$\{(\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy}), (\text{Nancy}, \text{Epinal}), (\text{Nancy}, \text{Saint-Dié})\}$
$\omega(\text{Nancy}) =$
$\{\{\text{Metz}, \text{Nancy}\}, \{\text{Strasbourg}, \text{Nancy}\}, \{\text{Epinal}, \text{Nancy}\}, \{\text{Saint-Dié}, \text{Nancy}\}\}$

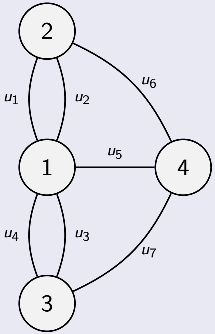


Graphes : définitions

Exemple

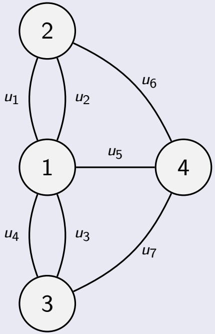
Graphes : définitions

Exemple



Graphes : définitions

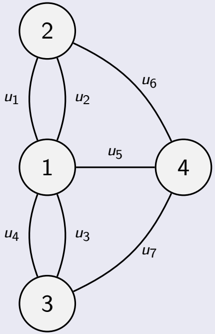
Exemple



$\omega(\{1,2\})$

Graphes : définitions

Exemple



$\omega(\{1,2\}) = \{u_3, u_4, u_5, u_6\}$

Graphes : définitions

Sous-graphe

Graphes : définitions

Sous-graphe

Le sous-graphe engendré par $F \subset E$ est le graphe dont les sommets sont les éléments de F et dont les arcs sont les arcs de $\mathcal{G}$ ayant leurs deux extrémités dans F.

Graphes : définitions

Sous-graphe

Le sous-graphe engendré par $F \subset E$ est le graphe dont les sommets sont les éléments de F et dont les arcs sont les arcs de $\mathcal{G}$ ayant leurs deux extrémités dans F.

Un sous-graphe complet est une clique.

Graphes : définitions

Sous-graphe

Le sous-graphe engendré par $F \subset E$ est le graphe dont les sommets sont les éléments de F et dont les arcs sont les arcs de $\mathcal{G}$ ayant leurs deux extrémités dans F.
Un sous-graphe complet est une clique.
Exemple

Graphes : définitions

Sous-graphe

Le sous-graphe engendré par $F \subset E$ est le graphe dont les sommets sont les éléments de F et dont les arcs sont les arcs de $\mathcal{G}$ ayant leurs deux extrémités dans F.

Un sous-graphe complet est une clique.

Exemple

Si $\mathcal{G}$ est le graphe représentant le métro de Paris, le graphe représentant les liaisons entre deux stations du 15$^{\text{ème}}$ arrondissement est un sous-graphe.

Graphes : définitions

Graphe partiel

Graphes : définitions

Graphe partiel

Soit $\mathcal{B} \subset \mathcal{A}$.

Graphes : définitions

Graphe partiel

Soit $\mathcal{B} \subset \mathcal{A}$. Le graphe partiel engendré par $\mathcal{B}$ est le graphe dont les arcs sont les arcs de $\mathcal{B}$.

Graphes : définitions

Graphe partiel

Soit $\mathcal{B} \subset \mathcal{A}$. Le graphe partiel engendré par $\mathcal{B}$ est le graphe dont les arcs sont les arcs de $\mathcal{B}$.

Exemple

Graphes : définitions

Graphe partiel

Soit $\mathcal{B} \subset \mathcal{A}$. Le graphe partiel engendré par $\mathcal{B}$ est le graphe dont les arcs sont les arcs de $\mathcal{B}$.

Exemple

Si $\mathcal{G}$ est le graphe représentant le réseau routier français, le graphe des routes départementales est un graphe partiel.

Matrice d'incidence ou matrice d'incidence sommets-arcs

Graphes : définitions

Matrice d'incidence ou matrice d'incidence sommets-arcs

La matrice d'incidence sommets-arcs d'un 1-graphe $\mathcal{G} = (E, \mathcal{A})$ est la matrice $A = (a_{ij})$ de type (n, m) $(n = |E|,\ m = |\mathcal{A}|)$ telle que $a_{ik} = 1$ et $a_{jk} = -1$ si $k = (i, j) \in \mathcal{A}$, 0 sinon.

Graphes : définitions

Matrice d'incidence ou matrice d'incidence sommets-arcs

La matrice d'incidence sommets-arcs d'un 1-graphe $\mathcal{G} = (E, \mathcal{A})$ est la matrice $A = (a_{ij})$ de type (n, m) $(n = |E|,\ m = |\mathcal{A}|)$ telle que $a_{ik} = 1$ et $a_{jk} = -1$ si $k = (i, j) \in \mathcal{A}$, 0 sinon.

Cas d'un graphe non orienté

Graphes : définitions

Matrice d'incidence ou matrice d'incidence sommets-arcs

La matrice d'incidence sommets-arcs d'un 1-graphe $\mathcal{G} = (E, \mathcal{A})$ est la matrice $A = (a_{ij})$ de type (n, m) $(n = |E|,\ m = |\mathcal{A}|)$ telle que $a_{ik} = 1$ et $a_{jk} = -1$ si $k = (i, j) \in \mathcal{A}$, 0 sinon.

Cas d'un graphe non orienté

Dans le cas d'un graphe non orienté, on pose $a_{ik} = a_{jk} = 1$ si $\{i, j\} \in \mathcal{A}$, 0 sinon.

Graphes : définitions

Matrice d'incidence sommets-arcs

Graphes : définitions

Matrice d'incidence sommets-arcs

$$
\begin{array}{c c}
 & \begin{array}{cccccc} 1 & 2 & \cdots & k & \cdots & \mathrm{m} \end{array} \\
\begin{array}{c} 1 \\ 2 \\ \cdots \\ i \\ \cdots \\ j \\ \cdots \\ n \end{array} &
\begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1k} & \cdots & a_{1m} \\
a_{21} & a_{22} & \cdots & a_{2k} & \cdots & a_{2m} \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
a_{i1} & a_{i2} & \cdots & a_{ik} & \cdots & a_{im} \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
a_{j1} & a_{j2} & \cdots & a_{jk} & \cdots & a_{jm} \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
a_{n1} & a_{n2} & \cdots & a_{nk} & \cdots & a_{nm}
\end{pmatrix}
\end{array}
$$

Graphes : définitions

Matrice d’incidence sommets-arcs : exemple

Graphes : définitions

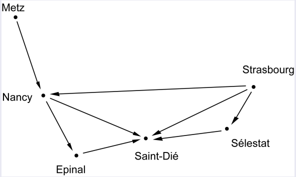
Matrice d'incidence sommets-arcs : exemple

$$
\begin{array}{c c}
 & \begin{array}{cccccccc} \text{MN} & \text{NE} & \text{ESD} & \text{NSD} & \text{SN} & \text{SSD} & \text{SSe} & \text{SeSD} \end{array} \\
\begin{array}{c} \text{E} \\ \text{M} \\ \text{N} \\ \text{SD} \\ \text{Se} \\ \text{S} \end{array} &
\begin{pmatrix}
0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & 0 & 1 & -1 & 0 & 0 & 0 \\
0 & 0 & -1 & -1 & 0 & -1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$

Graphes : définitions

Matrice d'incidence sommets-arcs : exemple

$$
\begin{array}{c}
 \\ E \\ M \\ N \\ SD \\ Se \\ S
\end{array}
\begin{array}{c}
\begin{array}{cccccccc} \text{MN} & \text{NE} & \text{ESD} & \text{NSD} & \text{SN} & \text{SSD} & \text{SSe} & \text{SeSD} \end{array} \\
\begin{pmatrix}
0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & 0 & 1 & -1 & 0 & 0 & 0 \\
0 & 0 & -1 & -1 & 0 & -1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$



Graphes : définitions

Matrice d'incidence sommets-arêtes : exemple

Graphes : définitions

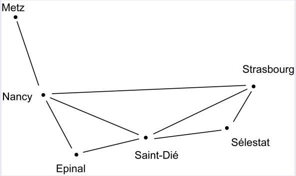
Matrice d'incidence sommets-arêtes : exemple

$$
\begin{array}{c}
\\ E \\ M \\ N \\ SD \\ Se \\ S
\end{array}
\begin{array}{c}
\begin{array}{cccccccc} \text{MN} & \text{NE} & \text{ESD} & \text{NSD} & \text{SN} & \text{SSD} & \text{SSe} & \text{SeSD} \end{array} \\
\begin{pmatrix}
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$

Graphes : définitions

Matrice d'incidence sommets-arêtes : exemple

$$\begin{array}{c c}
 & \begin{array}{cccccccc} MN & NE & ESD & NSD & SN & SSD & SSe & SeSD \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ S \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}
\end{array}$$



Graphes : définitions

Matrice d'adjacence ou matrice d'incidence sommets-sommets

Graphes : définitions

Matrice d'adjacence ou matrice d'incidence sommets-sommets

La matrice d'adjacence d'un 1-graphe $\mathcal{G} = (E, \mathcal{A})$ est la matrice $A = (a_{ij})$ carrée d'ordre n telle que $a_{ij} = 1$ si $(i, j) \in \mathcal{A}$, 0 sinon.

Graphes : définitions

Matrice d'adjacence ou matrice d'incidence sommets-sommets

La matrice d'adjacence d'un 1-graphe $\mathcal{G} = (E, \mathcal{A})$ est la matrice $A = (a_{ij})$ carrée d'ordre n telle que $a_{ij} = 1$ si $(i, j) \in \mathcal{A}$, 0 sinon.

Cas d'un graphe non orienté

Graphes : définitions

Matrice d'adjacence ou matrice d'incidence sommets-sommets

La matrice d'adjacence d'un 1-graphe $\mathcal{G} = (E, \mathcal{A})$ est la matrice $A = (a_{ij})$ carrée d'ordre n telle que $a_{ij} = 1$ si $(i, j) \in \mathcal{A}$, 0 sinon.

Cas d'un graphe non orienté

Dans le cas d'un graphe non orienté, on pose $a_{ij} = a_{ji} = 1$ si $\{i, j\} \in \mathcal{A}$, 0 sinon.

Graphes : définitions

Matrice d'adjacence

Graphes : définitions

Matrice d'adjacence

$$
\begin{array}{c c}
 & \begin{array}{cccccc} 1 & 2 & \cdots & j & \cdots & n \end{array} \\
\begin{array}{c} 1 \\ 2 \\ \cdots \\ i \\ \cdots \\ n \end{array} &
\begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2n} \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{in} \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
a_{n1} & a_{n2} & \cdots & a_{nj} & \cdots & a_{nn}
\end{pmatrix}
\end{array}
$$

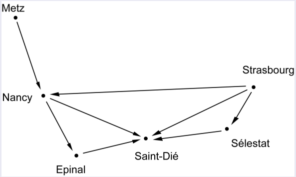
Graphes : définitions

Exemple : graphe orienté

Graphes : définitions

Exemple : graphe orienté

$$
\begin{array}{c c}
 & \begin{array}{cccccc} \mathrm{E} & \mathrm{M} & \mathrm{N} & \mathrm{SD} & \mathrm{Se} & \mathrm{Stg} \end{array} \\
\begin{array}{c} \mathrm{E} \\ \mathrm{M} \\ \mathrm{N} \\ \mathrm{SD} \\ \mathrm{Se} \\ \mathrm{Stg} \end{array} &
\begin{pmatrix}
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$



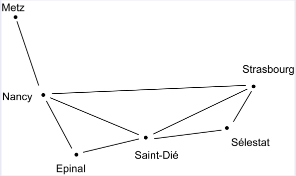
Graphes : définitions

Exemple : graphe non orienté

Graphes : définitions

Exemple : graphe non orienté

$$
\begin{array}{cc}
 & \begin{array}{cccccc} \mathrm{E} & \mathrm{M} & \mathrm{N} & \mathrm{SD} & \mathrm{Se} & \mathrm{Stg} \end{array} \\
\begin{array}{c} \mathrm{E} \\ \mathrm{M} \\ \mathrm{N} \\ \mathrm{SD} \\ \mathrm{Se} \\ \mathrm{Stg} \end{array} &
\begin{pmatrix}
0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$



Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Pour réduire le nombre d'informations et le stockage des 0 dans les matrices,

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Pour réduire le nombre d'informations et le stockage des 0 dans les matrices, on peut utiliser deux matrices lignes π et σ de longueur $n+1$ et $m+1$

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Pour réduire le nombre d'informations et le stockage des 0 dans les matrices, on peut utiliser deux matrices lignes π et σ de longueur $n+1$ et $m+1$ comportant la liste des positions $\pi(i)$ du premier successeur du sommet i

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Pour réduire le nombre d'informations et le stockage des 0 dans les matrices, on peut utiliser deux matrices lignes π et σ de longueur $n+1$ et $m+1$ comportant la liste des positions $\pi(i)$ du premier successeur du sommet i dans le tableau σ des successeurs des sommets 1, 2, ..., n.

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Pour réduire le nombre d'informations et le stockage des 0 dans les matrices, on peut utiliser deux matrices lignes π et σ de longueur $n+1$ et $m+1$ comportant la liste des positions $\pi(i)$ du premier successeur du sommet i dans le tableau σ des successeurs des sommets 1, 2, ..., n.
Si un sommet n'a pas de successeur, on passe au sommet suivant.

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Graphes : définitions

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Autre représentation : graphe orienté (liste de succession)

Remplissage

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Remplissage

On complète d'abord la matrice σ des successeurs des différents sommets.

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Remplissage

On complète d'abord la matrice σ des successeurs des différents sommets.
On pose $\pi(1) = 1$.

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Remplissage

On complète d'abord la matrice σ des successeurs des différents sommets.
On pose $\pi(1) = 1$.
Pour $i \in E$

1. si $\Gamma_i \neq \emptyset$ alors $\pi(i)$ est le numéro j de la case du premier successeur de i.

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Remplissage

On complète d'abord la matrice σ des successeurs des différents sommets.
On pose $\pi(1) = 1$.
Pour $i \in E$

1. si $\Gamma_i \neq \emptyset$ alors $\pi(i)$ est le numéro j de la case du premier successeur de i.
2. sinon $\pi(i) = \pi(i+1)$ (on passe au suivant).

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Remplissage

On complète d'abord la matrice σ des successeurs des différents sommets.
On pose $\pi(1) = 1$.
Pour $i \in E$

1. si $\Gamma_i \neq \emptyset$ alors $\pi(i)$ est le numéro j de la case du premier successeur de i.
2. sinon $\pi(i) = \pi(i+1)$ (on passe au suivant).

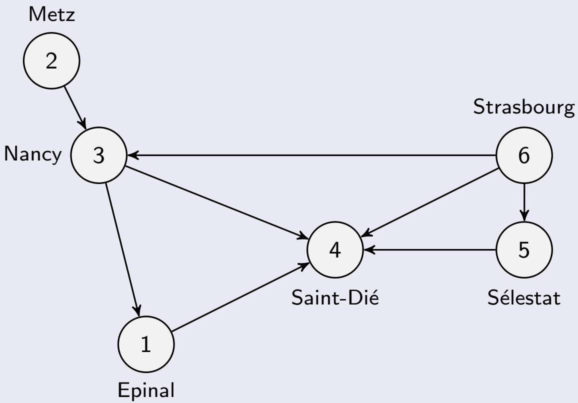
$\pi(n+1) = m+1$

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)



Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

Graphes : définitions

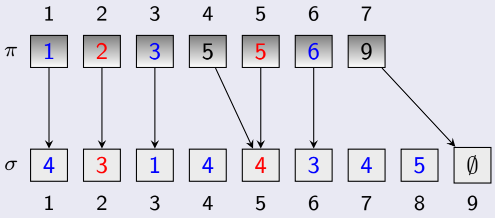
Autre représentation : graphe orienté (liste de succession)

$\Gamma_1 = \{4\}$, $\Gamma_2 = \{3\}$, $\Gamma_3 = \{1, 4\}$, $\Gamma_4 = \{\}$, $\Gamma_5 = \{4\}$, $\Gamma_6 = \{3, 4, 5\}$.

Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

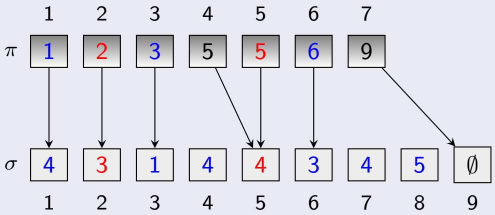
$\Gamma_1 = \{4\}$, $\Gamma_2 = \{3\}$, $\Gamma_3 = \{1, 4\}$, $\Gamma_4 = \{\}$, $\Gamma_5 = \{4\}$, $\Gamma_6 = \{3, 4, 5\}$.



Graphes : définitions

Autre représentation : graphe orienté (liste de succession)

$\Gamma_1 = \{4\}$, $\Gamma_2 = \{3\}$, $\Gamma_3 = \{1,4\}$, $\Gamma_4 = \{\}$, $\Gamma_5 = \{4\}$, $\Gamma_6 = \{3,4,5\}$.



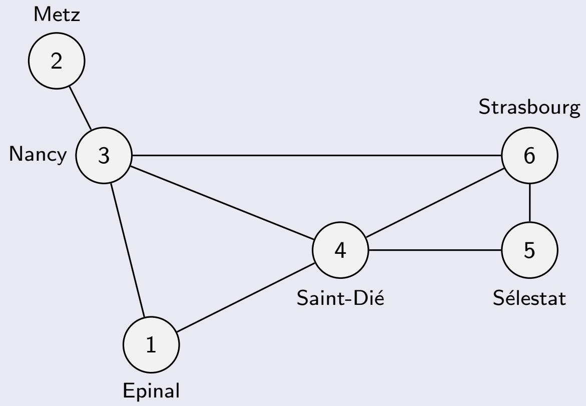
Remarque : $d^+(i) = \pi(i+1) - \pi(i)$

Graphes : définitions

Autre représentation : graphe non orienté (liste de succession)

Graphes : définitions

Autre représentation : graphe non orienté (liste de succession)



Graphes : définitions

Autre représentation : graphe non orienté (liste de succession)

Graphes : définitions

Autre représentation : graphe non orienté (liste de succession)

$\Gamma_1 = \{3, 4\}$, $\Gamma_2 = \{3\}$, $\Gamma_3 = \{1, 2, 4, 6\}$, $\Gamma_4 = \{1, 3, 5, 6\}$, $\Gamma_5 = \{4, 6\}$, $\Gamma_6 = \{3, 4, 5\}$.

Graphes : définitions

Autre représentation : graphe non orienté (liste de succession)

$\Gamma_1 = \{3,4\}$, $\Gamma_2 = \{3\}$, $\Gamma_3 = \{1,2,4,6\}$, $\Gamma_4 = \{1,3,5,6\}$, $\Gamma_5 = \{4,6\}$, $\Gamma_6 = \{3,4,5\}$.

	1	2	3	4	5	6	7
π	1	3	4	8	12	14	17

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
σ	3	4	3	1	2	4	6	1	3	5	6	4	6	3	4	5	$\emptyset$

Graphes : définitions

Autre représentation : graphe non orienté (liste de succession)

$\Gamma_1 = \{3,4\}$, $\Gamma_2 = \{3\}$, $\Gamma_3 = \{1,2,4,6\}$, $\Gamma_4 = \{1,3,5,6\}$, $\Gamma_5 = \{4,6\}$, $\Gamma_6 = \{3,4,5\}$.

	1	2	3	4	5	6	7
π	1	3	4	8	12	14	17

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
σ	3	4	3	1	2	4	6	1	3	5	6	4	6	3	4	5	$\emptyset$

Remarque : $d(i) = \pi(i+1) - \pi(i)$

Graphes : définitions

Chaîne

Graphes : définitions

Chaîne

Une chaîne de longueur q est une séquence de q arêtes $C = (\alpha_1, \alpha_2, ..., \alpha_q)$

Chaîne

Une chaîne de longueur q est une séquence de q arêtes $C = (\alpha_1, \alpha_2, \ldots, \alpha_q)$ telle que chaque arête α_k $(2 \leq k \leq q-1)$ ait une extrémité commune avec α_{k-1} et l'autre extrémité commune avec α_{k+1}.

Graphes : définitions

Chaîne

Une chaîne de longueur q est une séquence de q arêtes $C = (\alpha_1, \alpha_2, \ldots, \alpha_q)$ telle que chaque arête α_k $(2 \leq k \leq q-1)$ ait une extrémité commune avec α_{k-1} et l'autre extrémité commune avec α_{k+1}.
L'extrémité i de α_1 non adjacente à α_2, et l'extrémité j de α_q non adjacente à α_{q-1} sont appelées les extrémités de la chaîne C.

Graphes : définitions

Chaîne

Une chaîne de longueur q est une séquence de q arêtes $C = (\alpha_1, \alpha_2, ..., \alpha_q)$ telle que chaque arête α_k $(2 \leq k \leq q-1)$ ait une extrémité commune avec α_{k-1} et l'autre extrémité commune avec α_{k+1}.
L'extrémité i de α_1 non adjacente à α_2, et l'extrémité j de α_q non adjacente à α_{q-1} sont appelées les extrémités de la chaîne C.
On dit aussi que la chaîne C joint les sommets i et j.

Graphes : définitions

Exemple

Graphes : définitions

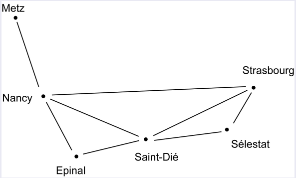
Exemple

$(\{Metz, Nancy\}, \{Strasbourg, Nancy\})$ est une chaîne de longueur 2 d'extrémités les sommets Metz et Strasbourg.

Graphes : définitions

Exemple

({Metz, Nancy}, {Strasbourg, Nancy}) est une chaîne de longueur 2 d'extrémités les sommets Metz et Strasbourg.
Autre notation : Metz - Nancy - Strasbourg.



Graphes : définitions

Chaîne élémentaire, cycle, chaîne simple

Graphes : définitions

Chaîne élémentaire, cycle, chaîne simple

Une chaîne élémentaire est une chaîne telle qu'en la parcourant on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).

Graphes : définitions

Chaîne élémentaire, cycle, chaîne simple

Une chaîne élémentaire est une chaîne telle qu'en la parcourant on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un cycle est une chaîne dont les extrémités coïncident.

Chaîne élémentaire, cycle, chaîne simple

Une chaîne élémentaire est une chaîne telle qu'en la parcourant on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un cycle est une chaîne dont les extrémités coïncident.
Un cycle élémentaire est un cycle tel qu'en le parcourant on ne rencontre pas deux fois le même sommet (sauf les extrémités).

Graphes : définitions

Chaîne élémentaire, cycle, chaîne simple

Une chaîne élémentaire est une chaîne telle qu'en la parcourant on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un cycle est une chaîne dont les extrémités coïncident.
Un cycle élémentaire est un cycle tel qu'en le parcourant on ne rencontre pas deux fois le même sommet (sauf les extrémités).
Une chaîne simple est une chaîne dont toutes les arêtes sont différentes.

Graphes : définitions

Chaîne élémentaire, cycle, chaîne simple

Une chaîne élémentaire est une chaîne telle qu'en la parcourant on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un cycle est une chaîne dont les extrémités coïncident.
Un cycle élémentaire est un cycle tel qu'en le parcourant on ne rencontre pas deux fois le même sommet (sauf les extrémités).
Une chaîne simple est une chaîne dont toutes les arêtes sont différentes.
Un cycle simple est un cycle tel qu'en le parcourant on ne rencontre pas deux fois la même arête.

Graphes : définitions

Exemple

Graphes : définitions

Exemple

$(\{\text{Metz}, \text{Nancy}\}, \{\text{Strasbourg}, \text{Nancy}\})$ est une chaîne élémentaire et simple.

Graphes : définitions

Exemple

$(\{\text{Metz}, \text{Nancy}\}, \{\text{Strasbourg}, \text{Nancy}\})$ est une chaîne élémentaire et simple.

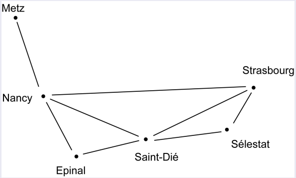
$(\{\text{Strasbourg}, \text{Nancy}\}, \{\text{Nancy}, \text{Saint-Dié}\}, \{\text{Saint-Dié}, \text{Sélestat}\}, \{\text{Sélestat}, \text{Strasbourg}\})$ est un cycle élémentaire et simple.

Graphes : définitions

Exemple

({Metz, Nancy}, {Strasbourg, Nancy}) est une chaîne élémentaire et simple.

({Strasbourg, Nancy}, {Nancy, Saint-Dié}, {Saint-Dié, Sélestat}, {Sélestat, Strasbourg}) est un cycle élémentaire et simple.



Graphes : définitions

Chemin

Graphes : définitions

Chemin

Un chemin de longueur q est une séquence de q arcs :
$C = (\alpha_1, \alpha_2, \cdots, \alpha_q)$, avec : $\alpha_1 = (i_0, i_1)$, $\alpha_2 = (i_1, i_2)$, ..., $\alpha_q = (i_{q-1}, i_q)$.

Graphes : définitions

Chemin

Un chemin de longueur q est une séquence de q arcs :
$C = (\alpha_1, \alpha_2, \cdots, \alpha_q)$, avec : $\alpha_1 = (i_0, i_1)$, $\alpha_2 = (i_1, i_2)$, ..., $\alpha_q = (i_{q-1}, i_q)$.
Remarque

Graphes : définitions

Chemin

Un chemin de longueur q est une séquence de q arcs :
$C = (\alpha_1, \alpha_2, \cdots, \alpha_q)$, avec : $\alpha_1 = (i_0, i_1)$, $\alpha_2 = (i_1, i_2)$, ..., $\alpha_q = (i_{q-1}, i_q)$.

Remarque

Un chemin est une chaîne dont tous les arcs sont orientés dans le même sens.

Graphes : définitions

Chemin

Un chemin de longueur q est une séquence de q arcs :
$C = (\alpha_1, \alpha_2, \cdots, \alpha_q)$, avec : $\alpha_1 = (i_0, i_1)$, $\alpha_2 = (i_1, i_2)$, ..., $\alpha_q = (i_{q-1}, i_q)$.

Remarque

Un chemin est une chaîne dont tous les arcs sont orientés dans le même sens.

Le sommet i_0 est l'extrémité initiale du chemin C.

Graphes : définitions

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié

Chemin

Un chemin de longueur q est une séquence de q arcs :
$C = (\alpha_1, \alpha_2, \cdots, \alpha_q)$, avec : $\alpha_1 = (i_0, i_1)$, $\alpha_2 = (i_1, i_2)$, ..., $\alpha_q = (i_{q-1}, i_q)$.

Remarque

Un chemin est une chaîne dont tous les arcs sont orientés dans le même sens.
Le sommet i_0 est l'extrémité initiale du chemin C.
Le sommet i_q est son extrémité terminale.

Graphes : définitions

Chemin

Un chemin de longueur q est une séquence de q arcs :
$C = (\alpha_1, \alpha_2, \cdots, \alpha_q)$, avec : $\alpha_1 = (i_0, i_1)$, $\alpha_2 = (i_1, i_2)$, ..., $\alpha_q = (i_{q-1}, i_q)$.

Remarque

Un chemin est une chaîne dont tous les arcs sont orientés dans le même sens.
Le sommet i_0 est l'extrémité initiale du chemin C.
Le sommet i_q est son extrémité terminale.
Dans le cas d'un graphe simple, un chemin est entièrement défini par la succession des sommets qu'il rencontre (dans l'ordre).

Graphes : définitions

Exemple

Graphes : définitions

Exemple

((Strasbourg, Sélestat), (Sélestat, Saint-Dié)) est un chemin de longueur 2 allant du sommet Strasbourg au sommet Saint-Dié.

Graphes : définitions

Exemple

((Strasbourg, Sélestat), (Sélestat, Saint-Dié)) est un chemin de longueur 2 allant du sommet Strasbourg au sommet Saint-Dié.
Notation : Strasbourg - Sélestat - Saint-Dié.

Graphes : définitions

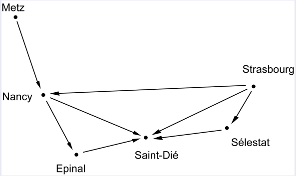
Exemple

$((\text{Strasbourg}, \text{Sélestat}), (\text{Sélestat}, \text{Saint-Dié}))$ est un chemin de longueur 2 allant du sommet Strasbourg au sommet Saint-Dié.
Notation : Strasbourg - Sélestat - Saint-Dié.
Mais $((\text{Metz}, \text{Nancy}), (\text{Strasbourg}, \text{Nancy}))$ n'est pas un chemin.

Graphes : définitions

Exemple

((Strasbourg, Sélestat), (Sélestat, Saint-Dié)) est un chemin de longueur 2 allant du sommet Strasbourg au sommet Saint-Dié.
Notation : Strasbourg - Sélestat - Saint-Dié.
Mais ((Metz, Nancy), (Strasbourg, Nancy)) n'est pas un chemin.



Graphes : définitions

Chemin élémentaire, circuit, chemin simple

Graphes : définitions

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Chemin élémentaire, circuit, chemin simple

Un chemin élémentaire est un chemin tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).

Graphes : définitions

Chemin élémentaire, circuit, chemin simple

Un chemin élémentaire est un chemin tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un circuit est un chemin dont les extrémités coïncident.

Graphes : définitions

Chemin élémentaire, circuit, chemin simple

Un chemin élémentaire est un chemin tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un circuit est un chemin dont les extrémités coïncident.
Un circuit élémentaire est un circuit tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf les extrémités).

Graphes : définitions

Chemin élémentaire, circuit, chemin simple

Un chemin élémentaire est un chemin tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un circuit est un chemin dont les extrémités coïncident.
Un circuit élémentaire est un circuit tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf les extrémités).
Un chemin simple est un chemin dont tous arcs sont différents.

Graphes : définitions

Chemin élémentaire, circuit, chemin simple

Un chemin élémentaire est un chemin tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf éventuellement les extrémités).
Un circuit est un chemin dont les extrémités coïncident.
Un circuit élémentaire est un circuit tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet (sauf les extrémités).
Un chemin simple est un chemin dont tous arcs sont différents.
Un circuit simple est un cycle tel qu'en le parcourant on ne rencontre pas deux fois le même arc.

Graphes : définitions

Chemin élémentaire, circuit

Graphes : définitions

Chemin élémentaire, circuit

((Strasbourg, Sélestat), (Sélestat, Saint-Dié)) est un chemin élémentaire et simple de longueur 2 allant du sommet Strasbourg au sommet Saint-Dié:

Graphes : définitions

Chemin élémentaire, circuit

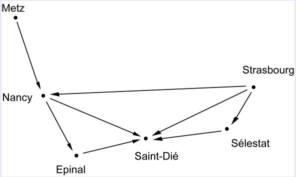
((Strasbourg, Sélestat), (Sélestat, Saint-Dié)) est un chemin élémentaire et simple de longueur 2 allant du sommet Strasbourg au sommet Saint-Dié: Strasbourg - Sélestat - Saint-Dié.

Graphes : définitions

Chemin élémentaire, circuit

((Strasbourg, Sélestat), (Sélestat, Saint-Dié)) est un chemin élémentaire et simple de longueur 2 allant du sommet Strasbourg au sommet Saint-Dié: Strasbourg - Sélestat - Saint-Dié.

Le graphe ne comporte pas de circuit.



Graphes : définitions

Source, puits

Graphes : définitions

Source, puits

Dans un graphe orienté, une source est un sommet sans prédécesseur.

Graphes : définitions

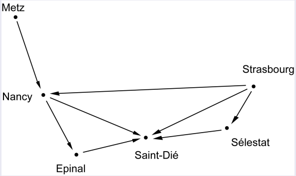
Source, puits

Dans un graphe orienté, une source est un sommet sans prédécesseur.
Un puits est un sommet sans succcesseur.

Graphes : définitions

Source, puits

Dans un graphe orienté, une source est un sommet sans prédécesseur.
Un puits est un sommet sans succcesseur.
Exemple : Metz et Strasbourg sont des sources. Saint-Dié est un puits.



Graphes : définitions

Source, puits et circuit

Graphes : définitions

Source, puits et circuit

Propriété : Tout graphe sans circuit possède au moins une source et un puits.

Source, puits et circuit

Propriété : Tout graphe sans circuit possède au moins une source et un puits.

Démonstration :

Graphes : définitions

Source, puits et circuit

Propriété : Tout graphe sans circuit possède au moins une source et un puits.

Démonstration :

Soit C un chemin de G qui soit maximal :

Graphes : définitions

Source, puits et circuit

Propriété : Tout graphe sans circuit possède au moins une source et un puits.

Démonstration :

Soit C un chemin de G qui soit maximal : C=$(x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik})$ est tel qu'il n'existe pas de sommet x de G tel que $(x, x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik})$ ou $(x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik}, x)$ soient des chemins de C.

Graphes : définitions

Source, puits et circuit

Propriété : Tout graphe sans circuit possède au moins une source et un puits.

Démonstration :

Soit C un chemin de G qui soit maximal : C=$(x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik})$ est tel qu'il n'existe pas de sommet x de G tel que $(x, x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik})$ ou $(x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik}, x)$ soient des chemins de C.

Un tel chemin existe puisque G est sans circuit.

Source, puits et circuit

Propriété : Tout graphe sans circuit possède au moins une source et un puits.

Démonstration :
Soit C un chemin de G qui soit maximal : C=$(x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik})$ est tel qu'il n'existe pas de sommet x de G tel que $(x, x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik})$ ou $(x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik}, x)$ soient des chemins de C.
Un tel chemin existe puisque G est sans circuit.
Cela signifie que x_{i0} est une source et x_{ik} est un puits.

Graphes : définitions

Distance, diamètre

Graphes : définitions

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Distance, diamètre

La distance $d(i,j)$ d'un sommet i à un sommet j est la longueur du plus court chemin (respectivement chaîne) de i à j.

Graphes : définitions

Distance, diamètre

La distance $d(i,j)$ d'un sommet i à un sommet j est la longueur du plus court chemin (respectivement chaîne) de i à j.
On pose $d(i,i) = 0$

Graphes : définitions

Distance, diamètre

La **distance** $d(i,j)$ d'un sommet i à un sommet j est la longueur du plus court chemin (respectivement chaîne) de i à j.
On pose $d(i,i) = 0$ et $d(i,j) = \infty$ s'il n'existe pas de chemin (respectivement chaîne) de i à j.

Graphes : définitions

Distance, diamètre

La distance $d(i, j)$ d'un sommet i à un sommet j est la longueur du plus court chemin (respectivement chaîne) de i à j.
On pose $d(i, i) = 0$ et $d(i, j) = \infty$ s'il n'existe pas de chemin (respectivement chaîne) de i à j.
Le diamètre d'un graphe est la plus grande distance entre deux sommets du graphe :

Graphes : définitions

Distance, diamètre

La distance $d(i,j)$ d'un sommet i à un sommet j est la longueur du plus court chemin (respectivement chaîne) de i à j.
On pose $d(i,i) = 0$ et $d(i,j) = \infty$ s'il n'existe pas de chemin (respectivement chaîne) de i à j.
Le diamètre d'un graphe est la plus grande distance entre deux sommets du graphe :

$$\delta(\mathcal{G}) = \max_{i,j \in E} d(i,j)$$

Graphes : définitions

Écartement, rayon, centre

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

Graphes : définitions

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

$$e(i) = \max_{j \in E} d(i,j)$$

Graphes : définitions

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

$e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe

Graphes : définitions

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

$e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe

$\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Graphes : définitions

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

$e(i) = \max_{j \in E} d(i, j)$

Rayon d'un graphe

$\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Un centre : tout sommet d'écartement minimal.

Graphes : définitions

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

$$e(i) = \max_{j \in E} d(i,j)$$

Rayon d'un graphe

$$\rho(\mathcal{G}) = \min_{i \in E} e(i)$$

Un centre : tout sommet d'écartement minimal.

Le centre : l'ensemble des centres.

Graphes : définitions

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

$$e(i) = \max_{j \in E} d(i,j)$$

Rayon d'un graphe

$$\rho(\mathcal{G}) = \min_{i \in E} e(i)$$

Un centre : tout sommet d'écartement minimal.

Le centre : l'ensemble des centres.

Remarque :

Graphes : définitions

Écartement, rayon, centre

Écartement d'un sommet (ou excentricité)

$e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe

$\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Un centre : tout sommet d'écartement minimal.

Le centre : l'ensemble des centres.

Remarque :

$\delta(\mathcal{G}) = \max_{i \in E} e(i)$

Graphes : définitions

Distance, diamètre

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :
$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :
$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$
$d\,(\text{Saint-Dié}, \text{Strasbourg}) = 1$

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :

$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

$d\,(\text{Saint-Dié}, \text{Strasbourg}) = 1$

$d\,(\text{Strasbourg}, \text{Epinal}) = 2$

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :

$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

$d\,(\text{Saint-Dié}, \text{Strasbourg}) = 1$

$d\,(\text{Strasbourg}, \text{Epinal}) = 2$

$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :

$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

$d\,(\text{Saint-Dié}, \text{Strasbourg}) = 1$

$d\,(\text{Strasbourg}, \text{Epinal}) = 2$

$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$

$d\,(\text{Sélestat}, \text{Metz}) = 3$

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :

$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

$d\,(\text{Saint-Dié}, \text{Strasbourg}) = 1$

$d\,(\text{Strasbourg}, \text{Epinal}) = 2$

$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$

$d\,(\text{Sélestat}, \text{Metz}) = 3$

$\delta\,(\mathcal{G}) = 3$

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :

$d\left(\text{Strasbourg}, \text{Saint-Dié}\right) = 1$

$d\left(\text{Saint-Dié}, \text{Strasbourg}\right) = 1$

$d\left(\text{Strasbourg}, \text{Epinal}\right) = 2$

$d\left(\text{Strasbourg}, \text{Strasbourg}\right) = 0$

$d\left(\text{Sélestat}, \text{Metz}\right) = 3$

$\delta\left(\mathcal{G}\right) = 3$

Écartement minimal 2 pour Épinal, Nancy, Saint-Dié et Strasbourg.

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :

$d\left(\text{Strasbourg}, \text{Saint-Dié}\right) = 1$

$d\left(\text{Saint-Dié}, \text{Strasbourg}\right) = 1$

$d\left(\text{Strasbourg}, \text{Epinal}\right) = 2$

$d\left(\text{Strasbourg}, \text{Strasbourg}\right) = 0$

$d\left(\text{Sélestat}, \text{Metz}\right) = 3$

$\delta\left(\mathcal{G}\right) = 3$

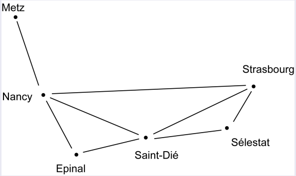
Écartement minimal 2 pour Épinal, Nancy, Saint-Dié et Strasbourg.

$\rho\left(\mathcal{G}\right) = 2.$

Graphes : définitions

Distance, diamètre

Dans le graphe non orienté des accès à Saint-Dié :
$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$
$d\,(\text{Saint-Dié}, \text{Strasbourg}) = 1$
$d\,(\text{Strasbourg}, \text{Epinal}) = 2$
$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$
$d\,(\text{Sélestat}, \text{Metz}) = 3$
$\delta\,(\mathcal{G}) = 3$
Écartement minimal 2 pour Épinal, Nancy, Saint-Dié et Strasbourg.
$\rho\,(\mathcal{G}) = 2$. Épinal, Nancy, Saint-Dié et Strasbourg sont les centres.



Graphes : définitions

Distance, diamètre

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :
$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :
$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$
$d\,(\text{Saint-Dié}, \text{Strasbourg}) = \infty$

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :

$d\left(\text{Strasbourg}, \text{Saint-Dié}\right) = 1$

$d\left(\text{Saint-Dié}, \text{Strasbourg}\right) = \infty$

$d\left(\text{Strasbourg}, \text{Epinal}\right) = 2$

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :

$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

$d\,(\text{Saint-Dié}, \text{Strasbourg}) = \infty$

$d\,(\text{Strasbourg}, \text{Epinal}) = 2$

$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :

$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

$d\,(\text{Saint-Dié}, \text{Strasbourg}) = \infty$

$d\,(\text{Strasbourg}, \text{Epinal}) = 2$

$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$

$d\,(\text{Strasbourg}, \text{Metz}) = \infty$

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :

$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$

$d\,(\text{Saint-Dié}, \text{Strasbourg}) = \infty$

$d\,(\text{Strasbourg}, \text{Epinal}) = 2$

$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$

$d\,(\text{Strasbourg}, \text{Metz}) = \infty$

Tous les sommets ont un écartement infini et sont les centres.

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :
$d\,(\text{Strasbourg}, \text{Saint-Dié}) = 1$
$d\,(\text{Saint-Dié}, \text{Strasbourg}) = \infty$
$d\,(\text{Strasbourg}, \text{Epinal}) = 2$
$d\,(\text{Strasbourg}, \text{Strasbourg}) = 0$
$d\,(\text{Strasbourg}, \text{Metz}) = \infty$
Tous les sommets ont un écartement infini et sont les centres.
$\delta\,(\mathcal{G}) = \rho\,(\mathcal{G}) = \infty$

Graphes : définitions

Distance, diamètre

Dans le graphe orienté des accès à Saint-Dié :

$d\left(\text{Strasbourg}, \text{Saint-Dié}\right) = 1$

$d\left(\text{Saint-Dié}, \text{Strasbourg}\right) = \infty$

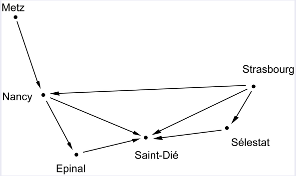
$d\left(\text{Strasbourg}, \text{Epinal}\right) = 2$

$d\left(\text{Strasbourg}, \text{Strasbourg}\right) = 0$

$d\left(\text{Strasbourg}, \text{Metz}\right) = \infty$

Tous les sommets ont un écartement infini et sont les centres.

$\delta\left(\mathcal{G}\right) = \rho\left(\mathcal{G}\right) = \infty$



Graphes : définitions

Nombre de chemins de longueur fixée

Graphes : définitions

Nombre de chemins de longueur fixée

Théorème : Soit $A = (a_{ij})$ la matrice d'adjacence d'un graphe (orienté ou non, éventuellement généralisée à un p-graphes ou un multigraphe).

Graphes : définitions

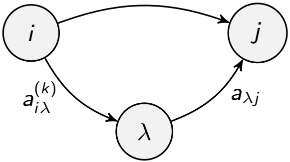
Nombre de chemins de longueur fixée

Théorème : Soit $A = (a_{ij})$ la matrice d'adjacence d'un graphe (orienté ou non, éventuellement généralisée à un p-graphes ou un multigraphe).
$A^k = \left(a_{ij}^{(k)}\right)$ est la matrice des nombres de chemins (respectivement de chaînes) de longueur k qui mènent de i à j.

Graphes : définitions

Nombre de chemins de longueur fixée

Théorème : Soit $A = (a_{ij})$ la matrice d'adjacence d'un graphe (orienté ou non, éventuellement généralisée à un p-graphes ou un multigraphe).
$A^k = \left(a_{ij}^{(k)}\right)$ est la matrice des nombres de chemins (respectivement de chaînes) de longueur k qui mènent de i à j.



Graphes : définitions

Démonstration (graphe orienté d'ordre n)

Graphes : définitions

Démonstration (graphe orienté d'ordre n)

La démonstration peut se faire par récurrence.

Graphes : définitions

Démonstration (graphe orienté d'ordre n)

La démonstration peut se faire par récurrence.

$k = 1$ correspond à la définition de la matrice d'adjacence $A = \left(a_{ij}^{(1)}\right)$.

Graphes : définitions

Démonstration (graphe orienté d'ordre n)

La démonstration peut se faire par récurrence.

$k = 1$ correspond à la définition de la matrice d'adjacence $A = \left(a_{ij}^{(1)}\right)$.

$A^k = \left(a_{ij}^{(k)}\right)$ est la matrice des nombres de chemin de longueur k qui mènent de i à j.

Graphes : définitions

Démonstration (graphe orienté d'ordre n)

La démonstration peut se faire par récurrence.

$k = 1$ correspond à la définition de la matrice d'adjacence $A = \left(a_{ij}^{(1)}\right)$.

$A^k = \left(a_{ij}^{(k)}\right)$ est la matrice des nombres de chemin de longueur k qui mènent de i à j.

Un chemin de i à j de longueur $k+1$ est un chemin de i à un sommet λ de longueur k suivi d'un arc (λ, j).

Graphes : définitions

Démonstration (graphe orienté d'ordre n)

La démonstration peut se faire par récurrence.

$k = 1$ correspond à la définition de la matrice d'adjacence $A = \left(a_{ij}^{(1)}\right)$.

$A^k = \left(a_{ij}^{(k)}\right)$ est la matrice des nombres de chemin de longueur k qui mènent de i à j.

Un chemin de i à j de longueur $k+1$ est un chemin de i à un sommet λ de longueur k suivi d'un arc (λ, j).

Le nombre $a_{ij}^{(k+1)}$ de chemins de i à j de longueur $k+1$ est donc $\displaystyle\sum_{\lambda=1}^{n} a_{i\lambda}^{(k)} a_{\lambda j}^{(1)}$.

Graphes : définitions

Démonstration (graphe orienté d'ordre n)

La démonstration peut se faire par récurrence.

$k = 1$ correspond à la définition de la matrice d'adjacence $A = \left(a_{ij}^{(1)}\right)$.

$A^k = \left(a_{ij}^{(k)}\right)$ est la matrice des nombres de chemin de longueur k qui mènent de i à j.

Un chemin de i à j de longueur $k+1$ est un chemin de i à un sommet λ de longueur k suivi d'un arc (λ, j).

Le nombre $a_{ij}^{(k+1)}$ de chemins de i à j de longueur $k+1$ est donc $\sum\limits_{\lambda=1}^{n} a_{i\lambda}^{(k)} a_{\lambda j}^{(1)}$.

Pour un graphe non orienté, on remplace chemin par chaîne et arc par arête.

Graphes : définitions

Démonstration (graphe orienté d'ordre n)

La démonstration peut se faire par récurrence.

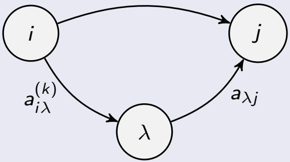
$k = 1$ correspond à la définition de la matrice d'adjacence $A = \left(a_{ij}^{(1)}\right)$.

$A^k = \left(a_{ij}^{(k)}\right)$ est la matrice des nombres de chemin de longueur k qui mènent de i à j.

Un chemin de i à j de longueur $k+1$ est un chemin de i à un sommet λ de longueur k suivi d'un arc (λ, j).

Le nombre $a_{ij}^{(k+1)}$ de chemins de i à j de longueur $k+1$ est donc $\sum_{\lambda=1}^{n} a_{i\lambda}^{(k)} a_{\lambda j}^{(1)}$.

Pour un graphe non orienté, on remplace chemin par chaîne et arc par arête.

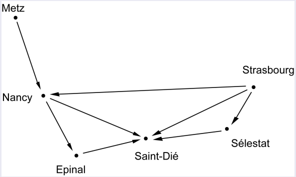


Nombre de chemins : exemple

Graphes : définitions

Nombre de chemins : exemple

$$A = \begin{array}{cc} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

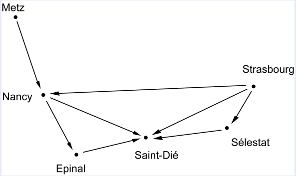


Nombre de chemins : exemple

Graphes : définitions

Nombre de chemins : exemple

$$A^2 = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 & 0 \end{pmatrix} \end{array}$$



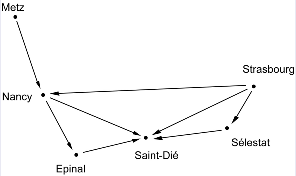
Graphes : définitions

Nombre de chemins : exemple

Graphes : définitions

Nombre de chemins : exemple

$$A^3 = \begin{array}{c} \\ \mathrm{E} \\ \mathrm{M} \\ \mathrm{N} \\ \mathrm{SD} \\ \mathrm{Se} \\ \mathrm{Stg} \end{array} \begin{array}{c} \begin{array}{cccccc} \mathrm{E} & \mathrm{M} & \mathrm{N} & \mathrm{SD} & \mathrm{Se} & \mathrm{Stg} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$



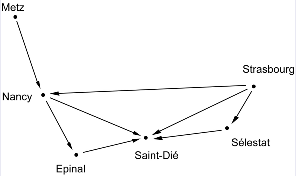
Graphes : définitions

Nombre de chemins : exemple

Graphes : définitions

Nombre de chemins : exemple

$$A^4 = \begin{array}{c|c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$



Graphes : définitions

Circuits

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Exemple

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Exemple

Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.
Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.
Exemple
Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.
Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.
Exemple
Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Exemple

Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.

Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n - 1$.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Exemple

Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.

Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n - 1$.

En effet, tout chemin de longueur au moins n comporte au moins $n + 1$ sommets (et donc deux fois un sommet).

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.
Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.
Exemple
Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.
Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n - 1$.
En effet, tout chemin de longueur au moins n comporte au moins $n + 1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.
Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.
Exemple
Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.
Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n-1$.
En effet, tout chemin de longueur au moins n comporte au moins $n+1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.
Réciproquement, si un graphe a au moins un circuit de longueur k_0,

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Exemple

Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.

Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n - 1$.

En effet, tout chemin de longueur au moins n comporte au moins $n + 1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.

Réciproquement, si un graphe a au moins un circuit de longueur k_0, alors pour tout entier $k > k_0$ il existe un chemin de longueur k

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Exemple

Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.

Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n-1$.

En effet, tout chemin de longueur au moins n comporte au moins $n+1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.

Réciproquement, si un graphe a au moins un circuit de longueur k_0, alors pour tout entier $k > k_0$ il existe un chemin de longueur k (il suffit de considérer les sommets du circuit).

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.
Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.
Exemple
Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.
Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n-1$.
En effet, tout chemin de longueur au moins n comporte au moins $n+1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.
Réciproquement, si un graphe a au moins un circuit de longueur k_0, alors pour tout entier $k > k_0$ il existe un chemin de longueur k (il suffit de considérer les sommets du circuit). Donc $A^k \neq 0$ pour $k \geq k_0$.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.
Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.
Exemple
Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.
Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n-1$.
En effet, tout chemin de longueur au moins n comporte au moins $n+1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.
Réciproquement, si un graphe a au moins un circuit de longueur k_0, alors pour tout entier $k > k_0$ il existe un chemin de longueur k (il suffit de considérer les sommets du circuit). Donc $A^k \neq 0$ pour $k \geq k_0$.
De plus, pour tout entier k, $1 \leq k < k_0$, $A^k \neq 0$. Sinon, $A^{k_0} = (0)$.

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.
Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.
Exemple
Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.
Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n-1$.
En effet, tout chemin de longueur au moins n comporte au moins $n+1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.
Réciproquement, si un graphe a au moins un circuit de longueur k_0, alors pour tout entier $k > k_0$ il existe un chemin de longueur k (il suffit de considérer les sommets du circuit). Donc $A^k \neq 0$ pour $k \geq k_0$.
De plus, pour tout entier k, $1 \leq k < k_0$, $A^k \neq 0$. Sinon, $A^{k_0} = (0)$.
On a donc :

Graphes : définitions

Circuits

Une matrice carrée A est dite nilpotente s'il existe un entier $k \geq 1$ tel que $A^k = 0$.

Théorème : Un graphe orienté $\mathcal{G}$ est sans circuit si et seulement si sa matrice d'adjacence est nilpotente.

Exemple

Le graphe des accès à Saint-Dié est sans circuit : $A^4 = 0$.

Démonstration

Soit $\mathcal{G}$ un graphe d'ordre n.

Si le graphe est sans circuit, les chemins ont une longueur qui ne peut dépasser $n - 1$.

En effet, tout chemin de longueur au moins n comporte au moins $n + 1$ sommets (et donc deux fois un sommet). Donc $A^n = 0$.

Réciproquement, si un graphe a au moins un circuit de longueur k_0, alors pour tout entier $k > k_0$ il existe un chemin de longueur k (il suffit de considérer les sommets du circuit). Donc $A^k \neq 0$ pour $k \geq k_0$.

De plus, pour tout entier k, $1 \leq k < k_0$, $A^k \neq 0$. Sinon, $A^{k_0} = (0)$.

On a donc :

Si un graphe a au moins un circuit alors $\forall k \geq 1$ $A^k \neq 0$: la matrice d'adjacence n'est pas nilpotente..

Fermeture transitive

Graphes : définitions

Fermeture transitive

On appelle fermeture transitive l'application $\widehat{\Gamma}$ définie par $\widehat{\Gamma}_i = \{i\} \cup \Gamma_i \cup \Gamma_i^2 \cup \cdots \Gamma_i^{n-1}$, Γ_i^k représentant l'ensemble des sommets que l'on peut atteindre à partir du sommet i par des chemins ayant exactement k arcs.

Graphes : définitions

Fermeture transitive

On appelle fermeture transitive l'application $\widehat{\Gamma}$ définie par $\widehat{\Gamma}_i = \{i\} \cup \Gamma_i \cup \Gamma_i^2 \cup \cdots \Gamma_i^{n-1}$, Γ_i^k représentant l'ensemble des sommets que l'on peut atteindre à partir du sommet i par des chemins ayant exactement k arcs.

Remarque : $\Gamma_i = \Gamma_i^1$.

Graphes : définitions

Fermeture transitive

On appelle fermeture transitive l'application $\widehat{\Gamma}$ définie par $\widehat{\Gamma}_i = \{i\} \cup \Gamma_i \cup \Gamma_i^2 \cup \cdots \Gamma_i^{n-1}$, Γ_i^k représentant l'ensemble des sommets que l'on peut atteindre à partir du sommet i par des chemins ayant exactement k arcs.

Remarque : $\Gamma_i = \Gamma_i^1$.

On en déduit que, puisque tout chemin élémentaire a au plus $n-1$ arcs, $\widehat{\Gamma}_i$ représente l'ensemble des sommets que l'on peut atteindre par un chemin à partir du sommet i.

Graphes : définitions

Fermeture transitive

On appelle **fermeture transitive** l'application $\widehat{\Gamma}$ définie par $\widehat{\Gamma}_i = \{i\} \cup \Gamma_i \cup \Gamma_i^2 \cup \cdots \Gamma_i^{n-1}$, Γ_i^k représentant l'ensemble des sommets que l'on peut atteindre à partir du sommet i par des chemins ayant exactement k arcs.
Remarque : $\Gamma_i = \Gamma_i^1$.
On en déduit que, puisque tout chemin élémentaire a au plus $n-1$ arcs, $\widehat{\Gamma}_i$ représente l'ensemble des sommets que l'on peut atteindre par un chemin à partir du sommet i.
On dit que $\widehat{\Gamma}_i$ est l'ensemble des **descendants** de i.

Graphes : définitions

Fermeture transitive

On appelle fermeture transitive l'application $\widehat{\Gamma}$ définie par $\widehat{\Gamma}_i = \{i\} \cup \Gamma_i \cup \Gamma_i^2 \cup \cdots \Gamma_i^{n-1}$, Γ_i^k représentant l'ensemble des sommets que l'on peut atteindre à partir du sommet i par des chemins ayant exactement k arcs.

Remarque : $\Gamma_i = \Gamma_i^1$.

On en déduit que, puisque tout chemin élémentaire a au plus $n-1$ arcs, $\widehat{\Gamma}_i$ représente l'ensemble des sommets que l'on peut atteindre par un chemin à partir du sommet i.

On dit que $\widehat{\Gamma}_i$ est l'ensemble des descendants de i.

De la même manière $\widehat{\Gamma}_i^{-1}$ représente l'ensemble des ancêtres de i.

Graphes : définitions

Exemple

Graphes : définitions

Exemple

$\Gamma_{Strasbourg} = \{\text{Sélestat}, \text{Nancy}, \text{Saint-Dié}\}$

Graphes : définitions

Exemple

$$\Gamma_{Strasbourg} = \{\text{Sélestat, Nancy, Saint-Dié}\}$$
$$\Gamma^2_{Strasbourg} = \{\text{Saint-Dié,Epinal}\}$$

Graphes : définitions

Exemple

$$\begin{aligned}
\Gamma_{Strasbourg} &= \{\text{Sélestat, Nancy, Saint-Dié}\} \\
\Gamma^2_{Strasbourg} &= \{\text{Saint-Dié,Epinal}\} \\
\Gamma^3_{Strasbourg} &= \{\text{Saint-Dié}\}
\end{aligned}$$

Graphes : définitions

Exemple

$$\Gamma_{Strasbourg} = \{\text{Sélestat, Nancy, Saint-Dié}\}$$
$$\Gamma^2_{Strasbourg} = \{\text{Saint-Dié,Epinal}\}$$
$$\Gamma^3_{Strasbourg} = \{\text{Saint-Dié}\}$$
$$\Gamma^4_{Strasbourg} = \{\} = \Gamma^5_{Strasbourg}$$

Graphes : définitions

Exemple

$$\Gamma_{Strasbourg} = \{\text{Sélestat},\ \text{Nancy},\ \text{Saint-Dié}\}$$
$$\Gamma^2_{Strasbourg} = \{\text{Saint-Dié},\text{Epinal}\}$$
$$\Gamma^3_{Strasbourg} = \{\text{Saint-Dié}\}$$
$$\Gamma^4_{Strasbourg} = \{\} = \Gamma^5_{Strasbourg}$$
$$\widehat{\Gamma}_{Strasbourg} = \{Strasbourg\} \cup \Gamma_{Strasbourg} \cup \Gamma^2_{Strasbourg} \cup \Gamma^3_{Strasbourg} \cup \Gamma^4_{Strasbourg} \cup \Gamma^5_{Strasbourg}$$

Graphes : définitions

Exemple

$\Gamma_{Strasbourg} = \{\text{Sélestat, Nancy, Saint-Dié}\}$

$\Gamma^2_{Strasbourg} = \{\text{Saint-Dié,Epinal}\}$

$\Gamma^3_{Strasbourg} = \{\text{Saint-Dié}\}$

$\Gamma^4_{Strasbourg} = \{\} = \Gamma^5_{Strasbourg}$

$\widehat{\Gamma}_{Strasbourg} = \{Strasbourg\} \cup \Gamma_{Strasbourg} \cup \Gamma^2_{Strasbourg} \cup \Gamma^3_{Strasbourg} \cup \Gamma^4_{Strasbourg} \cup \Gamma^5_{Strasbourg}$

$\widehat{\Gamma}_{Strasbourg} = \{\text{Strasbourg, Sélestat, Nancy, Saint-Dié, Epinal}\}.$

Graphes : définitions

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

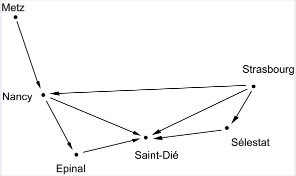
Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Exemple

$$\Gamma_{Strasbourg} = \{\text{Sélestat, Nancy, Saint-Dié}\}$$
$$\Gamma^2_{Strasbourg} = \{\text{Saint-Dié,Epinal}\}$$
$$\Gamma^3_{Strasbourg} = \{\text{Saint-Dié}\}$$
$$\Gamma^4_{Strasbourg} = \{\} = \Gamma^5_{Strasbourg}$$
$$\widehat{\Gamma}_{Strasbourg} = \{Strasbourg\} \cup \Gamma_{Strasbourg} \cup \Gamma^2_{Strasbourg} \cup \Gamma^3_{Strasbourg} \cup \Gamma^4_{Strasbourg} \cup \Gamma^5_{Strasbourg}$$
$$\widehat{\Gamma}_{Strasbourg} = \{\text{Strasbourg, Sélestat, Nancy, Saint-Dié, Epinal}\}.$$



Graphes : définitions

Connexité

Graphes : définitions

Connexité

Un graphe est dit connexe si, pour tout couple de sommets (i, j), il existe une chaîne joignant i et j.

Graphes : définitions

Connexité

Un graphe est dit connexe si, pour tout couple de sommets (i, j), il existe une chaîne joignant i et j.

Exemple

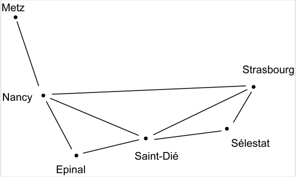
Graphes : définitions

Connexité

Un graphe est dit connexe si, pour tout couple de sommets (i, j), il existe une chaîne joignant i et j.

Exemple

Le graphe des accès possibles à Saint-Dié à partir des grandes villes périphériques est connexe (graphe d'un seul tenant).



Connexité

Connexité

La relation $i\mathfrak{R}j$ si et seulement si "soit $i = j$, soit il existe une chaîne joignant i et j" est une relation d'équivalence.

Graphes : définitions

Connexité

La relation $i\mathfrak{R}j$ si et seulement si "soit $i = j$, soit il existe une chaîne joignant i et j" est une relation d'équivalence.
Le nombre p de classes d'équivalence est appelé le nombre de connexité du graphe.

Graphes : définitions

Connexité

La relation $i\mathfrak{R}j$ si et seulement si "soit $i = j$, soit il existe une chaîne joignant i et j" est une relation d'équivalence.
Le nombre p de classes d'équivalence est appelé le nombre de connexité du graphe.
Un graphe est connexe si et seulement si son nombre de connexité est égal à 1.

Graphes : définitions

Connexité

La relation $i\mathfrak{R}j$ si et seulement si "soit $i = j$, soit il existe une chaîne joignant i et j" est une relation d'équivalence.
Le nombre p de classes d'équivalence est appelé le nombre de connexité du graphe.
Un graphe est connexe si et seulement si son nombre de connexité est égal à 1.
Les sous-graphes $\mathcal{G}_1$, $\mathcal{G}_2$, ..., $\mathcal{G}_p$ engendrés par les classes d'équivalence sont appelés les composantes connexes ou classes de connexité du graphe $\mathcal{G}$.

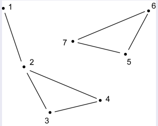
Graphes : définitions

Exemple

Graphes : définitions

Exemple

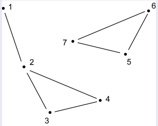
Exemple de graphe non-connexe (nombre de connexité $N = 2$) :



Graphes : définitions

Exemple

Exemple de graphe non-connexe (nombre de connexité $N = 2$) :



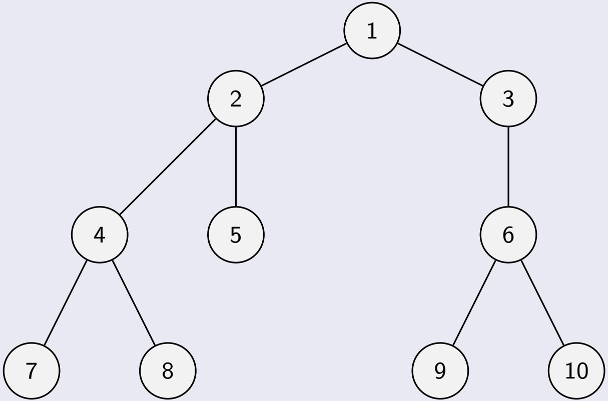
La vérification de la connexité d'un graphe est un des premiers problèmes de la théorie des graphes : vérification de la connexité d'un réseau électrique, d'un réseau téléphonique...

Graphes : définitions

Exemple de graphe connexe

Graphes : définitions

Exemple de graphe connexe



Forte connexité

Graphes : définitions

Forte connexité

Un graphe est fortement connexe si, étant donné deux sommets quelconques i et j dans cet ordre, il existe un chemin d'extrémité initiale i et d'extrémité finale j.

Graphes : définitions

Forte connexité

Un graphe est fortement connexe si, étant donné deux sommets quelconques i et j dans cet ordre, il existe un chemin d'extrémité initiale i et d'extrémité finale j.

Contre-exemple

Graphes : définitions

Forte connexité

Un graphe est fortement connexe si, étant donné deux sommets quelconques i et j dans cet ordre, il existe un chemin d'extrémité initiale i et d'extrémité finale j.

Contre-exemple

Le graphe des accès possibles à Saint-Dié à partir des grandes villes périphériques n'est pas fortement connexe :

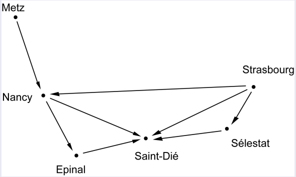
Graphes : définitions

Forte connexité

Un graphe est fortement connexe si, étant donné deux sommets quelconques i et j dans cet ordre, il existe un chemin d'extrémité initiale i et d'extrémité finale j.

Contre-exemple

Le graphe des accès possibles à Saint-Dié à partir des grandes villes périphériques n'est pas fortement connexe :
il n'existe pas de chemin d'extrémité initiale Saint-Dié et d'extrémité finale Strasbourg.



Graphes : définitions

Forte connexité : fermeture transitive

Graphes : définitions

Forte connexité : fermeture transitive

La fermeture transitive de chaque sommet i d'un graphe d'ordre n : $\widehat{\Gamma}_i = \{i\} \cup \Gamma_i \cup \Gamma_i^2 \cup \cdots \Gamma_i^{n-1}$, permet d'étudier la forte connenexité :

Graphes : définitions

Forte connexité : fermeture transitive

La fermeture transitive de chaque sommet i d'un graphe d'ordre n :
$\widehat{\Gamma}_i = \{i\} \cup \Gamma_i \cup \Gamma_i^2 \cup \cdots \Gamma_i^{n-1}$, permet d'étudier la forte connenexité :
$\mathcal{G}$ est fortement connexe si et seulement si $\forall i\ \widehat{\Gamma}_i = E$.

Graphes : définitions

Forte connexité : approche matricielle

Graphes : définitions

Forte connexité : approche matricielle

En notant $A = (a_{ij})$ et $A' = \left(a'_{ij}\right)$ deux matrices booléennes, on définit les opérations booléennes :

Graphes : définitions

Forte connexité : approche matricielle

En notant $A = (a_{ij})$ et $A' = (a'_{ij})$ deux matrices booléennes, on définit les opérations booléennes :

$A \vee A' = (s_{ij})$ par $s_{ij} = a_{ij} \vee a'_{ij}$

Graphes : définitions

Forte connexité : approche matricielle

En notant $A = (a_{ij})$ et $A' = \left(a'_{ij}\right)$ deux matrices booléennes, on définit les opérations booléennes :

$A \vee A' = (s_{ij})$ par $s_{ij} = a_{ij} \vee a'_{ij}$

$A \wedge A' = (p_{ij})$ par $p_{ij} = \left(a_{i1} \wedge a'_{1j}\right) \vee \left(a_{i2} \wedge a'_{2j}\right) \vee \ldots \vee \left(a_{in} \wedge a'_{nj}\right)$

Graphes : définitions

Forte connexité : approche matricielle

En notant $A = (a_{ij})$ et $A' = \left(a'_{ij}\right)$ deux matrices booléennes, on définit les opérations booléennes :

$A \vee A' = (s_{ij})$ par $s_{ij} = a_{ij} \vee a'_{ij}$

$A \wedge A' = (p_{ij})$ par $p_{ij} = \left(a_{i1} \wedge a'_{1j}\right) \vee \left(a_{i2} \wedge a'_{2j}\right) \vee \ldots \vee \left(a_{in} \wedge a'_{nj}\right)$

Autres notations : $A \vee A' = A + A'$ et $A \wedge A' = A.A' = A \times A' = AA'$.

Graphes : définitions

Forte connexité : approche matricielle

Graphes : définitions

Forte connexité : approche matricielle

On note A la matrice d'adjacence d'un 1-graphe $\mathcal{G}$

Graphes : définitions

Forte connexité : approche matricielle

On note A la matrice d'adjacence d'un 1-graphe $\mathcal{G}$ et $A^k = \left(a_{ij}^{(k)}\right)$ la matrice obtenue par la multiplication booléenne définie.

Graphes : définitions

Forte connexité : approche matricielle

On note A la matrice d'adjacence d'un 1-graphe $\mathcal{G}$ et $A^k = \left(a_{ij}^{(k)}\right)$ la matrice obtenue par la multiplication booléenne définie.
Si $a_{ij}^{(k)} = 1$ alors il existe un chemin de longueur k joignant i et j.

Forte connexité : approche matricielle

On note A la matrice d'adjacence d'un 1-graphe $\mathcal{G}$ et $A^k = \left(a_{ij}^{(k)}\right)$ la matrice obtenue par la multiplication booléenne définie.
Si $a_{ij}^{(k)} = 1$ alors il existe un chemin de longueur k joignant i et j. Sinon, il n'y en a pas.

Graphes : définitions

Forte connexité : approche matricielle

On note A la matrice d'adjacence d'un 1-graphe $\mathcal{G}$ et $A^k = \left(a_{ij}^{(k)}\right)$ la matrice obtenue par la multiplication booléenne définie.
Si $a_{ij}^{(k)} = 1$ alors il existe un chemin de longueur k joignant i et j. Sinon, il n'y en a pas.
On en déduit

1. $A \vee A^2 \vee A^3 \ldots \vee A^{n-1}$ précise les sommets qui peuvent être atteints par un chemin de longueur quelconque.

Graphes : définitions

Forte connexité : approche matricielle

On note A la matrice d'adjacence d'un 1-graphe $\mathcal{G}$ et $A^k = \left(a_{ij}^{(k)}\right)$ la matrice obtenue par la multiplication booléenne définie.
Si $a_{ij}^{(k)} = 1$ alors il existe un chemin de longueur k joignant i et j. Sinon, il n'y en a pas.
On en déduit

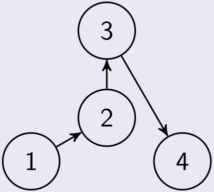
1. $A \vee A^2 \vee A^3 \ldots \vee A^{n-1}$ précise les sommets qui peuvent être atteints par un chemin de longueur quelconque.
2. $\mathcal{G}$ est fortement connexe si et seulement si $I_n \vee A \vee A^2 \vee A^3 \ldots \vee A^{n-1} = (1)$

Graphes : définitions

Exemple 1

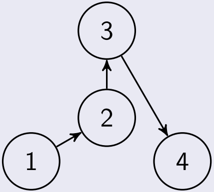
Graphes : définitions

Exemple 1



Graphes : définitions

Exemple 1



$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Graphes : définitions

Exemple 1

Graphes : définitions

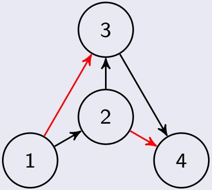
Exemple 1

$$A^2 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Graphes : définitions

Exemple 1

$$A^2 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$



Graphes : définitions

Exemple 1

Graphes : définitions

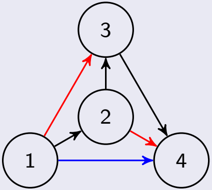
Exemple 1

$$A^3 = A^2 A = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Graphes : définitions

Exemple 1

$$A^3 = A^2 A = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$



Graphes : définitions

Exemple 1

Graphes : définitions

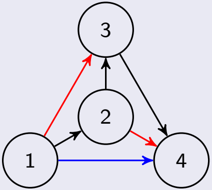
Exemple 1

$$A^4 = A^3 A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Graphes : définitions

Exemple 1

$$A^4 = A^3 A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$



Graphes : définitions

Exemple 1

Graphes : définitions

Exemple 1

$$A \vee A^2 \vee A^3 = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Graphes : définitions

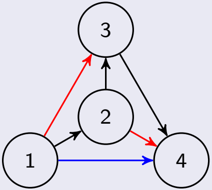
Exemple 1

$$A \vee A^2 \vee A^3 = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} ; I_4 \vee A \vee A^2 \vee A^3 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Graphes : définitions

Exemple 1

$$A \vee A^2 \vee A^3 = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} ;\ I_4 \vee A \vee A^2 \vee A^3 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



Graphes : définitions

Exemple 1

Graphes : définitions

Exemple 1

$I_4 \vee A \vee A^2 \vee A^3 \neq (1)$

Graphes : définitions

Exemple 1

$I_4 \vee A \vee A^2 \vee A^3 \neq (1)$

Le graphe n'est pas fortement connexe.

Graphes : définitions

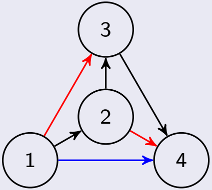
Exemple 1

$I_4 \vee A \vee A^2 \vee A^3 \neq (1)$
Le graphe n'est pas fortement connexe.
$A^4 = 0$: le graphe n'a pas de circuit.

Graphes : définitions

Exemple 1

$I_4 \vee A \vee A^2 \vee A^3 \neq (1)$
Le graphe n'est pas fortement connexe.
$A^4 = 0$: le graphe n'a pas de circuit.



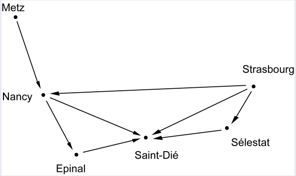
Graphes : définitions

Exemple 2

Graphes : définitions

Exemple 2

$$A = \begin{array}{c|c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$



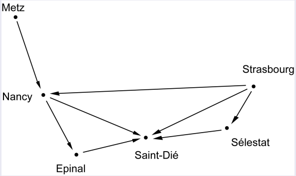
Graphes : définitions

Exemple 2

Graphes : définitions

Exemple 2

$$A^2 = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$



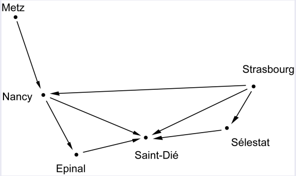
Graphes : définitions

Exemple 2

Graphes : définitions

Exemple 2

$$A^3 = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$



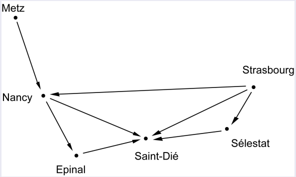
Graphes : définitions

Exemple 2

Graphes : définitions

Exemple 2

$$A^4 = \begin{array}{c c} & \begin{array}{cccccc} \text{E} & \text{M} & \text{N} & \text{SD} & \text{Se} & \text{Stg} \end{array} \\ \begin{array}{c} \text{E} \\ \text{M} \\ \text{N} \\ \text{SD} \\ \text{Se} \\ \text{Stg} \end{array} & \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$



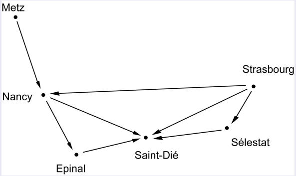
Graphes : définitions

Exemple 2

Graphes : définitions

Exemple 2

$$A \vee A^2 \vee A^3 \vee A^4 \vee A^5 = \begin{array}{c} \\ \mathrm{E} \\ \mathrm{M} \\ \mathrm{N} \\ \mathrm{SD} \\ \mathrm{Se} \\ \mathrm{Stg} \end{array} \begin{array}{c} \begin{array}{cccccc} \mathrm{E} & \mathrm{M} & \mathrm{N} & \mathrm{SD} & \mathrm{Se} & \mathrm{Stg} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$



Graphes : définitions

Exemple 2

Graphes : définitions

Exemple 2

$$I_6 \vee A \vee A^2 \vee A^3 \vee A^4 \vee A^5 =$$

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array} \neq (1)$$

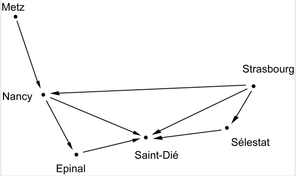
Graphes : définitions

Exemple 2

$$I_6 \vee A \vee A^2 \vee A^3 \vee A^4 \vee A^5 =$$

$$\begin{array}{c c} & \begin{array}{cccccc} \text{E} & \text{M} & \text{N} & \text{SD} & \text{Se} & \text{Stg} \end{array} \\ \begin{array}{c} \text{E} \\ \text{M} \\ \text{N} \\ \text{SD} \\ \text{Se} \\ \text{Stg} \end{array} & \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array} \neq (1)$$

Graphe non fortement connexe.



Graphes : définitions

Exemple 3

Graphes : définitions

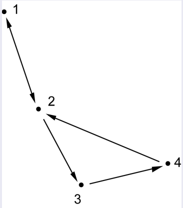
Exemple 3

Le graphe suivant est fortement connexe :

Graphes : définitions

Exemple 3

Le graphe suivant est fortement connexe :



Graphes : définitions

Arbre

Graphes : définitions

Arbre

Un arbre est un graphe connexe sans cycles.

Graphes : définitions

Arbre

Un arbre est un graphe connexe sans cycles.
Un graphe sans cycles qui n'est pas connexe est appelé une forêt (chaque composante connexe est un arbre).

Graphes : définitions

Arbre

Un arbre est un graphe connexe sans cycles.
Un graphe sans cycles qui n'est pas connexe est appelé une forêt (chaque composante connexe est un arbre).
Remarque : un arbre est un graphe simple.

Graphes : définitions

Arbre

Un arbre est un graphe connexe sans cycles.
Un graphe sans cycles qui n'est pas connexe est appelé une forêt (chaque composante connexe est un arbre).
Remarque : un arbre est un graphe simple.
Exemples

1. l'arbre généalogique d'une famille (les sommets sont les membres de la famille et les arêtes les liens de parenté directe),

Graphes : définitions

Arbre

Un arbre est un graphe connexe sans cycles.
Un graphe sans cycles qui n'est pas connexe est appelé une forêt (chaque composante connexe est un arbre).
Remarque : un arbre est un graphe simple.
Exemples

1. l'arbre généalogique d'une famille (les sommets sont les membres de la famille et les arêtes les liens de parenté directe),
2. une rivière et ses affluents (les sommets sont les sources, les confluents et l'embouchure, en supposant qu'il n'y a pas d'île).

Graphes : définitions

Exemple

Graphes : définitions

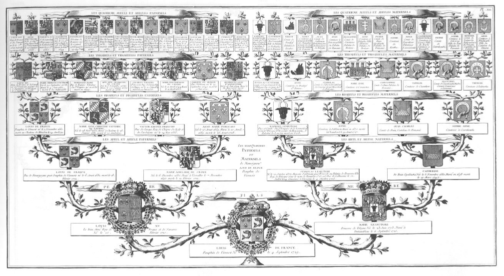
Exemple

Arbre généalogique

Graphes : définitions

Exemple

Arbre généalogique

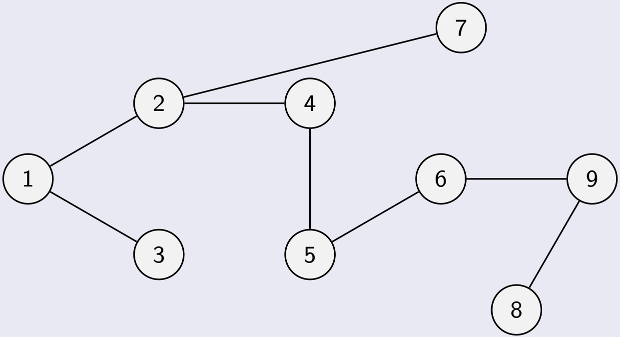


Graphes : définitions

Autre exemple

Graphes : définitions

Autre exemple



Graphes : définitions

Arbre : caractérisation

Graphes : définitions

Arbre : caractérisation

Un graphe $\mathcal{G}$ d'ordre n est un arbre s'il vérifie les conditions équivalentes suivantes :

Graphes : définitions

Arbre : caractérisation

Un graphe $\mathcal{G}$ d'ordre n est un arbre s'il vérifie les conditions équivalentes suivantes :

1. $\mathcal{G}$ est connexe et sans cycle.

Graphes : définitions

Arbre : caractérisation

Un graphe $\mathcal{G}$ d'ordre n est un arbre s'il vérifie les conditions équivalentes suivantes :

1. $\mathcal{G}$ est connexe et sans cycle.
2. Pour toute paire de sommets x et y de $\mathcal{G}$, il existe dans $\mathcal{G}$ une chaîne et une seule d'extrémités x et y.

Graphes : définitions

Arbre : caractérisation

Un graphe $\mathcal{G}$ d'ordre n est un arbre s'il vérifie les conditions équivalentes suivantes :

1. $\mathcal{G}$ est connexe et sans cycle.
2. Pour toute paire de sommets x et y de $\mathcal{G}$, il existe dans $\mathcal{G}$ une chaîne et une seule d'extrémités x et y.
3. $\mathcal{G}$ est connexe minimum au sens des arêtes pour cette propriété, c'est-à-dire qu'il n'est plus connexe si on lui supprime l'une quelconque de ses arêtes.

Graphes : définitions

Arbre : caractérisation

Un graphe $\mathcal{G}$ d'ordre n est un arbre s'il vérifie les conditions équivalentes suivantes :

1. $\mathcal{G}$ est connexe et sans cycle.
2. Pour toute paire de sommets x et y de $\mathcal{G}$, il existe dans $\mathcal{G}$ une chaîne et une seule d'extrémités x et y.
3. $\mathcal{G}$ est connexe minimum au sens des arêtes pour cette propriété, c'est-à-dire qu'il n'est plus connexe si on lui supprime l'une quelconque de ses arêtes.
4. $\mathcal{G}$ est sans cycle et maximum au sens des arêtes pour cette propriété, c'est-à-dire qu'on crée un cycle en ajoutant une arête rendant adjacents deux quelconques de ses sommets qui ne le sont pas.

Graphes : définitions

Arbre : caractérisation

Un graphe $\mathcal{G}$ d'ordre n est un arbre s'il vérifie les conditions équivalentes suivantes :

1. $\mathcal{G}$ est connexe et sans cycle.
2. Pour toute paire de sommets x et y de $\mathcal{G}$, il existe dans $\mathcal{G}$ une chaîne et une seule d'extrémités x et y.
3. $\mathcal{G}$ est connexe minimum au sens des arêtes pour cette propriété, c'est-à-dire qu'il n'est plus connexe si on lui supprime l'une quelconque de ses arêtes.
4. $\mathcal{G}$ est sans cycle et maximum au sens des arêtes pour cette propriété, c'est-à-dire qu'on crée un cycle en ajoutant une arête rendant adjacents deux quelconques de ses sommets qui ne le sont pas.
5. $\mathcal{G}$ est connexe et possède $n-1$ arêtes.

Graphes : définitions

Arbre : caractérisation

Un graphe $\mathcal{G}$ d'ordre n est un arbre s'il vérifie les conditions équivalentes suivantes :

1. $\mathcal{G}$ est connexe et sans cycle.
2. Pour toute paire de sommets x et y de $\mathcal{G}$, il existe dans $\mathcal{G}$ une chaîne et une seule d'extrémités x et y.
3. $\mathcal{G}$ est connexe minimum au sens des arêtes pour cette propriété, c'est-à-dire qu'il n'est plus connexe si on lui supprime l'une quelconque de ses arêtes.
4. $\mathcal{G}$ est sans cycle et maximum au sens des arêtes pour cette propriété, c'est-à-dire qu'on crée un cycle en ajoutant une arête rendant adjacents deux quelconques de ses sommets qui ne le sont pas.
5. $\mathcal{G}$ est connexe et possède $n-1$ arêtes.
6. $\mathcal{G}$ est sans cycles et possède $n-1$ arêtes.

Graphes : définitions

Algorithme de détermination des composantes connexes

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Marquer a en bleu

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Marquer a en bleu

1. **tant que** il reste des sommets marqués en bleu faire
   choisir un sommet y marqué en bleu

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Marquer a en bleu

1. **tant que** il reste des sommets marqués en bleu faire
   choisir un sommet y marqué en bleu
   si tous les voisins du sommet y sont déjà marqués

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Marquer a en bleu

1. **tant que** il reste des sommets marqués en bleu faire
   choisir un sommet y marqué en bleu
   si tous les voisins du sommet y sont déjà marqués
   - **alors** marquer le sommet y en rouge

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Marquer a en bleu

1. **tant que** il reste des sommets marqués en bleu faire
   choisir un sommet y marqué en bleu
   si tous les voisins du sommet y sont déjà marqués
   - **alors** marquer le sommet y en rouge
   - **sinon**
     choisir un sommet z parmi les voisins non encore marqués du sommet y

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Marquer a en bleu

1. **tant que** il reste des sommets marqués en bleu faire
   choisir un sommet y marqué en bleu
   si tous les voisins du sommet y sont déjà marqués
   - **alors** marquer le sommet y en rouge
   - **sinon**
     choisir un sommet z parmi les voisins non encore marqués du sommet y
     marquer le sommet z en bleu

Graphes : définitions

Algorithme de détermination des composantes connexes

Donnée : Un graphe $\mathcal{G}$ et un sommet a de $\mathcal{G}$
Résultat : L'ensemble des sommets de la composante connexe de a dans $\mathcal{G}$

Marquer a en bleu

1. **tant que** il reste des sommets marqués en bleu faire
   choisir un sommet y marqué en bleu
   si tous les voisins du sommet y sont déjà marqués
   - **alors** marquer le sommet y en rouge
   - **sinon**
     choisir un sommet z parmi les voisins non encore marqués du sommet y
     marquer le sommet z en bleu
2. Retourner l'ensemble des sommets marqués

Graphes : définitions

Connexité : recherche en largeur ou en profondeur

Graphes : définitions

Connexité : recherche en largeur ou en profondeur

Stratégies possibles pour le choix de y :

Graphes : définitions

Connexité : recherche en largeur ou en profondeur

Stratégies possibles pour le choix de y :

- *premier marqué en bleu, premier choisi* : *parcours en largeur* de la composante connexe (algorithme breadth first search),

Graphes : définitions

Connexité : recherche en largeur ou en profondeur

Stratégies possibles pour le choix de y :

- *premier marqué en bleu, premier choisi* : *parcours en largeur* de la composante connexe (algorithme breadth first search),
- *dernier marqué en bleu, premier choisi* : *parcours en profondeur* de la composante connexe (algorithme depth first search).

Graphes : définitions

Connexité : recherche en largeur ou en profondeur

Stratégies possibles pour le choix de y :

- *premier marqué en bleu, premier choisi* : *parcours en largeur* de la composante connexe (algorithme breadth first search),
- *dernier marqué en bleu, premier choisi* : *parcours en profondeur* de la composante connexe (algorithme depth first search).

On peut repérer le numéro de l'itération où un sommet donné est coloré en bleu, puis celui de l'itération où est coloré en rouge.

Connexité : recherche en largeur ou en profondeur

Stratégies possibles pour le choix de y :

- *premier marqué en bleu, premier choisi* : *parcours en largeur* de la composante connexe (algorithme breadth first search),
- *dernier marqué en bleu, premier choisi* : *parcours en profondeur* de la composante connexe (algorithme depth first search).

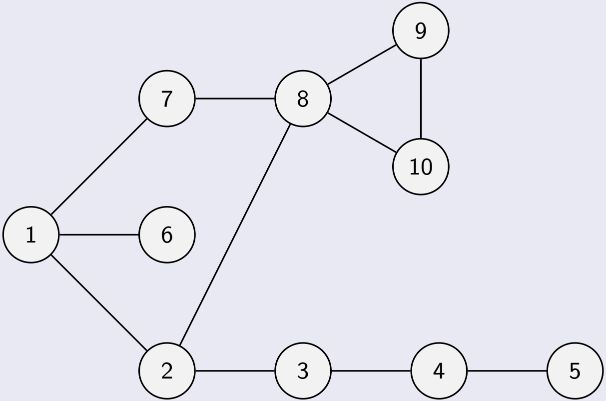
On peut repérer le numéro de l'itération où un sommet donné est coloré en bleu, puis celui de l'itération où est coloré en rouge.
Le choix de z est libre : par exemple l'ordre croissant des numéros de sommet.

Graphes : définitions

Exemple de parcours en largeur

Graphes : définitions

Exemple de parcours en largeur

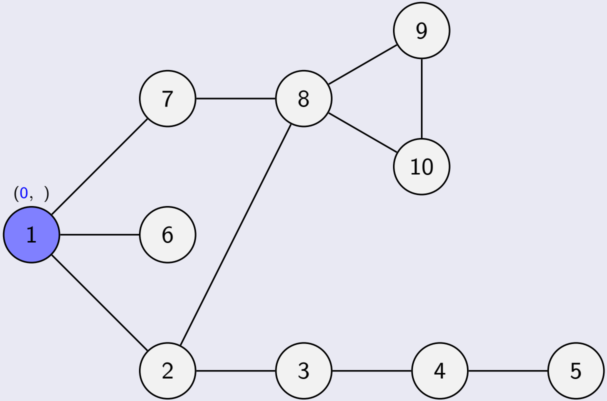


Graphes : définitions

Exemple de parcours en largeur : itération 0

Graphes : définitions

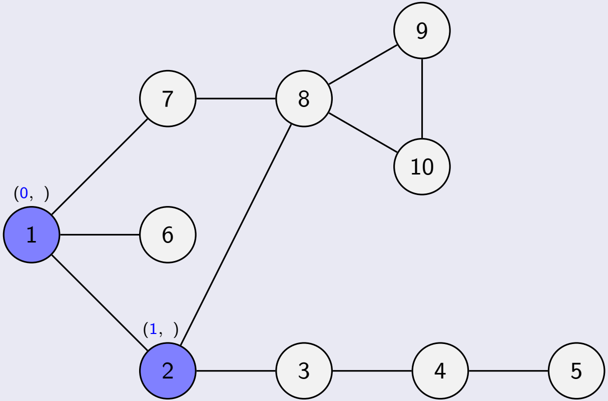
Exemple de parcours en largeur : itération 0



Exemple de parcours en largeur : itération 1

Graphes : définitions

Exemple de parcours en largeur : itération 1

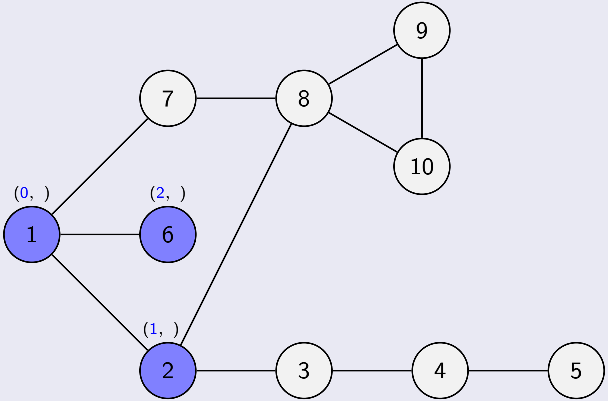


Graphes : définitions

Exemple de parcours en largeur : itération 2

Graphes : définitions

Exemple de parcours en largeur : itération 2

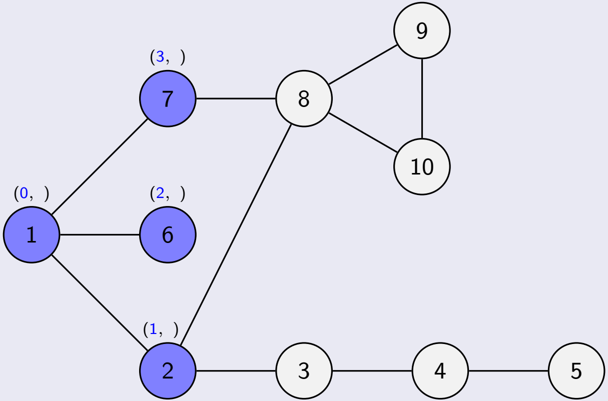


Graphes : définitions

Exemple de parcours en largeur : itération 3

Graphes : définitions

Exemple de parcours en largeur : itération 3

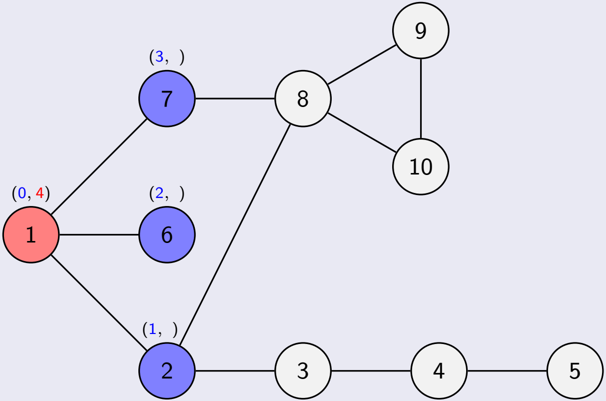


Graphes : définitions

Exemple de parcours en largeur : itération 4

Graphes : définitions

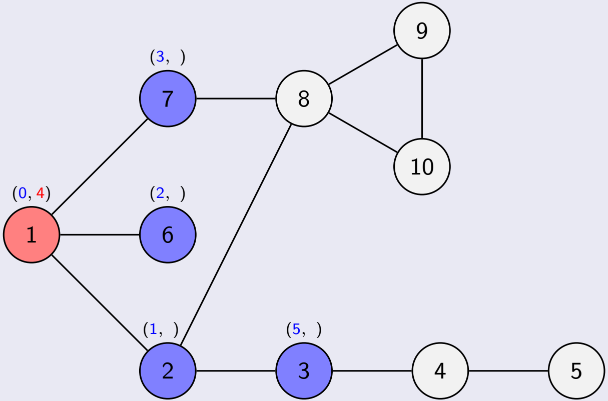
Exemple de parcours en largeur : itération 4



Exemple de parcours en largeur : itération 5

Graphes : définitions

Exemple de parcours en largeur : itération 5

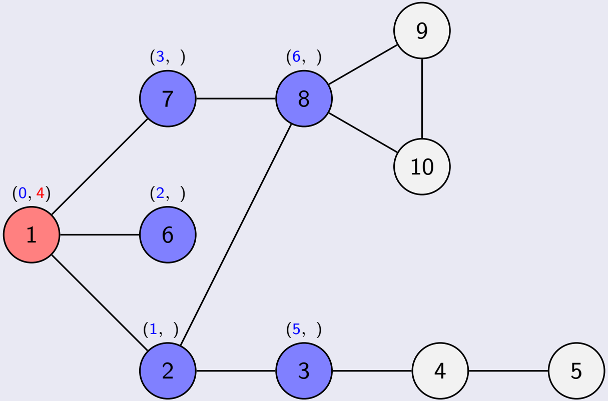


Graphes : définitions

Exemple de parcours en largeur : itération 6

Graphes : définitions

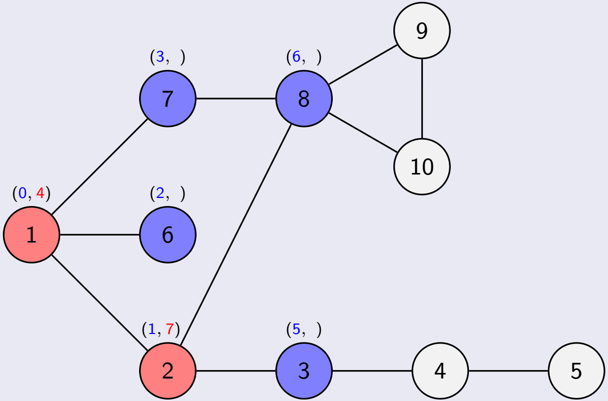
Exemple de parcours en largeur : itération 6



Exemple de parcours en largeur : itération 7

Graphes : définitions

Exemple de parcours en largeur : itération 7

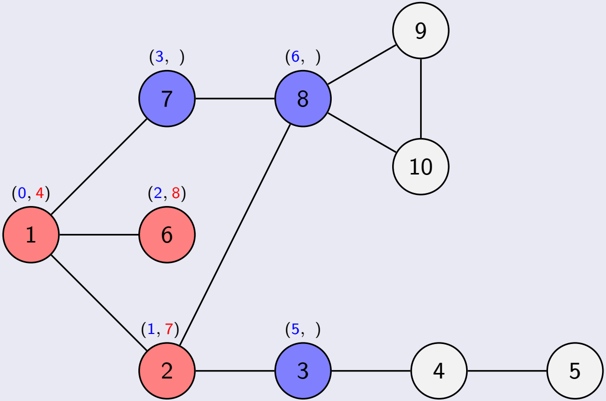


Graphes : définitions

Exemple de parcours en largeur : itération 8

Graphes : définitions

Exemple de parcours en largeur : itération 8

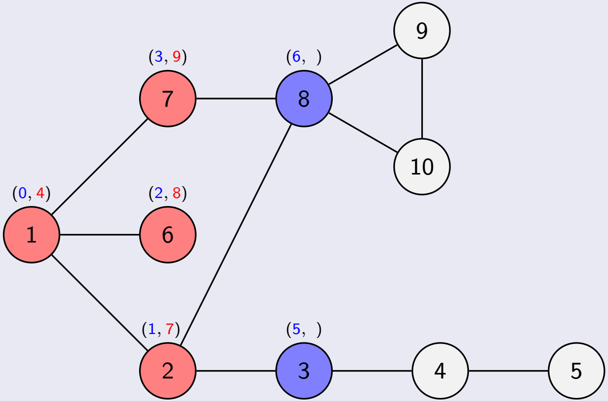


Graphes : définitions

Exemple de parcours en largeur : itération 9

Graphes : définitions

Exemple de parcours en largeur : itération 9

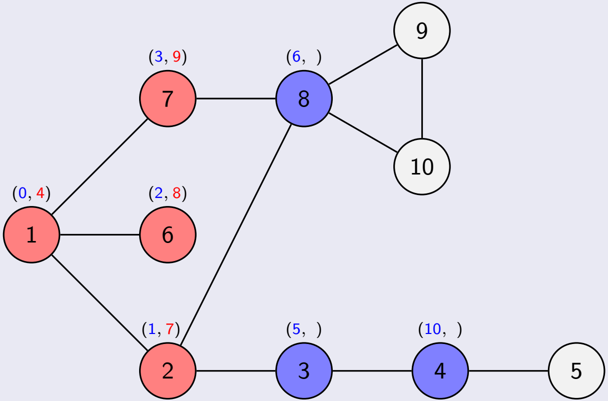


Graphes : définitions

Exemple de parcours en largeur : itération 10

Graphes : définitions

Exemple de parcours en largeur : itération 10

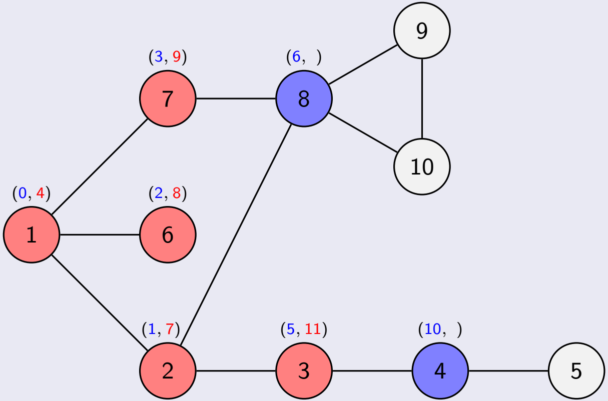


Graphes : définitions

Exemple de parcours en largeur : itération 11

Graphes : définitions

Exemple de parcours en largeur : itération 11

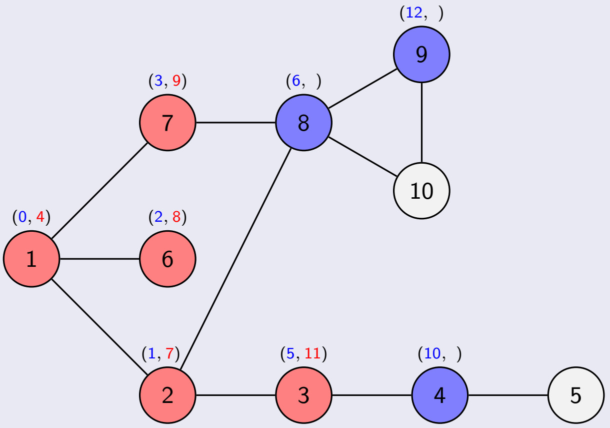


Graphes : définitions

Exemple de parcours en largeur : itération 12

Graphes : définitions

Exemple de parcours en largeur : itération 12



Graphes : définitions

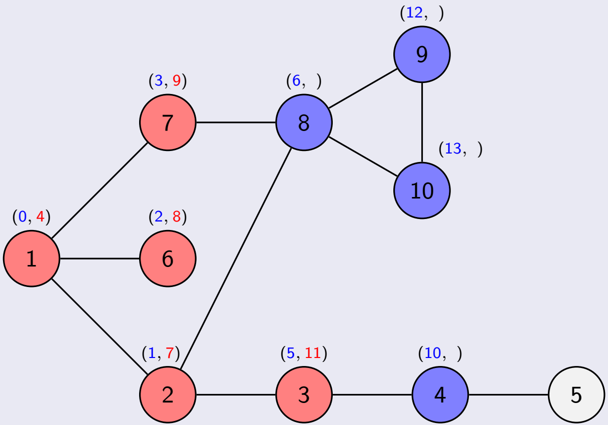
Exemple de parcours en largeur : itération 13

Graphes : définitions

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Exemple de parcours en largeur : itération 13

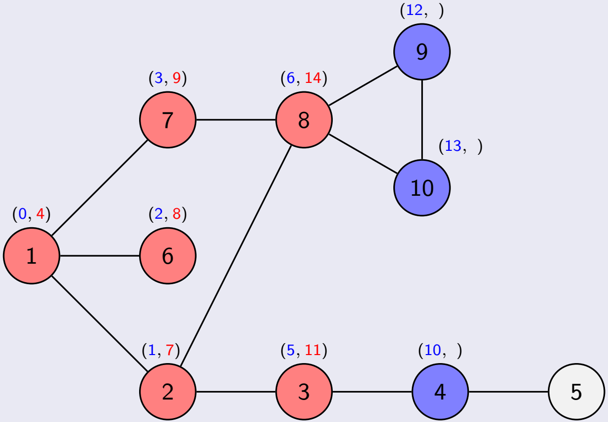


Graphes : définitions

Exemple de parcours en largeur : itération 14

Graphes : définitions

Exemple de parcours en largeur : itération 14



Graphes : définitions

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

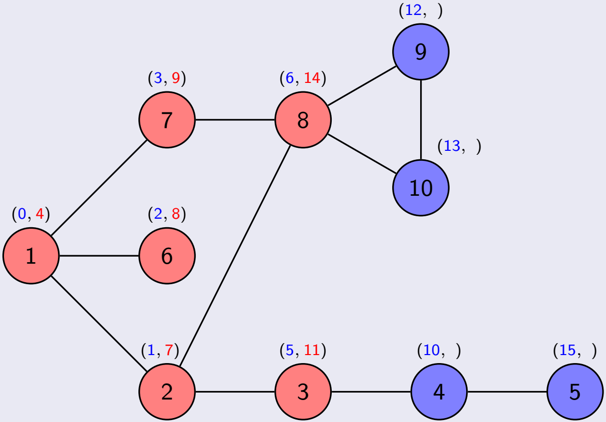
Exemple de parcours en largeur : itération 15

Graphes : définitions

M2201
Graphes et
langages

Département
Informatique
IUT de
Saint-Dié

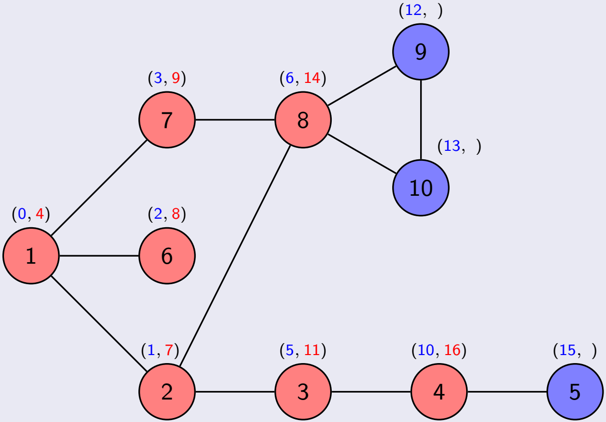
Exemple de parcours en largeur : itération 15



Exemple de parcours en largeur : itération 16

Graphes : définitions

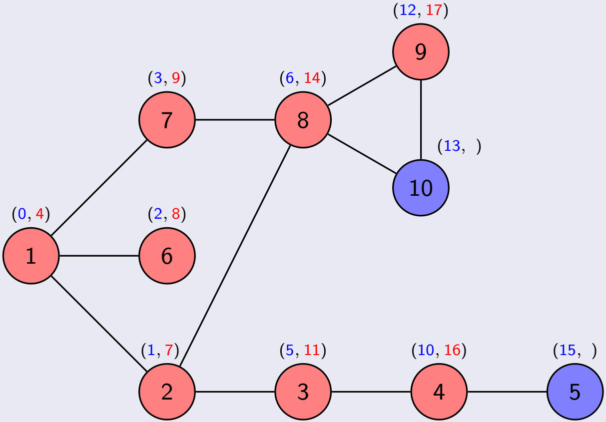
Exemple de parcours en largeur : itération 16



Exemple de parcours en largeur : itération 17

Graphes : définitions

Exemple de parcours en largeur : itération 17

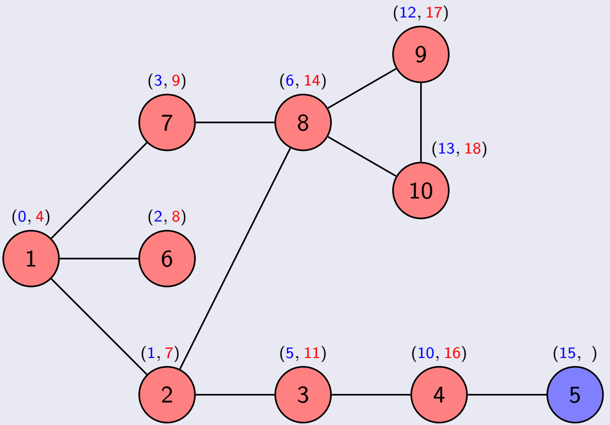


Graphes : définitions

Exemple de parcours en largeur : itération 18

Graphes : définitions

Exemple de parcours en largeur : itération 18



Graphes : définitions

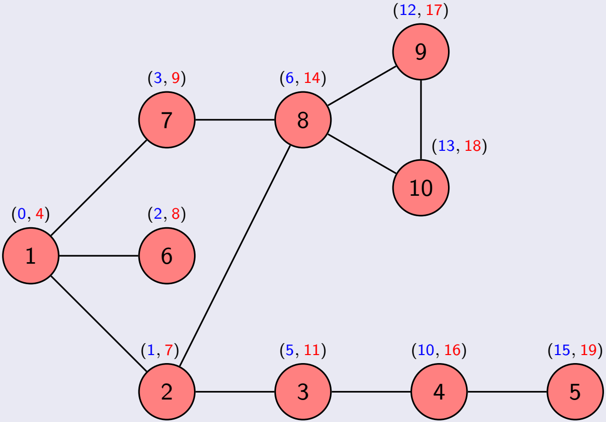
M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Exemple de parcours en largeur : itération 19

Graphes : définitions

Exemple de parcours en largeur : itération 19

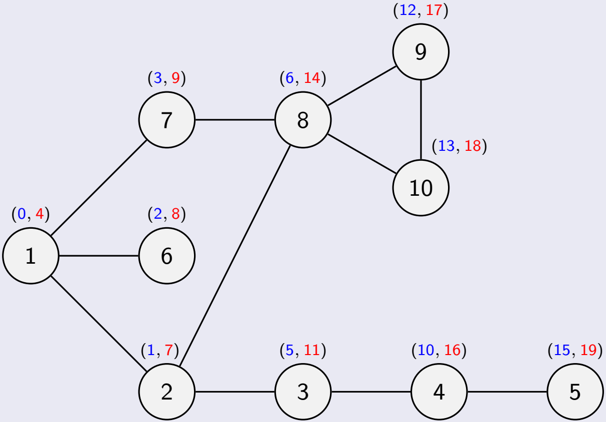


Graphes : définitions

Exemple de parcours en largeur : résultat

Graphes : définitions

Exemple de parcours en largeur : résultat



Graphes : définitions

Exemple de parcours en largeur : résultat

Graphes : définitions

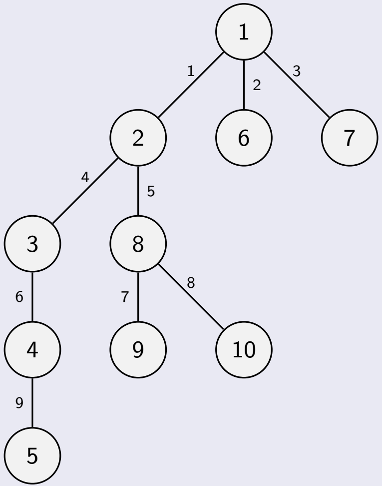
Exemple de parcours en largeur : résultat

sommet	coloration en bleu	coloration en rouge
1	0	4
2	1	7
3	5	11
4	10	16
5	15	19
6	2	8
7	3	9
8	6	14
9	12	17
10	13	18

Exemple de parcours en largeur : arbre associé au parcours

Graphes : définitions

Exemple de parcours en largeur : arbre associé au parcours

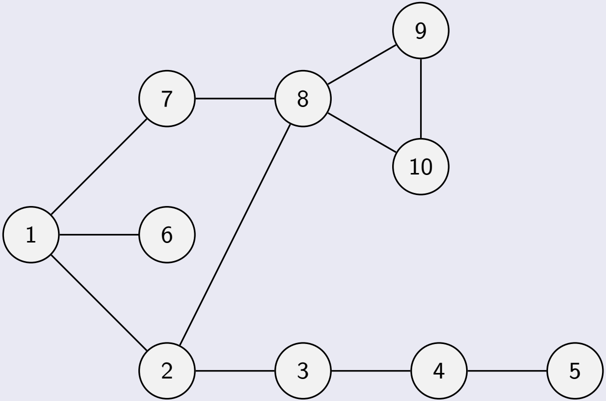


Graphes : définitions

Exemple de parcours en profondeur

Graphes : définitions

Exemple de parcours en profondeur

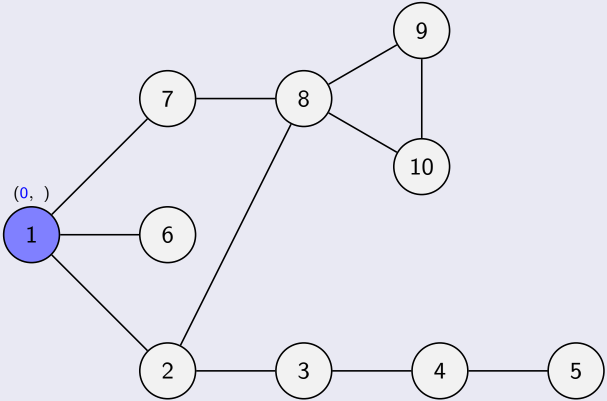


Graphes : définitions

Exemple de parcours en profondeur : itération 0

Graphes : définitions

Exemple de parcours en profondeur : itération 0

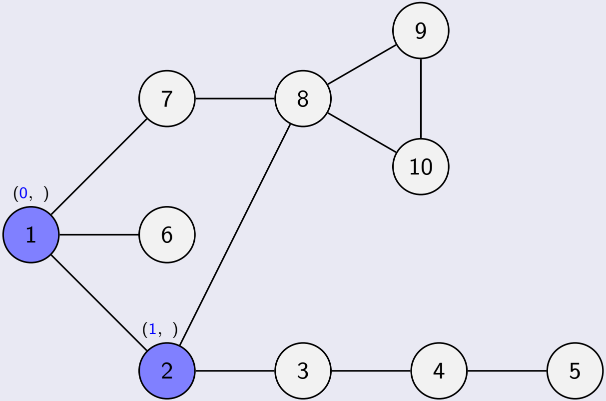


Graphes : définitions

Exemple de parcours en profondeur : itération 1

Graphes : définitions

Exemple de parcours en profondeur : itération 1

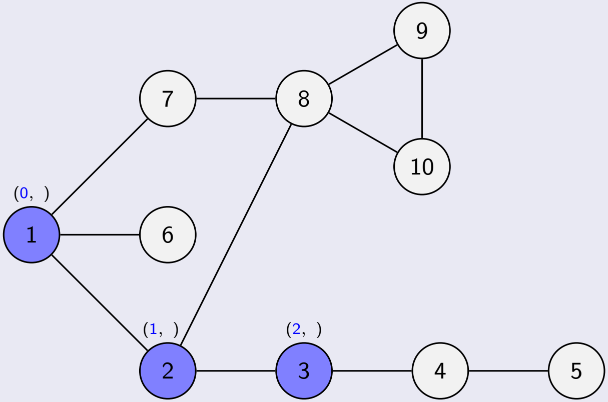


Graphes : définitions

Exemple de parcours en profondeur : itération 2

Graphes : définitions

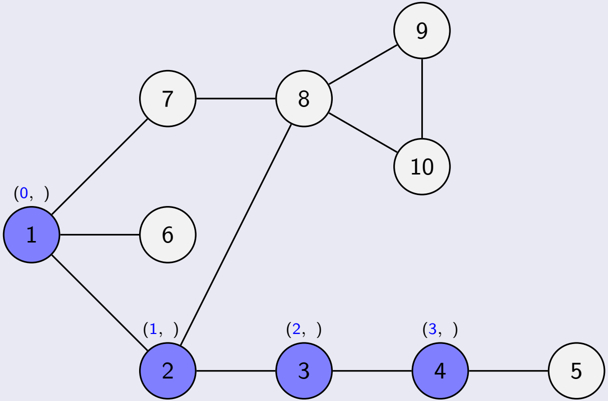
Exemple de parcours en profondeur : itération 2



Exemple de parcours en profondeur : itération 3

Graphes : définitions

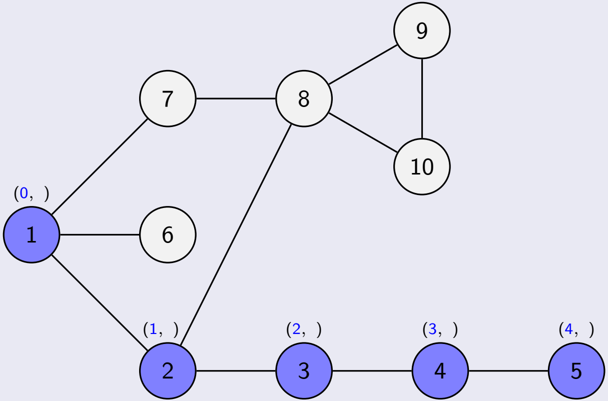
Exemple de parcours en profondeur : itération 3



Exemple de parcours en profondeur : itération 4

Graphes : définitions

Exemple de parcours en profondeur : itération 4

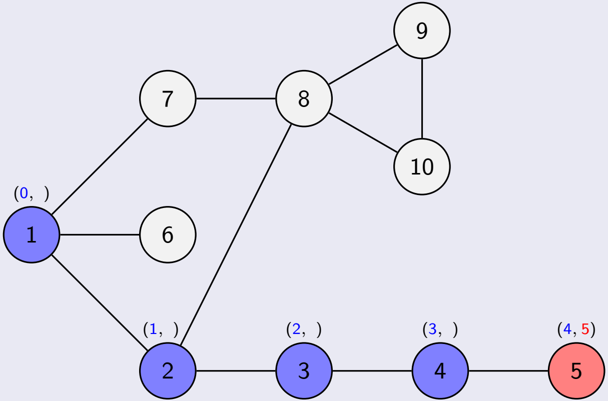


Graphes : définitions

Exemple de parcours en profondeur : itération 5

Graphes : définitions

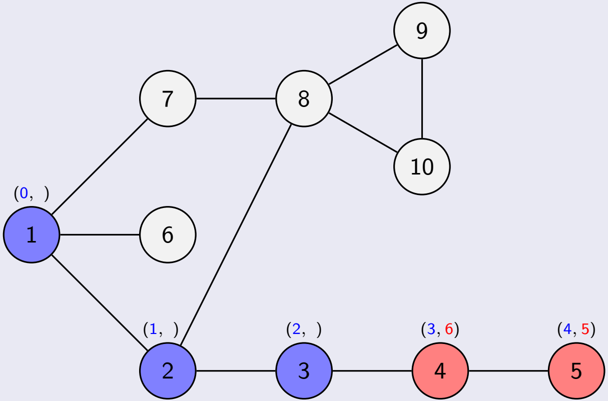
Exemple de parcours en profondeur : itération 5



Exemple de parcours en profondeur : itération 6

Graphes : définitions

Exemple de parcours en profondeur : itération 6

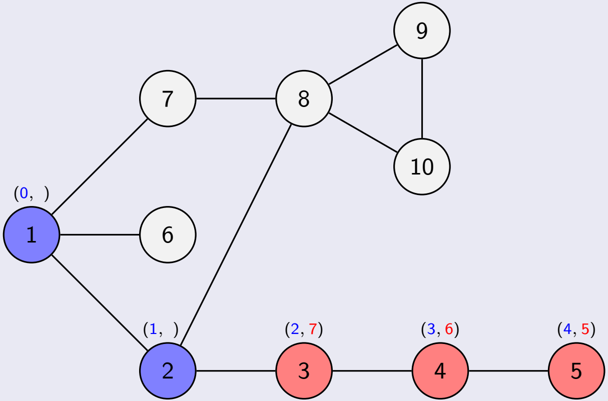


Graphes : définitions

Exemple de parcours en profondeur : itération 7

Graphes : définitions

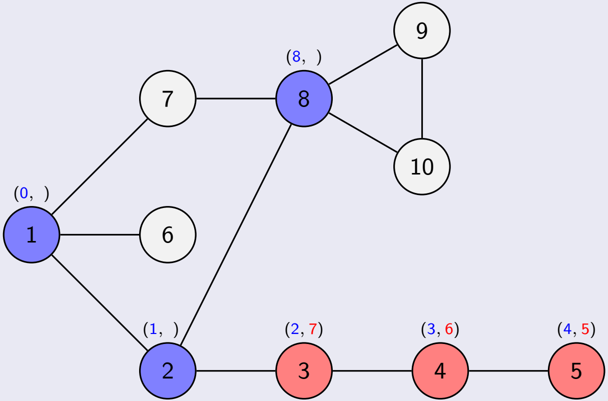
Exemple de parcours en profondeur : itération 7



Exemple de parcours en profondeur : itération 8

Graphes : définitions

Exemple de parcours en profondeur : itération 8

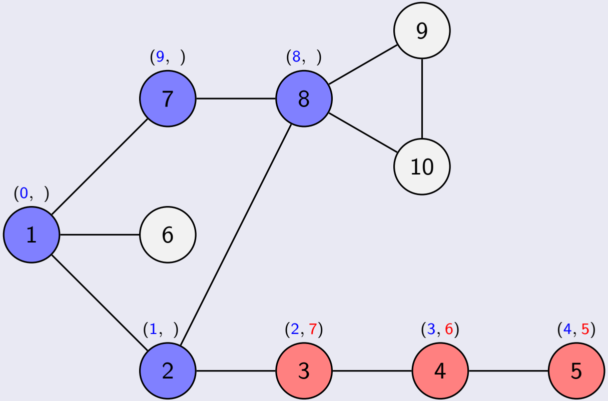


Graphes : définitions

Exemple de parcours en profondeur : itération 9

Graphes : définitions

Exemple de parcours en profondeur : itération 9

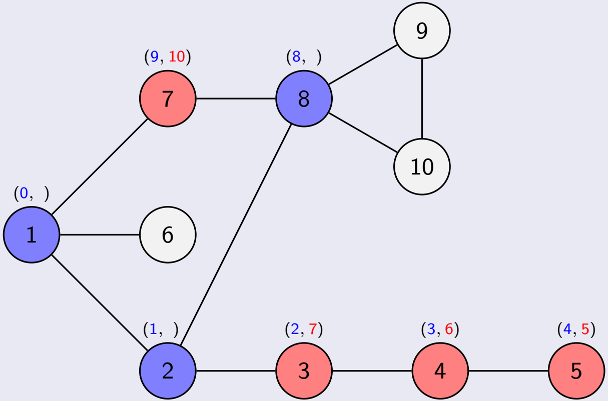


Graphes : définitions

Exemple de parcours en profondeur : itération 10

Graphes : définitions

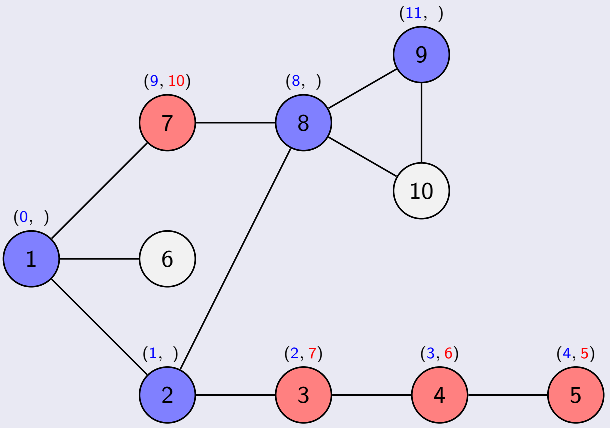
Exemple de parcours en profondeur : itération 10



Exemple de parcours en profondeur : itération 11

Graphes : définitions

Exemple de parcours en profondeur : itération 11

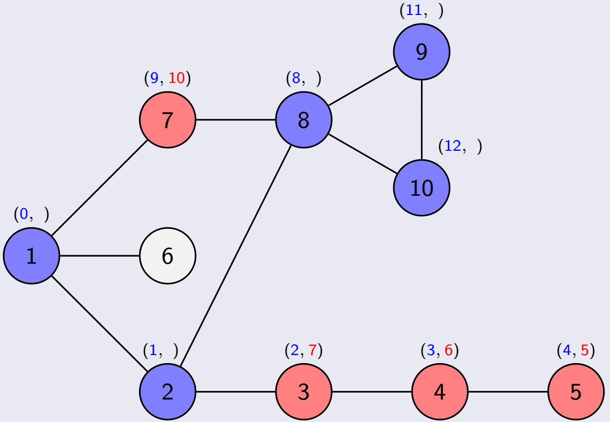


Graphes : définitions

Exemple de parcours en profondeur : itération 12

Graphes : définitions

Exemple de parcours en profondeur : itération 12

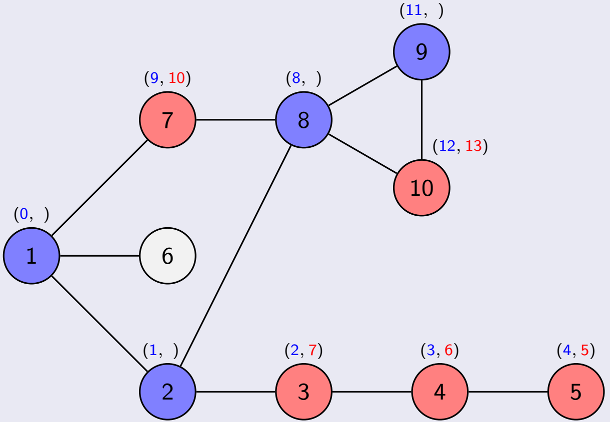


Graphes : définitions

Exemple de parcours en profondeur : itération 13

Graphes : définitions

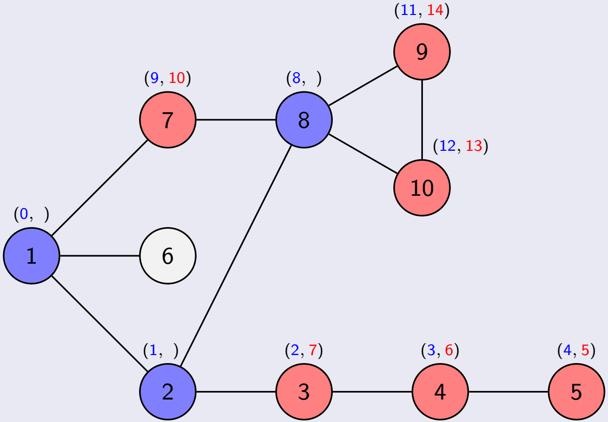
Exemple de parcours en profondeur : itération 13



Exemple de parcours en profondeur : itération 14

Graphes : définitions

Exemple de parcours en profondeur : itération 14

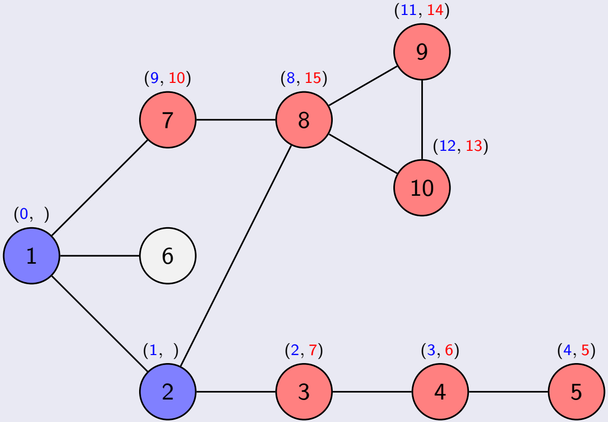


Graphes : définitions

Exemple de parcours en profondeur : itération 15

Graphes : définitions

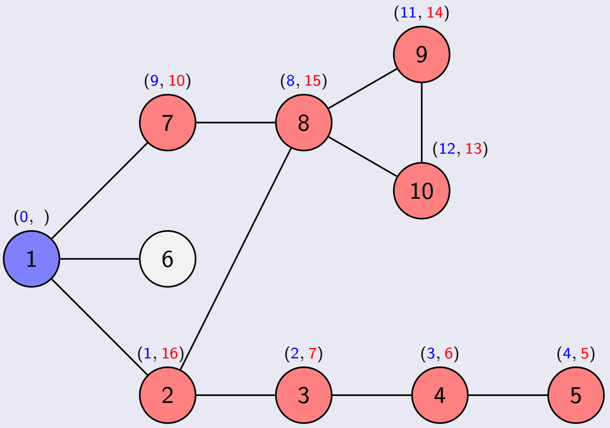
Exemple de parcours en profondeur : itération 15



Exemple de parcours en profondeur : itération 16

Graphes : définitions

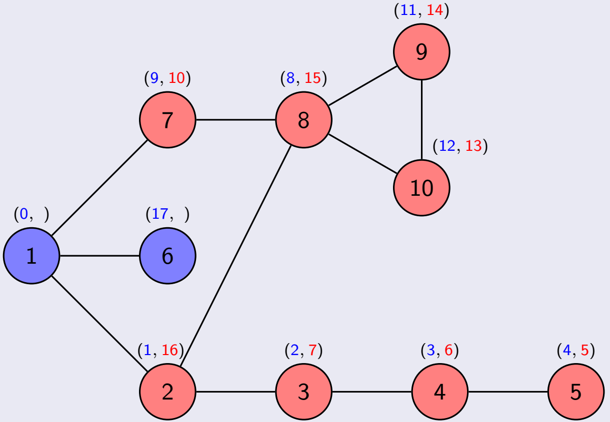
Exemple de parcours en profondeur : itération 16



Exemple de parcours en profondeur : itération 17

Graphes : définitions

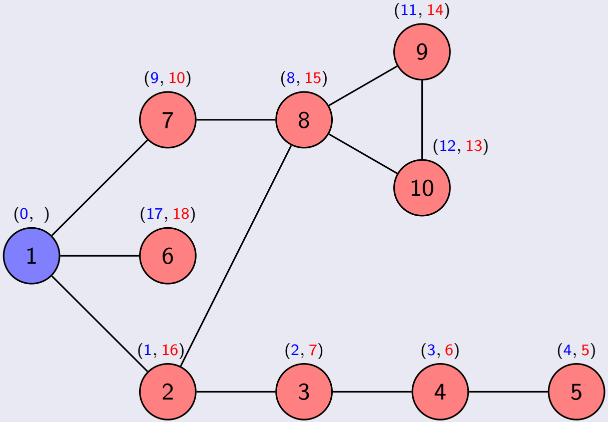
Exemple de parcours en profondeur : itération 17



Exemple de parcours en profondeur : itération 18

Graphes : définitions

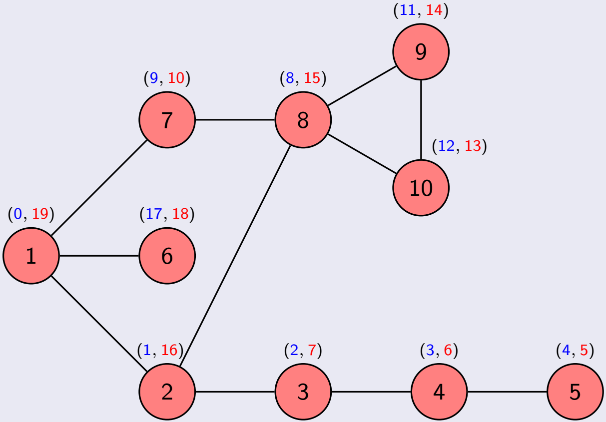
Exemple de parcours en profondeur : itération 18



Exemple de parcours en profondeur : itération 19

Graphes : définitions

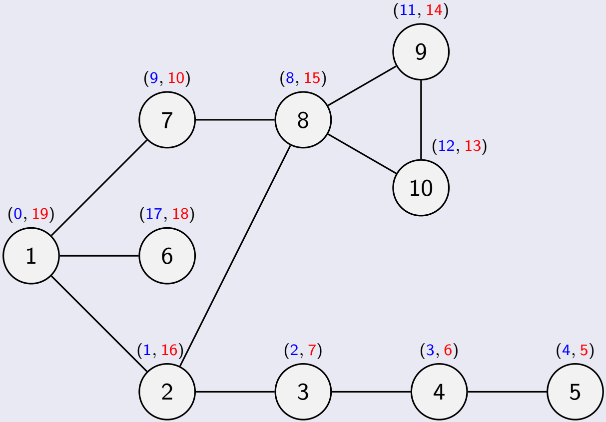
Exemple de parcours en profondeur : itération 19



Exemple de parcours en profondeur : résultat

Graphes : définitions

Exemple de parcours en profondeur : résultat



Graphes : définitions

Exemple de parcours en profondeur : résultat

Exemple de parcours en profondeur : résultat

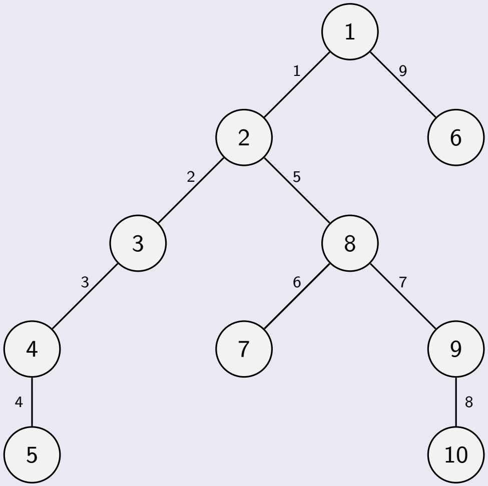
sommet	coloration en bleu	coloration en rouge
1	0	19
2	1	16
3	2	7
4	3	6
5	4	5
6	17	18
7	9	10
8	8	15
9	11	14
10	12	13

Graphes : définitions

Exemple de parcours en profondeur : arbre associé au parcours

Graphes : définitions

Exemple de parcours en profondeur : arbre associé au parcours



Graphes : définitions

Chaîne eulérienne

Graphes : définitions

Chaîne eulérienne

Une chaîne eulérienne d'un graphe $\mathcal{G}$ non orienté est une chaîne de $\mathcal{G}$ qui contient une fois et une seule chaque arête de $\mathcal{G}$.

Graphes : définitions

Chaîne eulérienne

Une chaîne eulérienne d'un graphe $\mathcal{G}$ non orienté est une chaîne de $\mathcal{G}$ qui contient une fois et une seule chaque arête de $\mathcal{G}$.
Un cycle eulérien de $\mathcal{G}$ est une chaîne eulérienne de $\mathcal{G}$ qui est un cycle, c'est-à-dire une chaîne eulérienne dont les extrémités sont confondues.

Graphes : définitions

Chaîne eulérienne

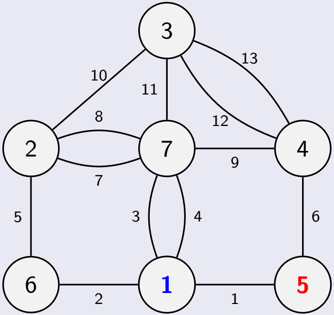
Une chaîne eulérienne d'un graphe $\mathcal{G}$ non orienté est une chaîne de $\mathcal{G}$ qui contient une fois et une seule chaque arête de $\mathcal{G}$.
Un cycle eulérien de $\mathcal{G}$ est une chaîne eulérienne de $\mathcal{G}$ qui est un cycle, c'est-à-dire une chaîne eulérienne dont les extrémités sont confondues.
Leonhard Euler (1707 à Bâle - 1783 à Saint-Pétersbourg) est un mathématicien suisse.

Graphes : définitions

Cycle eulérien

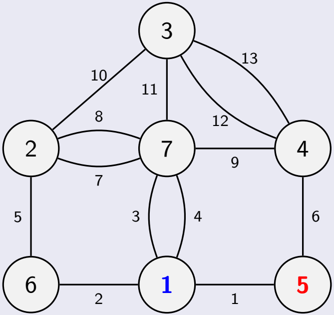
Graphes : définitions

Cycle eulérien



Graphes : définitions

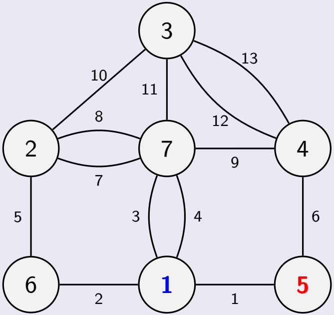
Cycle eulérien



Cycles eulériens à partir du sommet

Graphes : définitions

Cycle eulérien

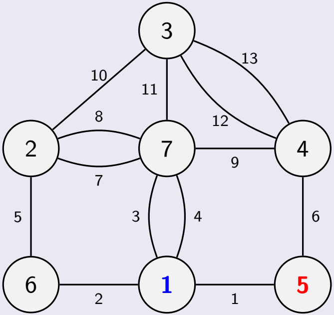


Cycles eulériens à partir du sommet

1. 1 : arêtes 4 - 11 - 13 - 12 - 10 - 8 - 9 - 6 - 1 - 3 - 7 - 5 - 2

Graphes : définitions

Cycle eulérien



Cycles eulériens à partir du sommet

1. 1 : arêtes 4 - 11 - 13 - 12 - 10 - 8 - 9 - 6 - 1 - 3 - 7 - 5 - 2
2. 5 : arêtes 6 - 13 - 12 - 9 - 11 - 10 - 8 - 7 - 5 - 2 - 4 - 3 - 1

Graphes : définitions

Cycle eulérien

Graphes : définitions

Cycle eulérien

Théorème 1

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.
C_1 est en fait un cycle.

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.
C_1 est en fait un cycle. Sinon, son extrémité finale serait de degré impair.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.
C_1 est en fait un cycle. Sinon, son extrémité finale serait de degré impair.
Si ce cycle C_1 contient toutes les arêtes du graphe $\mathcal{G}$, C_1 est un cycle eulérien.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.
C_1 est en fait un cycle. Sinon, son extrémité finale serait de degré impair.
Si ce cycle C_1 contient toutes les arêtes du graphe $\mathcal{G}$, C_1 est un cycle eulérien.
Sinon, on prend le sous-graphe $\mathcal{G}_1$ de $\mathcal{G}$ obtenu en éliminant les arêtes de C_1 et ses sommets qui ne sont incidents à aucune des arêtes restantes.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.
C_1 est en fait un cycle. Sinon, son extrémité finale serait de degré impair.
Si ce cycle C_1 contient toutes les arêtes du graphe $\mathcal{G}$, C_1 est un cycle eulérien.
Sinon, on prend le sous-graphe $\mathcal{G}_1$ de $\mathcal{G}$ obtenu en éliminant les arêtes de C_1 et ses sommets qui ne sont incidents à aucune des arêtes restantes.
Comme $\mathcal{G}$ est connexe, $\mathcal{G}_1$ possède au moins un sommet commun avec le cycle C_1. Soit i_1 un tel sommet.
Les sommets de $\mathcal{G}_1$ sont encore de degré pair.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.
C_1 est en fait un cycle. Sinon, son extrémité finale serait de degré impair.
Si ce cycle C_1 contient toutes les arêtes du graphe $\mathcal{G}$, C_1 est un cycle eulérien.
Sinon, on prend le sous-graphe $\mathcal{G}_1$ de $\mathcal{G}$ obtenu en éliminant les arêtes de C_1 et ses sommets qui ne sont incidents à aucune des arêtes restantes.
Comme $\mathcal{G}$ est connexe, $\mathcal{G}_1$ possède au moins un sommet commun avec le cycle C_1. Soit i_1 un tel sommet.
Les sommets de $\mathcal{G}_1$ sont encore de degré pair.
On construit un nouveau cycle C_2 dans $\mathcal{G}_1$ à partir de i_1 qu'on intègre à C_1 au sommet i_1.

Graphes : définitions

Cycle eulérien

Théorème 1 : un graphe connexe $\mathcal{G}$ admet un cycle eulérien si et seulement si tous ses sommets sont de degré pair.

Soit G eulérien. Notons C un cycle eulérien et i un sommet de $\mathcal{G}$. C contient toutes les arêtes de $\mathcal{G}$, donc toutes les $d(i)$ arêtes ayant i comme extrémité.
Lors d'un parcours de C on arrive en i autant de fois qu'on en repart, chaque arête de $\mathcal{G}$ étant présente une et seule fois dans C, $d(i)$ est nécessairement pair.
Réciproquement, supposons que tous les sommets de $\mathcal{G}$ soient de degré pair.
On construit une chaîne simple C_1, aussi longue que possible, à partir d'un sommet i_0.
C_1 est en fait un cycle. Sinon, son extrémité finale serait de degré impair.
Si ce cycle C_1 contient toutes les arêtes du graphe $\mathcal{G}$, C_1 est un cycle eulérien.
Sinon, on prend le sous-graphe $\mathcal{G}_1$ de $\mathcal{G}$ obtenu en éliminant les arêtes de C_1 et ses sommets qui ne sont incidents à aucune des arêtes restantes.
Comme $\mathcal{G}$ est connexe, $\mathcal{G}_1$ possède au moins un sommet commun avec le cycle C_1. Soit i_1 un tel sommet.
Les sommets de $\mathcal{G}_1$ sont encore de degré pair.
On construit un nouveau cycle C_2 dans $\mathcal{G}_1$ à partir de i_1 qu'on intègre à C_1 au sommet i_1.
Si on a toutes les arêtes du graphe on a fini, sinon on recommence, ce qui a une fin puisque le nombre de sommets est fini.

Graphes : définitions

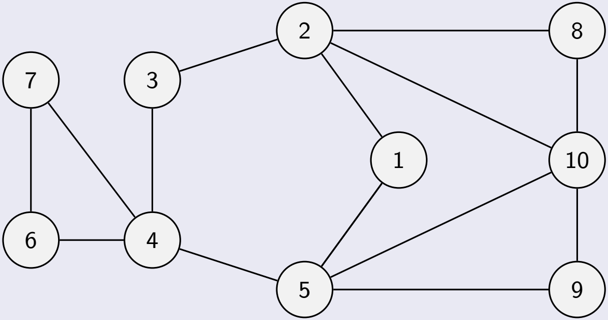
Cycle eulérien

Graphes : définitions

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Cycle eulérien



Graphes : définitions

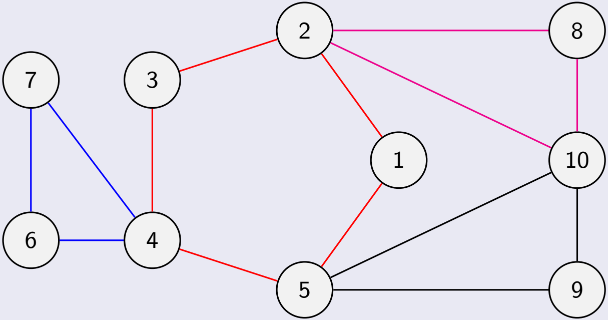
Cycle eulérien

Graphes : définitions

M2201
Graphes et langages

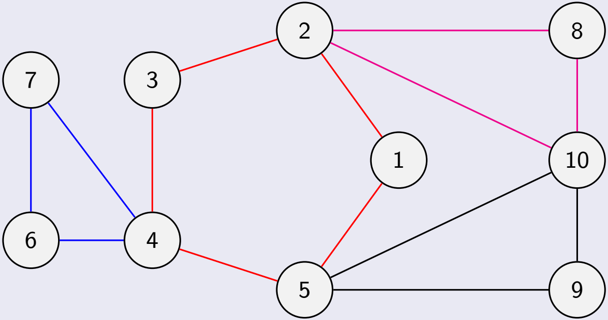
Département Informatique IUT de Saint-Dié

Cycle eulérien



Graphes : définitions

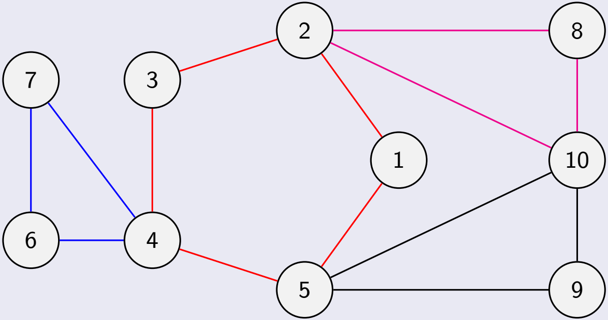
Cycle eulérien



Cycle 1 Cycle 2 Cycle 3 Cycle 4

Graphes : définitions

Cycle eulérien

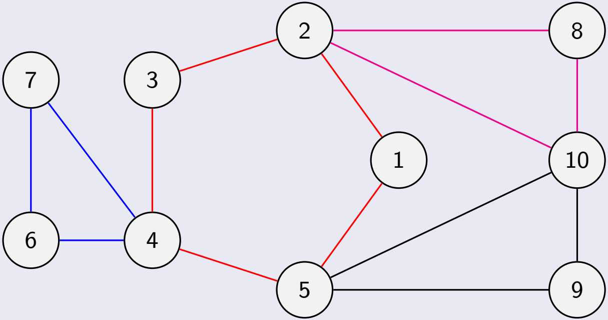


Cycle 1 | Cycle 2 | Cycle 3 | Cycle 4

1 - 2 - 3 - 4 - 5 - 1

Graphes : définitions

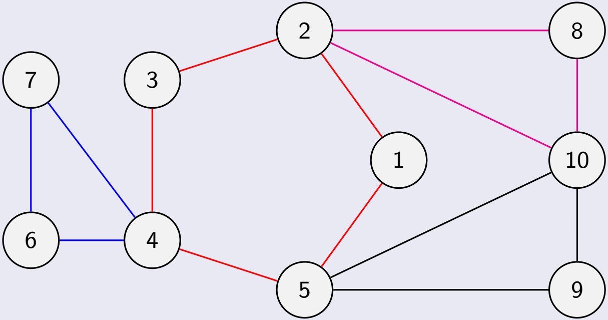
Cycle eulérien



Cycle 1	Cycle 2	Cycle 3	Cycle 4
1 - 2 - 3 - 4 - 5 - 1	$\boxed{2}$ - 8 - 10 - $\boxed{2}$		

Graphes : définitions

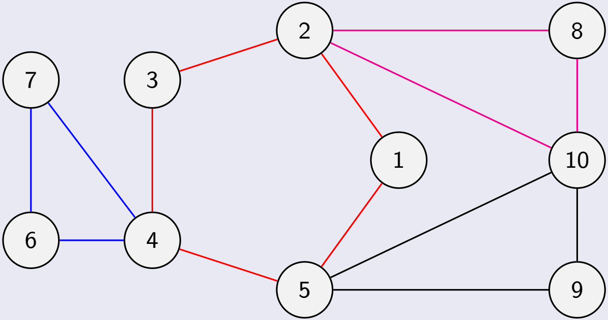
Cycle eulérien



Cycle 1	Cycle 2	Cycle 3	Cycle 4
1 - 2 - 3 - 4 - 5 - 1	$\boxed{2}$ - 8 - 10 - $\boxed{2}$	$\boxed{4}$ - 6 - 7 - $\boxed{4}$	

Graphes : définitions

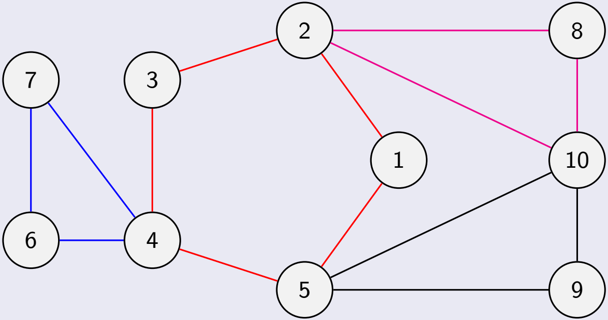
Cycle eulérien



Cycle 1	Cycle 2	Cycle 3	Cycle 4
1 - 2 - 3 - 4 - 5 - 1	$\boxed{2}$ - 8 - 10 - $\boxed{2}$	$\boxed{4}$ - 6 - 7 - $\boxed{4}$	$\boxed{5}$ - 9 - 10 - $\boxed{5}$

Graphes : définitions

Cycle eulérien



Cycle 1	Cycle 2	Cycle 3	Cycle 4
1 - 2 - 3 - 4 - 5 - 1	$\boxed{2}$ - 8 - 10 - $\boxed{2}$	$\boxed{4}$ - 6 - 7 - $\boxed{4}$	$\boxed{5}$ - 9 - 10 - $\boxed{5}$

Cycle eulérien : 1 - $\boxed{2}$ - 8 - 10 - $\boxed{2}$ - 3 - $\boxed{4}$ - 6 - 7 - $\boxed{4}$ - $\boxed{5}$ - 9 - 10 - $\boxed{5}$ - 1

Graphes : définitions

Chaîne eulérienne

Chaîne eulérienne

Théorème 2

Graphes : définitions

Chaîne eulérienne

Théorème 2 : un graphe connexe $\mathcal{G}$ admet une chaîne eulérienne distincte d'un cycle si et seulement si le nombre de sommets de $\mathcal{G}$ de degré impair est égal à 2.

Graphes : définitions

Chaîne eulérienne

Théorème 2 : un graphe connexe $\mathcal{G}$ admet une chaîne eulérienne distincte d'un cycle si et seulement si le nombre de sommets de $\mathcal{G}$ de degré impair est égal à 2.

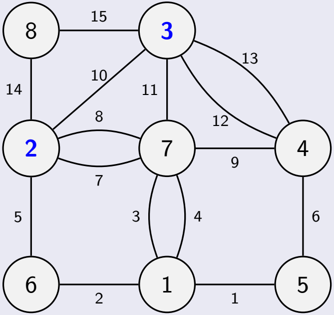
Dans ce cas, si i et j sont les deux sommets de $\mathcal{G}$ de degré impair, alors le graphe $\mathcal{G}$ admet une chaîne eulérienne d'extrémités i et j.

Graphes : définitions

Chaîne eulérienne

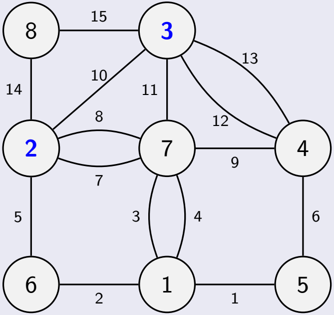
Graphes : définitions

Chaîne eulérienne



Graphes : définitions

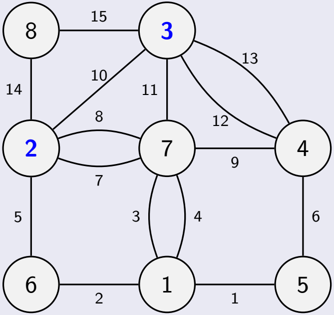
Chaîne eulérienne



Deux sommets de degré impair : 2 et 3.

Graphes : définitions

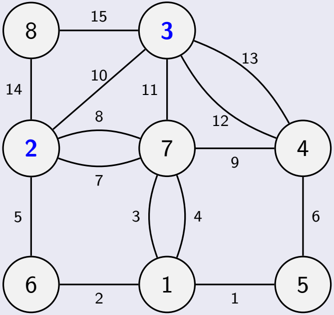
Chaîne eulérienne



Deux sommets de degré impair : 2 et 3.
Chaîne eulérienne du sommet 2 au sommet 3 :

Graphes : définitions

Chaîne eulérienne



Deux sommets de degré impair : 2 et 3.
Chaîne eulérienne du sommet 2 au sommet 3 :
7 - 3 - 2 - 5 - 14 - 15 - 13 - 12 - 11 - 9 - 6 - 1 - 4 - 8 - 10

Graphes : définitions

Chaîne eulérienne

Graphes : définitions

Chaîne eulérienne

Un graphe $\mathcal{G}$ est dit

1. eulérien s'il admet un cycle eulérien.

Graphes : définitions

Chaîne eulérienne

Un graphe $\mathcal{G}$ est dit

1. eulérien s'il admet un cycle eulérien.
2. semi-eulérien s'il admet une chaîne eulérienne.

Graphes : définitions

Chaîne eulérienne

Un graphe $\mathcal{G}$ est dit

1. eulérien s'il admet un cycle eulérien.
2. semi-eulérien s'il admet une chaîne eulérienne.

Un graphe connexe est semi-eulérien si et seulement s'il possède zéro ou exactement deux sommets de degré impair.

Graphes : définitions

Chaîne eulérienne

Un graphe $\mathcal{G}$ est dit

1. eulérien s'il admet un cycle eulérien.
2. semi-eulérien s'il admet une chaîne eulérienne.

Un graphe connexe est semi-eulérien si et seulement s'il possède zéro ou exactement deux sommets de degré impair.
Si le nombre de sommets impairs

1. est nul, la chaîne est un cycle et le graphe est alors eulérien.

Graphes : définitions

Chaîne eulérienne

Un graphe $\mathcal{G}$ est dit

1. eulérien s'il admet un cycle eulérien.
2. semi-eulérien s'il admet une chaîne eulérienne.

Un graphe connexe est semi-eulérien si et seulement s'il possède zéro ou exactement deux sommets de degré impair.
Si le nombre de sommets impairs

1. est nul, la chaîne est un cycle et le graphe est alors eulérien.
2. est égal à deux, les chaînes eulériennes du graphe auront ces deux sommets comme extrémités.

Graphes : définitions

Chemin hamiltonien, chaîne hamiltonienne

Graphes : définitions

Chemin hamiltonien, chaîne hamiltonienne

Un chemin hamiltonien (chaîne hamiltonienne) est un chemin (une chaîne) passant une seule fois par chacun des sommets d'un graphe $\mathcal{G}$.

Graphes : définitions

Chemin hamiltonien, chaîne hamiltonienne

Un chemin hamiltonien (chaîne hamiltonienne) est un chemin (une chaîne) passant une seule fois par chacun des sommets d'un graphe $\mathcal{G}$.
Un circuit (cycle) hamiltonien est un circuit (cycle) passant une seule fois par chacun des sommets d'un graphe $\mathcal{G}$.

Graphes : définitions

Chemin hamiltonien, chaîne hamiltonienne

Un chemin hamiltonien (chaîne hamiltonienne) est un chemin (une chaîne) passant une seule fois par chacun des sommets d'un graphe $\mathcal{G}$.
Un circuit (cycle) hamiltonien est un circuit (cycle) passant une seule fois par chacun des sommets d'un graphe $\mathcal{G}$.
Un graphe hamiltonien est un graphe qui comporte un circuit (cycle) hamiltonien.

Graphes : définitions

Chemin hamiltonien, chaîne hamiltonienne

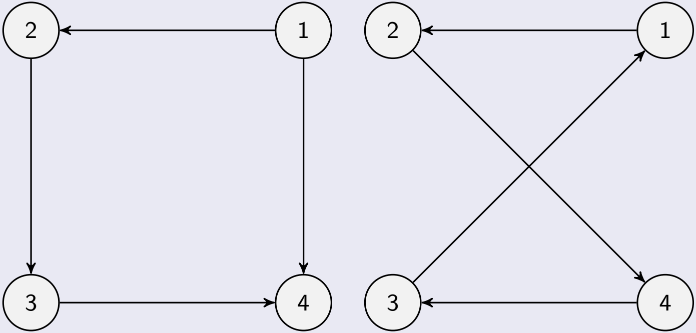
Un chemin hamiltonien (chaîne hamiltonienne) est un chemin (une chaîne) passant une seule fois par chacun des sommets d'un graphe $\mathcal{G}$.
Un circuit (cycle) hamiltonien est un circuit (cycle) passant une seule fois par chacun des sommets d'un graphe $\mathcal{G}$.
Un graphe hamiltonien est un graphe qui comporte un circuit (cycle) hamiltonien.
Sir William Rowan Hamilton (1805 - 1865) est un mathématicien, physicien et astronome irlandais (né et mort à Dublin).

Graphes : définitions

Chemin, circuit hamiltonien

Graphes : définitions

Chemin, circuit hamiltonien

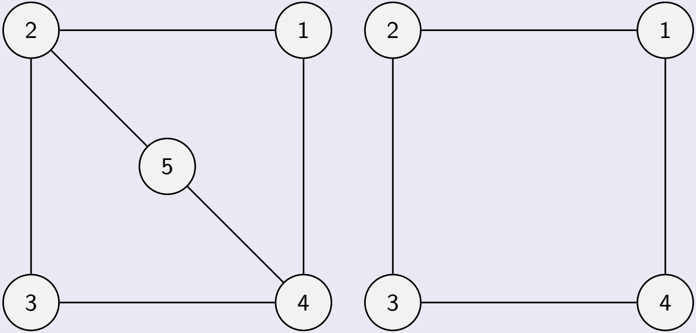


Graphes : définitions

Chaîne hamiltonienne, cycle hamiltonien

Graphes : définitions

Chaîne hamiltonienne, cycle hamiltonien



Graphes : définitions

Coloration d'un graphe

Graphes : définitions

Coloration d'un graphe

La coloration des sommets d'un graphe non orienté consiste à affecter une couleur à chacun des sommets de manière que deux sommets adjacents n'aient pas la même couleur.

Graphes : définitions

Coloration d'un graphe

La coloration des sommets d'un graphe non orienté consiste à affecter une couleur à chacun des sommets de manière que deux sommets adjacents n'aient pas la même couleur.
Le nombre chromatique d'un graphe $\mathcal{G}$ est le nombre $\gamma(\mathcal{G})$ minimum de couleurs distinctes nécessaires à la coloration des sommets.

Graphes : définitions

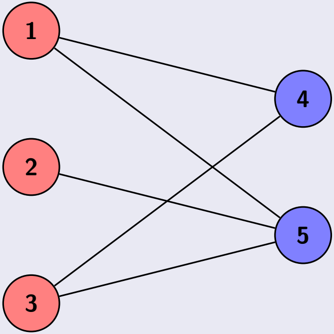
Coloration d'un graphe

La coloration des sommets d'un graphe non orienté consiste à affecter une couleur à chacun des sommets de manière que deux sommets adjacents n'aient pas la même couleur.
Le nombre chromatique d'un graphe $\mathcal{G}$ est le nombre $\gamma(\mathcal{G})$ minimum de couleurs distinctes nécessaires à la coloration des sommets.
La coloration des arêtes est définie de la même manière en échangeant les arêtes et les sommets.

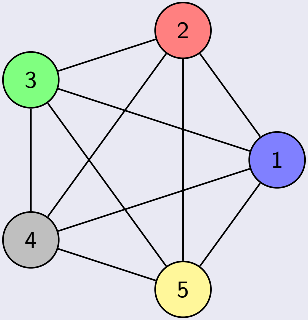
Coloration d'un graphe : exemples

Graphes : définitions

Coloration d'un graphe : exemples



$\gamma(\mathcal{G}) = 2$

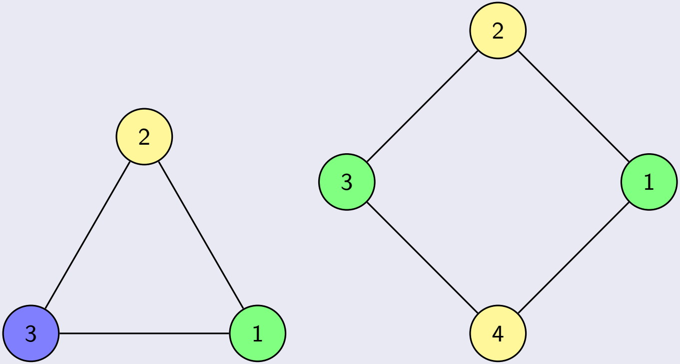


$\gamma(\mathcal{G}) = 5$

Coloration d'un graphe : cycles

Graphes : définitions

Coloration d'un graphe : cycles



$\gamma(\mathcal{G}) = 3$ $\gamma(\mathcal{G}) = 2$

Graphes : définitions

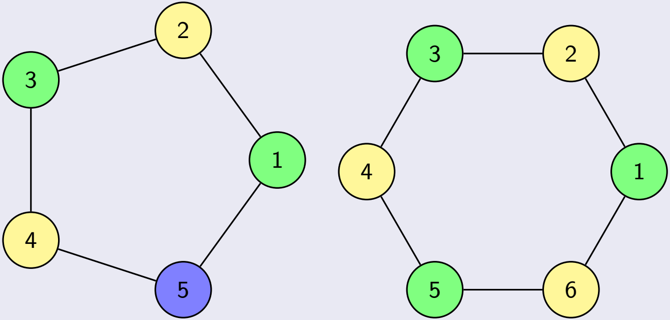
Coloration d'un graphe : cycles

Graphes : définitions

M2201
Graphes et
langages

Département
Informatique
IUT de
Saint-Dié

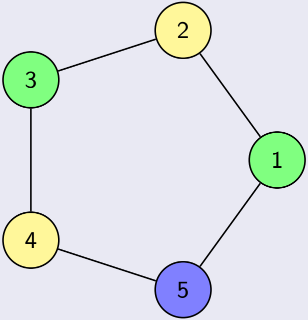
Coloration d'un graphe : cycles



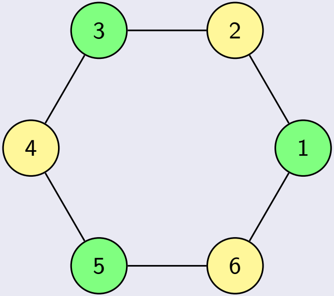
$\gamma(\mathcal{G}) = 3$ $\gamma(\mathcal{G}) = 2$

Graphes : définitions

Coloration d'un graphe : cycles



$\gamma(\mathcal{G}) = 3$



$\gamma(\mathcal{G}) = 2$

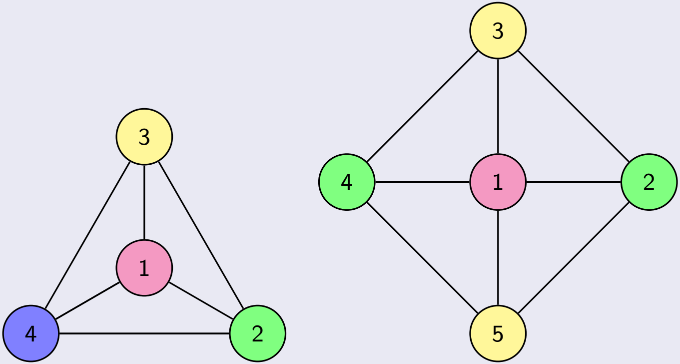
Le nombre chromatique d'un cycle d'ordre impair (respectivement pair) est 3 (respectivement 2).

Graphes : définitions

Coloration d'un graphe : roues (Wheel Graph)

Graphes : définitions

Coloration d'un graphe : roues (Wheel Graph)

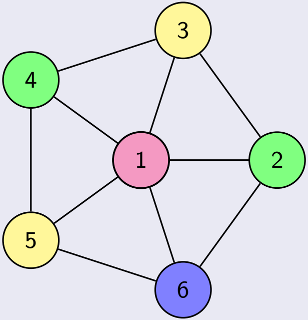


$\gamma(W_3) = 4$ $\gamma(W_4) = 3$

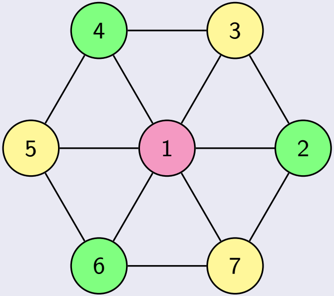
Coloration d'un graphe : roues (Wheel Graph)

Graphes : définitions

Coloration d'un graphe : roues (Wheel Graph)



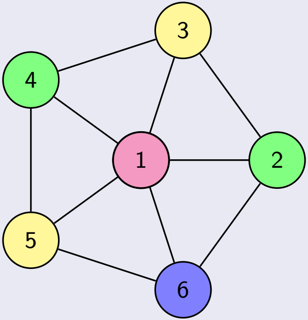
$\gamma(W_5) = 4$



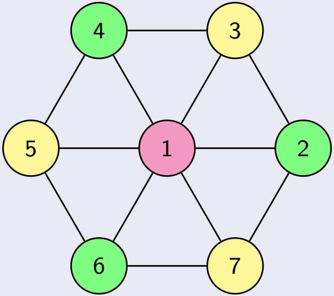
$\gamma(W_6) = 3$

Graphes : définitions

Coloration d'un graphe : roues (Wheel Graph)



$\gamma(W_5) = 4$



$\gamma(W_6) = 3$

Le nombre chromatique d'une roue ayant un nombre pair (respectivement impair) de sommets est 4 (respectivement 3).

Graphes : définitions

Coloration des arêtes

Graphes : définitions

Coloration des arêtes

La coloration des arêtes est définie de la même manière et se ramène au cas précédent en construisant le graphe obtenu en échangeant les arêtes et les sommets.

Coloration des arêtes

La coloration des arêtes est définie de la même manière et se ramène au cas précédent en construisant le graphe obtenu en échangeant les arêtes et les sommets.

1. les sommets sont les arêtes,

Graphes : définitions

Coloration des arêtes

La coloration des arêtes est définie de la même manière et se ramène au cas précédent en construisant le graphe obtenu en échangeant les arêtes et les sommets.

1. les sommets sont les arêtes,
2. les sommets du nouveau graphe sont adjacents lorsque les arêtes de l'ancien le sont.

Graphes : définitions

Coloration des arêtes

La coloration des arêtes est définie de la même manière et se ramène au cas précédent en construisant le graphe obtenu en échangeant les arêtes et les sommets.

1. les sommets sont les arêtes,
2. les sommets du nouveau graphe sont adjacents lorsque les arêtes de l'ancien le sont.

Le graphe obtenu $\mathcal{G}'$ obtenu est appelé graphe adjoint $\mathcal{G}'$ du graphe de départ $\mathcal{G}$.

Graphes : définitions

Coloration des arêtes

La coloration des arêtes est définie de la même manière et se ramène au cas précédent en construisant le graphe obtenu en échangeant les arêtes et les sommets.

1. les sommets sont les arêtes,
2. les sommets du nouveau graphe sont adjacents lorsque les arêtes de l'ancien le sont.

Le graphe obtenu $\mathcal{G}'$ obtenu est appelé graphe adjoint $\mathcal{G}'$ du graphe de départ $\mathcal{G}$.
Le nombre minimal de couleurs distinctes nécessaires à la coloration des arêtes du graphe $\mathcal{G}$ est appelé l'indice chromatique de $\mathcal{G}$ (notation : $q(\mathcal{G})$).

Graphes : définitions

Coloration des arêtes

La coloration des arêtes est définie de la même manière et se ramène au cas précédent en construisant le graphe obtenu en échangeant les arêtes et les sommets.

1. les sommets sont les arêtes,
2. les sommets du nouveau graphe sont adjacents lorsque les arêtes de l'ancien le sont.

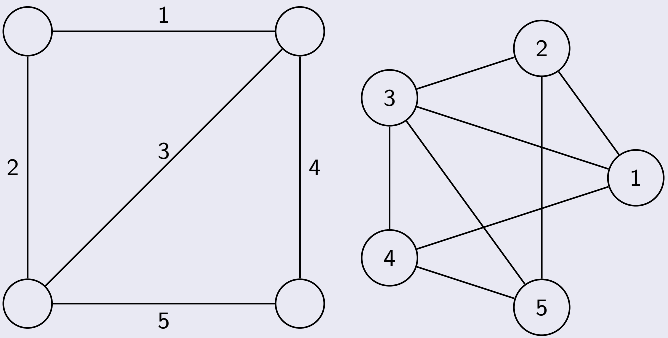
Le graphe obtenu $\mathcal{G}'$ obtenu est appelé graphe adjoint $\mathcal{G}'$ du graphe de départ $\mathcal{G}$.
Le nombre minimal de couleurs distinctes nécessaires à la coloration des arêtes du graphe $\mathcal{G}$ est appelé l'indice chromatique de $\mathcal{G}$ (notation : $q(\mathcal{G})$).
Il s'agit du nombre chromatique du graphe adjoint $\mathcal{G}'$: $q(\mathcal{G}) = \gamma(\mathcal{G}')$.

Graphes : définitions

Coloration des arêtes

Graphes : définitions

Coloration des arêtes

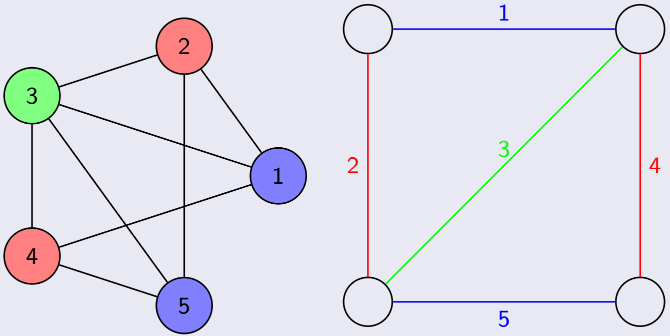


Graphes : définitions

Coloration des arêtes

Graphes : définitions

Coloration des arêtes



Le théorème des quatre couleurs, Augustus De Morgan

Graphes : définitions

Le théorème des quatre couleurs, Augustus De Morgan

Il suffit de quatre couleurs pour colorier n'importe quelle carte géographique en faisant en sorte que deux pays ayant une frontière commune (et non réduite à un point) ne soient jamais coloriés avec la même couleur et ce, quelles que soient la forme des frontières (1852).

Graphes : définitions

Le théorème des quatre couleurs, Augustus De Morgan

Il suffit de quatre couleurs pour colorier n'importe quelle carte géographique en faisant en sorte que deux pays ayant une frontière commune (et non réduite à un point) ne soient jamais coloriés avec la même couleur et ce, quelles que soient la forme des frontières (1852).

Démonstration en octobre 1977 par Kenneth Appel (américain) et Wolfgang Haken (allemand) : première preuve assistée par ordinateur de l'histoire.

Graphes : définitions

Le théorème des quatre couleurs, Augustus De Morgan

Il suffit de quatre couleurs pour colorier n'importe quelle carte géographique en faisant en sorte que deux pays ayant une frontière commune (et non réduite à un point) ne soient jamais coloriés avec la même couleur et ce, quelles que soient la forme des frontières (1852).

Démonstration en octobre 1977 par Kenneth Appel (américain) et Wolfgang Haken (allemand) : première preuve assistée par ordinateur de l'histoire.

Les deux mathématiciens simplifient le problème de l'étude d'une infinité de cas en celui de l'examen d'un nombre fini de situations (1936 exactement) auxquelles on peut ramener toutes les autres.

Le théorème des quatre couleurs, Augustus De Morgan

Il suffit de quatre couleurs pour colorier n'importe quelle carte géographique en faisant en sorte que deux pays ayant une frontière commune (et non réduite à un point) ne soient jamais coloriés avec la même couleur et ce, quelles que soient la forme des frontières (1852).
Démonstration en octobre 1977 par Kenneth Appel (américain) et Wolfgang Haken (allemand) : première preuve assistée par ordinateur de l'histoire.
Les deux mathématiciens simplifient le problème de l'étude d'une infinité de cas en celui de l'examen d'un nombre fini de situations (1936 exactement) auxquelles on peut ramener toutes les autres.

Augustus De Morgan (18061871) est un mathématicien britannique.

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Le théorème des quatre couleurs : autre formulation

Graphes : définitions

Le théorème des quatre couleurs : autre formulation

Un graphe planaire est un graphe qui peut être représenté dans un plan sans qu'aucune arête n'en croise une autre.

Graphes : définitions

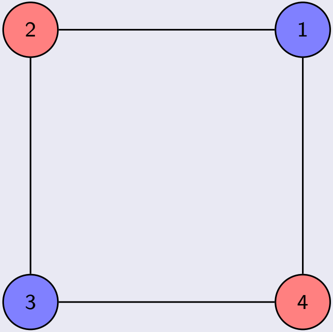
Le théorème des quatre couleurs : autre formulation

Un graphe planaire est un graphe qui peut être représenté dans un plan sans qu'aucune arête n'en croise une autre.
Tout graphe planaire est coloriable en 4 couleurs.

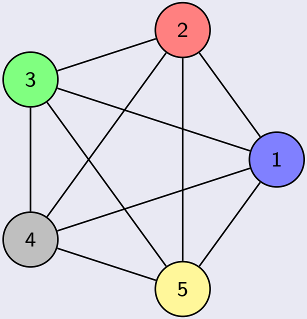
Graphes : définitions

Le théorème des quatre couleurs : autre formulation

Un graphe planaire est un graphe qui peut être représenté dans un plan sans qu'aucune arête n'en croise une autre.
Tout graphe planaire est coloriable en 4 couleurs.



$\gamma(\mathcal{G}) = 2$

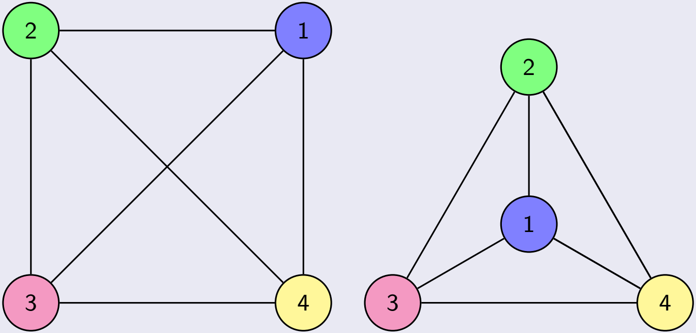


$\gamma(\mathcal{G}) = 5$

Le théorème des quatre couleurs

Graphes : définitions

Le théorème des quatre couleurs



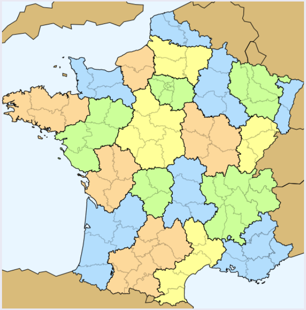
$$\gamma(\mathcal{G}) = 4$$

Graphes : définitions

Le théorème des quatre couleurs

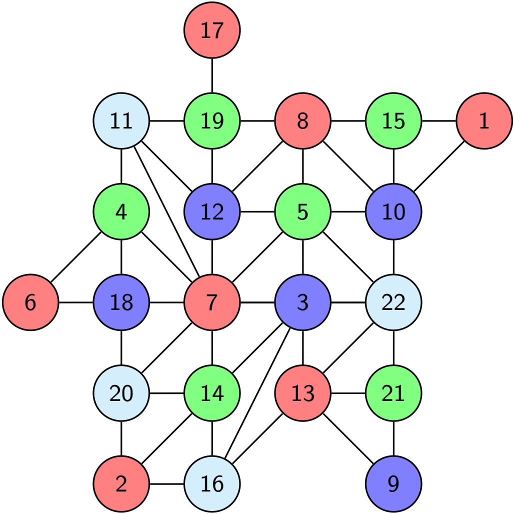
Graphes : définitions

Le théorème des quatre couleurs



TomKr

Graphes : définitions



$\gamma(\mathcal{G}) = 4$

Graphes : définitions

Le théorème des quatre couleurs

Graphes : définitions

Le théorème des quatre couleurs

ThomasCVB ⓘ

Graphes : définitions

Algorithme de Welsh-Powell

Graphes : définitions

Algorithme de Welsh-Powell

On note L la liste des sommets s_i classés suivant l'ordre décroissant de leur degré : $d(s_1) \geq d(s_2) \geq d(s_3) \geq ... \geq d(s_n)$.

Graphes : définitions

Algorithme de Welsh-Powell

On note L la liste des sommets s_i classés suivant l'ordre décroissant de leur degré : $d(s_1) \geq d(s_2) \geq d(s_3) \geq ... \geq d(s_n)$.

Initialisation :
L : liste des sommets dans l'ordre décroissant du degré
couleur $= 0$

```
Tant que L ≠ ∅ faire
        couleur=couleur+1
        couleur(s) = couleur
        Pour tout t dans L faire
                Si t ∉ Γs
                    couleur(t)=couleur
                    Γs = Γs ∪ Γt
                Fin si
        Fin faire
                Retirer de L les sommets portant une couleur
Fin faire
```

Graphes : définitions

Algorithme de Welsh-Powell

On note L la liste des sommets s_i classés suivant l'ordre décroissant de leur degré : $d(s_1) \geq d(s_2) \geq d(s_3) \geq ... \geq d(s_n)$.

Initialisation :
L : liste des sommets dans l'ordre décroissant du degré
couleur $= 0$

Tant que $L \neq \emptyset$ faire
  couleur=couleur+1
  couleur(s) = couleur
  Pour tout t dans L faire
    Si $t \notin \Gamma_s$
      couleur(t)=couleur
      $\Gamma_s = \Gamma_s \cup \Gamma_t$
    Fin si
  Fin faire
    Retirer de L les sommets portant une couleur
Fin faire

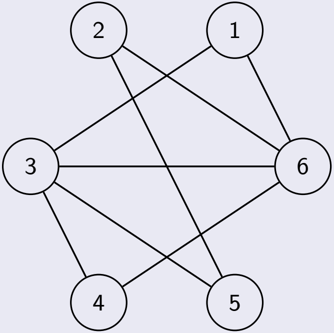
Il s'agit d'attribuer au premier sommet s_i non encore coloré la plus petite couleur non déjà affectée aux sommets s_1, ...,s_{i-1} déjà colorés, ainsi qu'à tous les sommets non adjacents.

Graphes : définitions

Algorithme de Welsh-Powell : exemple

Graphes : définitions

Algorithme de Welsh-Powell : exemple



Graphes : définitions

Algorithme de Welsh-Powell : exemple

Graphes : définitions

Algorithme de Welsh-Powell : exemple

Sommet	degré	Γ
1	2	$\{3,6\}$
2	2	$\{5,6\}$
3	4	$\{1,4,5,6\}$
4	2	$\{3,6\}$
5	2	$\{2,3\}$
6	4	$\{1,2,3,4\}$

Graphes : définitions

Algorithme de Welsh-Powell : exemple

Sommet	degré	Γ
1	2	$\{3,6\}$
2	2	$\{5,6\}$
3	4	$\{1,4,5,6\}$
4	2	$\{3,6\}$
5	2	$\{2,3\}$
6	4	$\{1,2,3,4\}$

Liste des sommets suivant l'ordre décroissant des degrés choisie :

Graphes : définitions

Algorithme de Welsh-Powell : exemple

Sommet	degré	Γ
1	2	$\{3,6\}$
2	2	$\{5,6\}$
3	4	$\{1,4,5,6\}$
4	2	$\{3,6\}$
5	2	$\{2,3\}$
6	4	$\{1,2,3,4\}$

Liste des sommets suivant l'ordre décroissant des degrés choisie :

$\boxed{L = (6, 3, 5, 4, 2, 1)}$

Graphes : définitions

Algorithme de Welsh-Powell : exemple

Sommet	degré	Γ
1	2	{3,6}
2	2	{5,6}
3	4	{1,4,5,6}
4	2	{3,6}
5	2	{2,3}
6	4	{1,2,3,4}

Liste des sommets suivant l'ordre décroissant des degrés choisie :

$\boxed{L = (6, 3, 5, 4, 2, 1)}$

Couleur	6	3	5	4	2	1	Γ	L
C_1	C_1	.	C_1	.	.	.	{1,2,3,4}	(3,4,2,1)

Graphes : définitions

Algorithme de Welsh-Powell : exemple

Sommet	degré	Γ
1	2	{3,6}
2	2	{5,6}
3	4	{1,4,5,6}
4	2	{3,6}
5	2	{2,3}
6	4	{1,2,3,4}

Liste des sommets suivant l'ordre décroissant des degrés choisie :

$\boxed{L = (6, 3, 5, 4, 2, 1)}$

Couleur	6	3	5	4	2	1	Γ	L
C_1	C_1	.	C_1	.	.	.	{1,2,3,4}	(3,4,2,1)
C_2	x	C_2	x	.	C_2	.	{1,4,5,6}	(4,1)

Graphes : définitions

Algorithme de Welsh-Powell : exemple

Sommet	degré	Γ
1	2	$\{3,6\}$
2	2	$\{5,6\}$
3	4	$\{1,4,5,6\}$
4	2	$\{3,6\}$
5	2	$\{2,3\}$
6	4	$\{1,2,3,4\}$

Liste des sommets suivant l'ordre décroissant des degrés choisie :

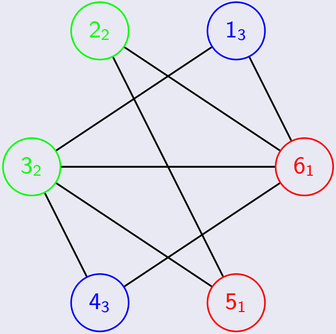
$\boxed{L = (6,3,5,4,2,1)}$

Couleur	6	3	5	4	2	1	Γ	L
C_1	C_1	.	C_1	.	.	.	$\{1,2,3,4\}$	$(3,4,2,1)$
C_2	x	C_2	x	.	C_2	.	$\{1,4,5,6\}$	$(4,1)$
C_3	x	x	x	C_3	x	C_3	$\{3,6\}$	$\emptyset$

Algorithme de Welsh-Powell : exemple

Graphes : définitions

Algorithme de Welsh-Powell : exemple



Graphes : définitions

Algorithme de Welsh-Powell (1967)

Graphes : définitions

Algorithme de Welsh-Powell (1967)

Dominic James Anthony Welsh (né en 1938) : mathématicien anglais.

Graphes : définitions

Algorithme de Welsh-Powell (1967)

Dominic James Anthony Welsh (né en 1938) : mathématicien anglais.
M. B. Powell

Le problème du plus court chemin

Définitions

Le problème du plus court chemin

Définitions

A chaque arc α d'un graphe $\mathcal{G} = (E, \mathcal{A})$, on associe un nombre $l(\alpha)$ appelé longueur de l'arc α.

Le problème du plus court chemin

Définitions

A chaque arc α d'un graphe $\mathcal{G} = (E, \mathcal{A})$, on associe un nombre $l(\alpha)$ appelé longueur de l'arc α.
On obtient ainsi un graphe valué par les longueurs $l(\alpha)$.

Le problème du plus court chemin

Définitions

A chaque arc α d'un graphe $\mathcal{G} = (E, \mathcal{A})$, on associe un nombre $l(\alpha)$ appelé longueur de l'arc α.
On obtient ainsi un graphe valué par les longueurs $l(\alpha)$.
Le problème du plus court chemin entre deux sommets i et j est de trouver un chemin $C(i, j)$ de i à j dont la longueur totale soit minimum.

Le problème du plus court chemin

Exemple

Le problème du plus court chemin

Exemple

Dans le cas du graphe des accès possibles à Saint-Dié à partir des grandes villes périphériques on associe à chaque arc la distance entre les deux villes correspondantes.

Le problème du plus court chemin

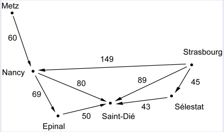
Exemple

Dans le cas du graphe des accès possibles à Saint-Dié à partir des grandes villes périphériques on associe à chaque arc la distance entre les deux villes correspondantes.
l(Strasbourg, Saint-Dié)=89, l(Strasbourg, Nancy)=149, l(Strasbourg, Sélestat)=45, l(Sélestat, Saint-Dié)=43, l(Metz, Nancy)=60, l(Nancy, Epinal)=69, l(Nancy, Saint-Dié)=80, l(Epinal, Saint-Dié)=50.

Le problème du plus court chemin

Exemple

Dans le cas du graphe des accès possibles à Saint-Dié à partir des grandes villes périphériques on associe à chaque arc la distance entre les deux villes correspondantes.
l(Strasbourg, Saint-Dié)=89, l(Strasbourg, Nancy)=149, l(Strasbourg, Sélestat)=45, l(Sélestat, Saint-Dié)=43, l(Metz, Nancy)=60, l(Nancy, Epinal)=69, l(Nancy, Saint-Dié)=80, l(Epinal, Saint-Dié)=50.



Algorithme de Moore-Dijkstra

Algorithme

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i, j) = l_{ij} \geq 0$,

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i, j) = l_{ij} \geq 0$,
et on recherche la longueur d'un plus court chemin du sommet 1 aux autres.

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i, j) = l_{ij} \geq 0$,
et on recherche la longueur d'un plus court chemin du sommet 1 aux autres.
On définit Δ_i ou $\Delta(i)$ comme la longueur minimale des chemins de 1 à i.

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i, j) = l_{ij} \geq 0$,
et on recherche la longueur d'un plus court chemin du sommet 1 aux autres.
On définit Δ_i ou $\Delta(i)$ comme la longueur minimale des chemins de 1 à i.
L'algorithme consiste en $n - 1$ itérations.

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i, j) = l_{ij} \geq 0$,
et on recherche la longueur d'un plus court chemin du sommet 1 aux autres.
On définit Δ_i ou $\Delta(i)$ comme la longueur minimale des chemins de 1 à i.
L'algorithme consiste en $n - 1$ itérations.
Au début de chaque itération, l'ensemble des sommets est partitionné en deux sous-ensembles :

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i,j) = l_{ij} \geq 0$,
et on recherche la longueur d'un plus court chemin du sommet 1 aux autres.
On définit Δ_i ou $\Delta(i)$ comme la longueur minimale des chemins de 1 à i.
L'algorithme consiste en $n-1$ itérations.
Au début de chaque itération, l'ensemble des sommets est partitionné en deux sous-ensembles :

1. S

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i,j) = l_{ij} \geq 0$,
et on recherche la longueur d'un plus court chemin du sommet 1 aux autres.
On définit Δ_i ou $\Delta(i)$ comme la longueur minimale des chemins de 1 à i.
L'algorithme consiste en $n-1$ itérations.
Au début de chaque itération, l'ensemble des sommets est partitionné en deux sous-ensembles :

1. S
2. $\overline{S} = E - S$

Algorithme de Moore-Dijkstra

Algorithme

On note $E = \{1, 2, 3, ..., n\}$, $l(i, j) = l_{ij} \geq 0$,
et on recherche la longueur d'un plus court chemin du sommet 1 aux autres.
On définit Δ_i ou $\Delta(i)$ comme la longueur minimale des chemins de 1 à i.
L'algorithme consiste en $n - 1$ itérations.
Au début de chaque itération, l'ensemble des sommets est partitionné en deux sous-ensembles :

1. S
2. $\overline{S} = E - S$

avec $1 \in S$.

Algorithme de Moore-Dijkstra (1959)

Algorithme

Algorithme de Moore-Dijkstra (1959)

Algorithme

Chaque sommet i de E est affecté du nombre $\delta_i = \delta(i)$ qui vérifie :

1. si $i \in S$ alors $\delta_i = \Delta_i$

Algorithme de Moore-Dijkstra (1959)

Algorithme

Chaque sommet i de E est affecté du nombre $\delta_i = \delta(i)$ qui vérifie :

1. si $i \in S$ alors $\delta_i = \Delta_i$
2. si $i \in \overline{S}$ alors $\delta_i = \min\limits_{j \in S \cap \Gamma_i^{-1}} (\delta_j + l_{ji})$

Algorithme de Moore-Dijkstra (1959)

Algorithme

Chaque sommet i de E est affecté du nombre $\delta_i = \delta(i)$ qui vérifie :

1. si $i \in S$ alors $\delta_i = \Delta_i$
2. si $i \in \overline{S}$ alors $\delta_i = \min\limits_{j \in S \cap \Gamma_i^{-1}} (\delta_j + l_{ji})$

δ_i pour $i \in \overline{S}$ donne la longueur minimale des chemins de 1 à i,

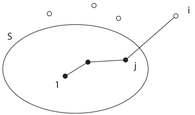
Algorithme de Moore-Dijkstra (1959)

Algorithme

Chaque sommet i de E est affecté du nombre $\delta_i = \delta(i)$ qui vérifie :

1. si $i \in S$ alors $\delta_i = \Delta_i$
2. si $i \in \overline{S}$ alors $\delta_i = \min\limits_{j \in S \cap \Gamma_i^{-1}} (\delta_j + l_{ji})$

δ_i pour $i \in \overline{S}$ donne la longueur minimale des chemins de 1 à i, soumis à la condition que tous ses sommets exceptés i sont dans S.



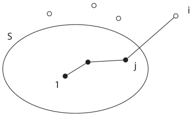
Algorithme de Moore-Dijkstra (1959)

Algorithme

Chaque sommet i de E est affecté du nombre $\delta_i = \delta(i)$ qui vérifie :

1. si $i \in S$ alors $\delta_i = \Delta_i$
2. si $i \in \overline{S}$ alors $\delta_i = \min\limits_{j \in S \cap \Gamma_i^{-1}} (\delta_j + l_{ji})$

δ_i pour $i \in \overline{S}$ donne la longueur minimale des chemins de 1 à i, soumis à la condition que tous ses sommets exceptés i sont dans S.



Edsger Wybe Dijkstra (1930 - 2002) est un mathématicien et informaticien néerlandais.

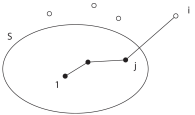
Algorithme de Moore-Dijkstra (1959)

Algorithme

Chaque sommet i de E est affecté du nombre $\delta_i = \delta(i)$ qui vérifie :

1. si $i \in S$ alors $\delta_i = \Delta_i$
2. si $i \in \overline{S}$ alors $\delta_i = \min\limits_{j \in S \cap \Gamma_i^{-1}} (\delta_j + l_{ji})$

δ_i pour $i \in \overline{S}$ donne la longueur minimale des chemins de 1 à i, soumis à la condition que tous ses sommets exceptés i sont dans S.



Edsger Wybe Dijkstra (1930 - 2002) est un mathématicien et informaticien néerlandais.
J Strother Moore est un informaticien américain.

Algorithme de Moore-Dijkstra

Algorithme

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$,

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$,

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
$\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$, $\delta_i = l_{1i}$ si $i \in \Gamma_1$,

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$, $\delta_i = l_{1i}$ si $i \in \Gamma_1$, ∞ sinon.

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$, $\delta_i = l_{1i}$ si $i \in \Gamma_1$, ∞ sinon.
2. a) Sélectionner $i \in \overline{S}$ tel que $\delta_i = \min\limits_{j \in \overline{S}} \delta_j$

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$, $\delta_i = l_{1i}$ si $i \in \Gamma_1$, ∞ sinon.
2. a) Sélectionner $i \in \overline{S}$ tel que $\delta_i = \min\limits_{j \in \overline{S}} \delta_j$

   b) $\overline{S} \leftarrow \overline{S} - \{i\}$

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes

Généralités

Arbre

Cycle eulérien, hamiltonien

Coloration

Le problème du plus court chemin

Arbre recouvrant de poids minimal

Langages et automates

Langages

Automates

Annexes

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$, $\delta_i = l_{1i}$ si $i \in \Gamma_1$, ∞ sinon.
2. a) Sélectionner $i \in \overline{S}$ tel que $\delta_i = \min\limits_{j \in \overline{S}} \delta_j$

   b) $\overline{S} \leftarrow \overline{S} - \{i\}$

   c) Si $\overline{S} = \{\}$ alors *FIN* sinon aller en 3

Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$, $\delta_i = l_{1i}$ si $i \in \Gamma_1$, ∞ sinon.
2. a) Sélectionner $i \in \overline{S}$ tel que $\delta_i = \min_{j \in \overline{S}} \delta_j$

   b) $\overline{S} \leftarrow \overline{S} - \{i\}$
   c) Si $\overline{S} = \{\}$ alors *FIN* sinon aller en 3
3. Faire pour tout $j \in \Gamma_i \cap \overline{S}$:

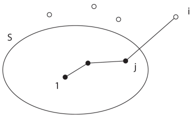
Algorithme de Moore-Dijkstra

Algorithme

1. Initialisation
   $\overline{S} = \{2, 3, \cdots, n\}$, $\delta_1 = 0$, $\delta_i = l_{1i}$ si $i \in \Gamma_1$, ∞ sinon.
2. a) Sélectionner $i \in \overline{S}$ tel que $\delta_i = \min_{j \in \overline{S}} \delta_j$

   b) $\overline{S} \leftarrow \overline{S} - \{i\}$

   c) Si $\overline{S} = \{\}$ alors *FIN* sinon aller en 3
3. Faire pour tout $j \in \Gamma_i \cap \overline{S}$:
   $\delta_j \leftarrow \min(\delta_j, \delta_i + l_{ij})$ et aller en 2



Algorithme de Moore-Dijkstra

Démonstration

Algorithme de Moore-Dijkstra

Démonstration

On démontre :

Algorithme de Moore-Dijkstra

Démonstration

On démontre :

Soit $i \in \overline{S}$ tel que $\delta_i = \min_{j \in \overline{S}} \delta_j$.

Algorithme de Moore-Dijkstra

Démonstration

On démontre :

Soit $i \in \overline{S}$ tel que $\delta_i = \min\limits_{j \in \overline{S}} \delta_j$. On a alors $\Delta_i = \delta_i$.

Algorithme de Moore-Dijkstra

Démonstration

On démontre :

Soit $i \in \overline{S}$ tel que $\delta_i = \min_{j \in \overline{S}} \delta_j$. On a alors $\Delta_i = \delta_i$.

Il existe un chemin de 1 à i de longueur δ_i.

Algorithme de Moore-Dijkstra

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Démonstration

On démontre :
Soit $i \in \overline{S}$ tel que $\delta_i = \min_{j \in \overline{S}} \delta_j$. On a alors $\Delta_i = \delta_i$.

Il existe un chemin de 1 à i de longueur δ_i.
Supposons qu'il existe un autre chemin de 1 à i : il passe par un premier sommet $k \in \overline{S}$.

Algorithme de Moore-Dijkstra

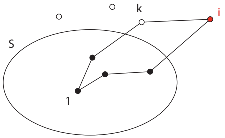
Démonstration

On démontre :
Soit $i \in \overline{S}$ tel que $\delta_i = \min\limits_{j \in \overline{S}} \delta_j$. On a alors $\Delta_i = \delta_i$.
Il existe un chemin de 1 à i de longueur δ_i.
Supposons qu'il existe un autre chemin de 1 à i : il passe par un premier sommet $k \in \overline{S}$.
Longueur du chemin de 1 à k : $\delta_k \geq \delta_i = \min\limits_{j \in \overline{S}} \delta_j$.

Algorithme de Moore-Dijkstra

Démonstration

On démontre :
Soit $i \in \overline{S}$ tel que $\delta_i = \min\limits_{j \in \overline{S}} \delta_j$. On a alors $\Delta_i = \delta_i$.
Il existe un chemin de 1 à i de longueur δ_i.
Supposons qu'il existe un autre chemin de 1 à i : il passe par un premier sommet $k \in \overline{S}$.
Longueur du chemin de 1 à k : $\delta_k \geq \delta_i = \min\limits_{j \in \overline{S}} \delta_j$.
On en déduit : longueur du chemin de 1 à i en passant par $k \geq \delta_i$.



Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 0 : initialisation

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 0 : initialisation

$\overline{S} = \{\text{Epinal, Metz, Nancy, Saint-Dié, Sélestat}\}$,

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 0 : initialisation

$\overline{S}$ = {Epinal, Metz, Nancy, Saint-Dié, Sélestat},

δ(Strasbourg)=0,

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 0 : initialisation

$\overline{S}$ = {Epinal, Metz, Nancy, Saint-Dié, Sélestat},

δ(Strasbourg)=0, δ(Nancy)=149, δ(Saint-Dié)=89, δ(Sélestat)=45,

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 0 : initialisation

$\overline{S}$ = {Epinal, Metz, Nancy, Saint-Dié, Sélestat},
δ(Strasbourg)=0, δ(Nancy)=149, δ(Saint-Dié)=89, δ(Sélestat)=45,
δ(Epinal)=δ(Metz)=∞

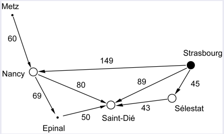
Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 0 : initialisation

$\overline{S}$ = {Epinal, Metz, Nancy, Saint-Dié, Sélestat},
δ(Strasbourg)=0, δ(Nancy)=149, δ(Saint-Dié)=89, δ(Sélestat)=45,
δ(Epinal)=δ(Metz)=∞

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞



Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 1

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 1

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45				

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 1

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88			

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 1

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 1

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞

$\delta(SD) = \min(\delta(SD), \delta(Se) + l(Se, SD)) = \min(89, 45+43) = \min(89, 88) = 88$

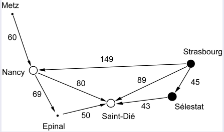
Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 1

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞

δ(SD)=min(δ(SD),δ(Se)+l(Se,SD))=min(89,45+43)=min(89,88)=88



Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 2

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 2

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 2

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 2

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 2

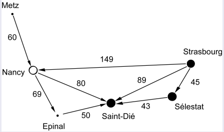
i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 2

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞



Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞

Algorithme de Moore-Dijkstra

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149		

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞

$$\delta(\mathrm{E})=\min(\delta(\mathrm{E}),\delta(\mathrm{N})+l(\mathrm{N},\mathrm{E}))=\min(\infty,149+69)=\min(\infty,218)=218$$

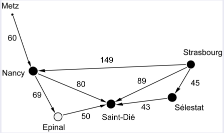
Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 3

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞

$$\delta(\mathrm{E})=\min(\delta(\mathrm{E}),\delta(\mathrm{N})+l(\mathrm{N},\mathrm{E}))=\min(\infty,149+69)=\min(\infty,218)=218$$



Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 4

Exemple : chemin le plus court à partir de Strasbourg

Etape 4

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 4

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞

Algorithme de Moore-Dijkstra

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Exemple : chemin le plus court à partir de Strasbourg

Etape 4

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 4

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 4

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞

Algorithme de Moore-Dijkstra

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Exemple : chemin le plus court à partir de Strasbourg

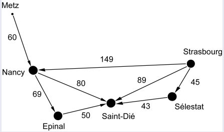
Etape 4

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞
E	$\{M\}$	$\{SD\}$	$\{\}$					218	∞

Exemple : chemin le plus court à partir de Strasbourg

Etape 4

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞
E	$\{M\}$	$\{SD\}$	$\{\}$					218	∞



Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞

Algorithme de Moore-Dijkstra

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞
E	$\{M\}$	$\{SD\}$	$\{\}$					218	∞

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞
E	$\{M\}$	$\{SD\}$	$\{\}$					218	∞
M	$\{\}$								∞

Algorithme de Moore-Dijkstra

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞
E	$\{M\}$	$\{SD\}$	$\{\}$					218	∞
M	$\{\}$								∞

Conclusion

Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞
E	$\{M\}$	$\{SD\}$	$\{\}$					218	∞
M	$\{\}$								∞

Conclusion
Δ(Epinal)=218, Δ(Metz)= ∞, Δ(Nancy)=149, Δ(Saint-Dié)=88, Δ(Sélestat)=45, Δ(Strasbourg)=0.

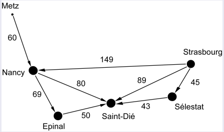
Algorithme de Moore-Dijkstra

Exemple : chemin le plus court à partir de Strasbourg

Etape 5

i	$\overline{S}$	Γ	$\overline{S} \cap \Gamma$	Stg	Se	SD	N	E	M
Stg	$\{E, M, N, SD, Se\}$	$\{N, SD, Se\}$	$\{N, SD, Se\}$	0	45	89	149	∞	∞
Se	$\{E, M, N, SD\}$	$\{SD\}$	$\{SD\}$		45	88	149	∞	∞
SD	$\{E, M, N\}$	$\{\}$	$\{\}$			88	149	∞	∞
N	$\{E, M\}$	$\{E, SD\}$	$\{E\}$				149	218	∞
E	$\{M\}$	$\{SD\}$	$\{\}$					218	∞
M	$\{\}$								∞

Conclusion
Δ(Epinal)=218, Δ(Metz)= ∞, Δ(Nancy)=149, Δ(Saint-Dié)=88, Δ(Sélestat)=45, Δ(Strasbourg)=0.
Dans le contexte, on en déduit que la distance minimale de Strasbourg à Saint-Dié est 88, le chemin correspondant est Strasbourg-Sélestat-Saint-Dié.



Algorithme matriciel de Floyd

Algorithme matriciel de Floyd (1962)

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd (1962)

On définit les matrices $L = (l_{ij})$ et $L' = (l'_{ij})$ où

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd (1962)

On définit les matrices $L = (l_{ij})$ et $L' = (l'_{ij})$ où

1. $l_{ii} = 0$, l_{ij} est la longueur de l'arc (i, j), si $(i, j) \in \mathcal{A}$, ∞ sinon,

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd (1962)

On définit les matrices $L = (l_{ij})$ et $L' = (l'_{ij})$ où

1. $l_{ii} = 0$, l_{ij} est la longueur de l'arc (i, j), si $(i, j) \in \mathcal{A}$, ∞ sinon,
2. l'_{ij} est la longueur du plus court chemin entre i et j si $j \in \widehat{\Gamma}_i$, ∞ sinon.

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd (1962)

On définit les matrices $L = (l_{ij})$ et $L' = (l'_{ij})$ où

1. $l_{ii} = 0$, l_{ij} est la longueur de l'arc (i, j), si $(i, j) \in \mathcal{A}$, ∞ sinon,
2. l'_{ij} est la longueur du plus court chemin entre i et j si $j \in \widehat{\Gamma}_i$, ∞ sinon.

Robert W. Floyd (1936 - 2001) est un informaticien américain.

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd consiste à poser $L^{(0)} = L$

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd consiste à poser $L^{(0)} = L$,
puis on calcule $L^{(1)}$ définie par $l_{ij}^{(1)} = min\left(l_{ij}^{(0)}, l_{i1}^{(0)} + l_{1j}^{(0)}\right)$

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd consiste à poser $L^{(0)} = L$,
puis on calcule $L^{(1)}$ définie par $l_{ij}^{(1)} = min\left(l_{ij}^{(0)}, l_{i1}^{(0)} + l_{1j}^{(0)}\right)$
qui est la longueur minimum des chemins de i à j ne pouvant avoir que 1 comme sommet intermédiaire.

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd consiste à poser $L^{(0)} = L$,
puis on calcule $L^{(1)}$ définie par $l_{ij}^{(1)} = min\left(l_{ij}^{(0)}, l_{i1}^{(0)} + l_{1j}^{(0)}\right)$
qui est la longueur minimum des chemins de i à j ne pouvant avoir que 1 comme sommet intermédiaire.On calcule ensuite les matrices $L^{(k)}$ définies à partir de $L^{(k-1)}$

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd consiste à poser $L^{(0)} = L$,
puis on calcule $L^{(1)}$ définie par $l_{ij}^{(1)} = min\left(l_{ij}^{(0)}, l_{i1}^{(0)} + l_{1j}^{(0)}\right)$
qui est la longueur minimum des chemins de i à j ne pouvant avoir que 1 comme sommet intermédiaire. On calcule ensuite les matrices $L^{(k)}$ définies à partir de $L^{(k-1)}$ par $l_{ij}^{(k)} = min\left(l_{ij}^{(k-1)}, l_{ik}^{(k-1)} + l_{kj}^{(k-1)}\right)$,

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd consiste à poser $L^{(0)} = L$,
puis on calcule $L^{(1)}$ définie par $l_{ij}^{(1)} = min\left(l_{ij}^{(0)}, l_{i1}^{(0)} + l_{1j}^{(0)}\right)$
qui est la longueur minimum des chemins de i à j ne pouvant avoir que 1 comme sommet intermédiaire. On calcule ensuite les matrices $L^{(k)}$ définies à partir de $L^{(k-1)}$ par $l_{ij}^{(k)} = min\left(l_{ij}^{(k-1)}, l_{ik}^{(k-1)} + l_{kj}^{(k-1)}\right)$,
qui représente la longueur minimum des chemins dont les seuls sommets intermédiaires sont les sommets de l'ensemble $\{1, 2, ..., k\}$.

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd consiste à poser $L^{(0)} = L$,
puis on calcule $L^{(1)}$ définie par $l_{ij}^{(1)} = min\left(l_{ij}^{(0)}, l_{i1}^{(0)} + l_{1j}^{(0)}\right)$
qui est la longueur minimum des chemins de i à j ne pouvant avoir que 1 comme sommet intermédiaire.On calcule ensuite les matrices $L^{(k)}$ définies à partir de $L^{(k-1)}$par $l_{ij}^{(k)} = min\left(l_{ij}^{(k-1)}, l_{ik}^{(k-1)} + l_{kj}^{(k-1)}\right)$,
qui représente la longueur minimum des chemins dont les seuls sommets intermédiaires sont les sommets de l'ensemble $\{1, 2, ..., k\}$.
On en déduit alors $L' = L^{(n)}$.

Algorithme matriciel de Floyd

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :

Pour k de 1 à n

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :

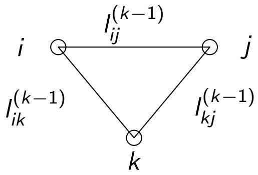
Pour k de 1 à n

Faire pour tout i et j de 1 à n $l_{ij} \leftarrow min\left(l_{ij}, l_{ik} + l_{kj}\right)$.

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :
Pour k de 1 à n
Faire pour tout i et j de 1 à n $l_{ij} \leftarrow min\left(l_{ij}, l_{ik} + l_{kj}\right)$.



Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :
Pour k de 1 à n

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :
Pour k de 1 à n
Faire pour tout i et j de 1 à n $l_{ij} \leftarrow min\left(l_{ij}, l_{ik} + l_{kj}\right)$.

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :
Pour k de 1 à n
Faire pour tout i et j de 1 à n $l_{ij} \leftarrow min\left(l_{ij}, l_{ik} + l_{kj}\right)$.
Remarque :

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :
Pour k de 1 à n
Faire pour tout i et j de 1 à n $l_{ij} \leftarrow min(l_{ij}, l_{ik} + l_{kj})$.
Remarque :

1. pour $i = k$ $l_{kk} = 0$ et $l_{kj} \leftarrow min(l_{kj}, l_{kk} + l_{kj}) = l_{kj}$

Algorithme matriciel de Floyd

Algorithme matriciel de Floyd

L'algorithme de Floyd est :
Pour k de 1 à n
Faire pour tout i et j de 1 à n $l_{ij} \leftarrow min\,(l_{ij}, l_{ik} + l_{kj})$.
Remarque :

1. pour $i = k$ $l_{kk} = 0$ et $l_{kj} \leftarrow min\,(l_{kj}, l_{kk} + l_{kj}) = l_{kj}$
2. pour $j = k$ $l_{kk} = 0$ et $l_{ik} \leftarrow min\,(l_{ik}, l_{ik} + l_{kk}) = l_{ik}$

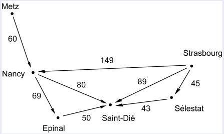
Algorithme de Floyd

Exemple

Algorithme de Floyd

Exemple

$$L^{(0)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ \infty & 0 & 60 & \infty & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ \infty & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$



Algorithme de Floyd

Etape 1

Algorithme de Floyd

Etape 1

$k = 1$ (Epinal) $l_{ij} = min\left(l_{ij}, l_{i1} + l_{1j}\right)$

Algorithme de Floyd

Etape 1

$k = 1$ (Epinal) $l_{ij} = min\left(l_{ij}, l_{i1} + l_{1j}\right)$

$$L^{(0)} = \begin{array}{c|c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ \infty & 0 & 60 & \infty & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ \infty & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array} = L^{(1)}$$

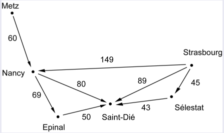
Algorithme de Floyd

Etape 1

$k = 1$ (Epinal) $l_{ij} = min\left(l_{ij}, l_{i1} + l_{1j}\right)$

$$L^{(0)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ \infty & 0 & 60 & \infty & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ \infty & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array} = L^{(1)}$$

$l_{N\ SD} = min(80, 69 + 50) = 80$



Algorithme de Floyd

Etape 2

Algorithme de Floyd

Etape 2

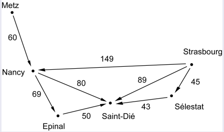
$k = 2$ (Metz) $l_{ij} = min\left(l_{ij}, l_{i2} + l_{2j}\right)$

Algorithme de Floyd

Etape 2

$k = 2$ (Metz) $l_{ij} = min\,(l_{ij}, l_{i2} + l_{2j})$

$$L^{(1)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ \infty & 0 & 60 & \infty & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ \infty & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array} = L^{(2)}$$



Algorithme de Floyd

Etape 3

Algorithme de Floyd

Etape 3

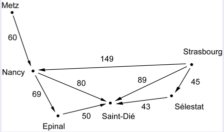
$k = 3$ (Nancy) $l_{ij} = min\,(l_{ij}, l_{i3} + l_{3j})$

Algorithme de Floyd

Etape 3

$k = 3$ (Nancy) $l_{ij} = min\left(l_{ij}, l_{i3} + l_{3j}\right)$

$$L^{(2)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ \infty & 0 & 60 & \infty & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ \infty & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$



Algorithme de Floyd

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Etape 3

Algorithme de Floyd

Etape 3

$k = 3$ (Nancy) $l_{ij} = min\left(l_{ij}, l_{i3} + l_{3j}\right)$

Algorithme de Floyd

Etape 3

$k = 3$ (Nancy) $l_{ij} = min(l_{ij}, l_{i3} + l_{3j})$

$$L^{(3)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$

Algorithme de Floyd

Etape 3

$k = 3$ (Nancy) $l_{ij} = min\,(l_{ij}, l_{i3} + l_{3j})$

$$L^{(3)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$

$l_{M\ E} = min(\infty, 60 + 69) = 129$

Algorithme de Floyd

Etape 3

$k = 3$ (Nancy) $l_{ij} = min\,(l_{ij}, l_{i3} + l_{3j})$

$$L^{(3)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$

$l_{M\ E} = min(\infty, 60 + 69) = 129$ $\qquad$ $l_{M\ SD} = min(\infty, 60 + 80) = 140$

Algorithme de Floyd

Etape 3

$k = 3$ (Nancy) $l_{ij} = min\left(l_{ij}, l_{i3} + l_{3j}\right)$

$$L^{(3)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$

$l_{M\ E} = min(\infty, 60 + 69) = 129$ $\quad l_{M\ SD} = min(\infty, 60 + 80) = 140$

$l_{Stg\ E} = min(\infty, 149 + 69) = 218$

Algorithme de Floyd

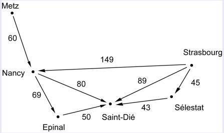
Etape 3

$k = 3$ (Nancy) $l_{ij} = min\left(l_{ij}, l_{i3} + l_{3j}\right)$

$$L^{(3)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$

$l_{M\ E} = min(\infty, 60 + 69) = 129$ $\quad l_{M\ SD} = min(\infty, 60 + 80) = 140$

$l_{Stg\ E} = min(\infty, 149 + 69) = 218$ $\quad l_{Stg\ SD} = min(89, 149 + 80) = 89$



Algorithme de Floyd

Etape 4

Algorithme de Floyd

Etape 4

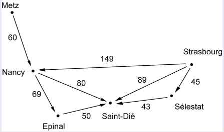
$k = 4$ (Saint-Dié) $l_{ij} = min\left(l_{ij}, l_{i4} + l_{4j}\right)$

Algorithme de Floyd

Etape 4

$k = 4$ (Saint-Dié) $l_{ij} = min\,(l_{ij}, l_{i4} + l_{4j})$

$$L^{(3)} = \begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array} = L^{(4)}$$



Algorithme de Floyd

Etape 5

Algorithme de Floyd

Etape 5

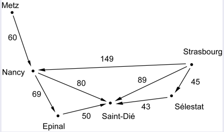
$k = 5$ (Sélestat) $l_{ij} = min\left(l_{ij}, l_{i5} + l_{5j}\right)$

Algorithme de Floyd

Etape 5

$k = 5$ (Sélestat) $l_{ij} = min\left(l_{ij}, l_{i5} + l_{5j}\right)$

$$L^{(4)} = \begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 89 & 45 & 0 \end{pmatrix} \end{array}$$



Algorithme de Floyd

Etape 5

Algorithme de Floyd

Etape 5

$k = 5$ (Sélestat) $l_{ij} = min\left(l_{ij}, l_{i5} + l_{5j}\right)$

Algorithme de Floyd

Etape 5

$k = 5$ (Sélestat) $l_{ij} = min\,(l_{ij}, l_{i5} + l_{5j})$

$$L^{(5)} = \begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 88 & 45 & 0 \end{pmatrix} \end{array}$$

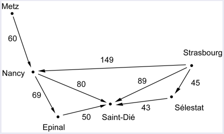
Algorithme de Floyd

Etape 5

$k = 5$ (Sélestat) $l_{ij} = min\,(l_{ij}, l_{i5} + l_{5j})$

$$L^{(5)} = \begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 88 & 45 & 0 \end{pmatrix} \end{array}$$

$l_{Stg\ SD} = min(89, 45 + 43) = 88$



Algorithme de Floyd

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié

Etape 6

Algorithme de Floyd

Etape 6

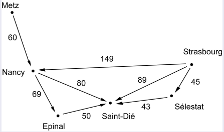
$k = 6$ (Strasbourg) $l_{ij} = min\left(l_{ij}, l_{i6} + l_{6j}\right)$

Algorithme de Floyd

Etape 6

$k = 6$ (Strasbourg) $l_{ij} = min\left(l_{ij}, l_{i6} + l_{6j}\right)$

$$L^{(5)} = \begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \left(\begin{array}{cccccc} 0 & \infty & \infty & 50 & \infty & \infty \\ 129 & 0 & 60 & 140 & \infty & \infty \\ 69 & \infty & 0 & 80 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 43 & 0 & \infty \\ 218 & \infty & 149 & 88 & 45 & 0 \end{array}\right) \end{array} = L^{(6)}$$



Algorithmes de Moore-Dijkstra et de Floyd

Remarque : algorithme de Floyd

Algorithmes de Moore-Dijkstra et de Floyd

Remarque : algorithme de Floyd

L'algorithme de Floyd peut s'appliquer avec des $l_{ij} < 0$.

Algorithmes de Moore-Dijkstra et de Floyd

Remarque : algorithme de Floyd

L'algorithme de Floyd peut s'appliquer avec des $l_{ij} < 0$.
Si en cours d'algorithme on a des $l_{ii} < 0$, il y a alors au moins un circuit négatif passant par i.

Remarque : algorithme de Floyd

L'algorithme de Floyd peut s'appliquer avec des $l_{ij} < 0$.
Si en cours d'algorithme on a des $l_{ii} < 0$, il y a alors au moins un circuit négatif passant par i.
Il n'y a alors pas de plus court chemin.

Remarque

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Les deux méthodes permettent de déterminer les distances $d(i,j)$ entre deux sommets, et donc les écartements $e(i)$, le diamètre, le rayon et les centres d'un graphe.

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Les deux méthodes permettent de déterminer les distances $d(i,j)$ entre deux sommets, et donc les écartements $e(i)$, le diamètre, le rayon et les centres d'un graphe.
Initiatilisation (cas orienté) :

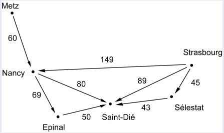
Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Les deux méthodes permettent de déterminer les distances $d(i,j)$ entre deux sommets, et donc les écartements $e(i)$, le diamètre, le rayon et les centres d'un graphe.

Initiatilisation (cas orienté) :

$$\begin{array}{cc} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 1 & \infty & \infty \\ \infty & 0 & 1 & \infty & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & 0 & \infty \\ \infty & \infty & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$



Remarque

Remarque

Résultat (cas orienté) :

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

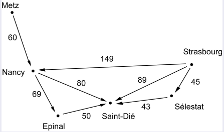
Résultat (cas orienté) :

$$\begin{array}{c} \\ \text{E} \\ \text{M} \\ \text{N} \\ \text{SD} \\ \text{Se} \\ \text{Stg} \end{array} \begin{array}{c} \begin{array}{cccccc} \text{E} & \text{M} & \text{N} & \text{SD} & \text{Se} & \text{Stg} \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 1 & \infty & \infty \\ 2 & 0 & 1 & 2 & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & 0 & \infty \\ 2 & \infty & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Remarque

Résultat (cas orienté) :

$$\begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 1 & \infty & \infty \\ 2 & 0 & 1 & 2 & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & 0 & \infty \\ 2 & \infty & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$



Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

$$\begin{array}{c c}
 & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} &
\begin{pmatrix}
0 & \infty & \infty & 1 & \infty & \infty \\
2 & 0 & 1 & 2 & \infty & \infty \\
1 & \infty & 0 & 1 & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & 1 & 0 & \infty \\
2 & \infty & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}$$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

$$\begin{array}{cc} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 1 & \infty & \infty \\ 2 & 0 & 1 & 2 & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & 0 & \infty \\ 2 & \infty & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

$$\begin{array}{c c}
 & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} &
\begin{pmatrix}
0 & \infty & \infty & 1 & \infty & \infty \\
2 & 0 & 1 & 2 & \infty & \infty \\
1 & \infty & 0 & 1 & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & 1 & 0 & \infty \\
2 & \infty & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}$$

Écartement d'un sommet : $e(i) = \max\limits_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min\limits_{i \in E} e(i)$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 1 & \infty & \infty \\ 2 & 0 & 1 & 2 & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & 0 & \infty \\ 2 & \infty & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max\limits_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min\limits_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

$$\begin{array}{c} \\ E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} \begin{array}{c} \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{pmatrix} 0 & \infty & \infty & 1 & \infty & \infty \\ 2 & 0 & 1 & 2 & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & 0 & \infty \\ 2 & \infty & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max\limits_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min\limits_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Diamètre d'un graphe : $\delta(\mathcal{G}) = \max\limits_{i \in E} e(i)$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & \infty & \infty & 1 & \infty & \infty \\ 2 & 0 & 1 & 2 & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 1 & 0 & \infty \\ 2 & \infty & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Diamètre d'un graphe : $\delta(\mathcal{G}) = \max_{i \in E} e(i)$

$\rho(\mathcal{G}) = \delta(\mathcal{G}) = \infty$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas orienté) :

$$
\begin{array}{c c}
 & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} &
\begin{pmatrix}
0 & \infty & \infty & 1 & \infty & \infty \\
2 & 0 & 1 & 2 & \infty & \infty \\
1 & \infty & 0 & 1 & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & 1 & 0 & \infty \\
2 & \infty & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$

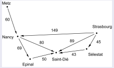
Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Diamètre d'un graphe : $\delta(\mathcal{G}) = \max_{i \in E} e(i)$

$\rho(\mathcal{G}) = \delta(\mathcal{G}) = \infty$ Tous les sommets sont un centre.



Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

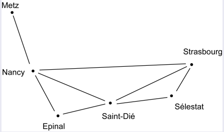
Initiatilisation (cas non orienté) :

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Initiatilisation (cas non orienté) :

$$
\begin{array}{c c}
 & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} &
\begin{pmatrix}
0 & \infty & 1 & 1 & \infty & \infty \\
\infty & 0 & 1 & \infty & \infty & \infty \\
1 & 1 & 0 & 1 & \infty & 1 \\
1 & \infty & 1 & 0 & 1 & 1 \\
\infty & \infty & \infty & 1 & 0 & 1 \\
\infty & \infty & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$



Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Algorithmes de Moore-Dijkstra et de Floyd

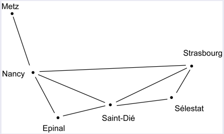
Remarque

Résultat (cas non orienté) :

Remarque

Résultat (cas non orienté) :

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 2 & 1 & 2 & 2 & 2 \\ 2 & 0 & 1 & 2 & 3 & 2 \\ 1 & 1 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 1 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 2 & 2 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$



Remarque

Remarque

Résultat (cas non orienté) :

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 2 & 1 & 2 & 2 & 2 \\ 2 & 0 & 1 & 2 & 3 & 2 \\ 1 & 1 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 1 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 2 & 2 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 2 & 1 & 2 & 2 & 2 \\ 2 & 0 & 1 & 2 & 3 & 2 \\ 1 & 1 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 1 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 2 & 2 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$
\begin{array}{cc}
 & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} &
\begin{pmatrix}
0 & 2 & 1 & 2 & 2 & 2 \\
2 & 0 & 1 & 2 & 3 & 2 \\
1 & 1 & 0 & 1 & 2 & 1 \\
1 & 2 & 1 & 0 & 1 & 1 \\
2 & 3 & 2 & 1 & 0 & 1 \\
2 & 2 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$

Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 2 & 1 & 2 & 2 & 2 \\ 2 & 0 & 1 & 2 & 3 & 2 \\ 1 & 1 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 1 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 2 & 2 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$\begin{array}{c|c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 2 & 1 & 2 & 2 & 2 \\ 2 & 0 & 1 & 2 & 3 & 2 \\ 1 & 1 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 1 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 2 & 2 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Diamètre d'un graphe : $\delta(\mathcal{G}) = \max_{i \in E} e(i)$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$
\begin{array}{c c}
 & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} &
\begin{pmatrix}
0 & 2 & 1 & 2 & 2 & 2 \\
2 & 0 & 1 & 2 & 3 & 2 \\
1 & 1 & 0 & 1 & 2 & 1 \\
1 & 2 & 1 & 0 & 1 & 1 \\
2 & 3 & 2 & 1 & 0 & 1 \\
2 & 2 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$

Écartement d'un sommet : $e(i) = \max\limits_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min\limits_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Diamètre d'un graphe : $\delta(\mathcal{G}) = \max\limits_{i \in E} e(i)$

$\rho(\mathcal{G}) = 2$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$\begin{array}{c c} & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\ \begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} & \begin{pmatrix} 0 & 2 & 1 & 2 & 2 & 2 \\ 2 & 0 & 1 & 2 & 3 & 2 \\ 1 & 1 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 1 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 2 & 2 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Écartement d'un sommet : $e(i) = \max_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Diamètre d'un graphe : $\delta(\mathcal{G}) = \max_{i \in E} e(i)$

$\rho(\mathcal{G}) = 2 \ \delta(\mathcal{G}) = 3$

Algorithmes de Moore-Dijkstra et de Floyd

Remarque

Résultat (cas non orienté) :

$$
\begin{array}{cc}
 & \begin{array}{cccccc} E & M & N & SD & Se & Stg \end{array} \\
\begin{array}{c} E \\ M \\ N \\ SD \\ Se \\ Stg \end{array} &
\begin{pmatrix}
0 & 2 & 1 & 2 & 2 & 2 \\
2 & 0 & 1 & 2 & 3 & 2 \\
1 & 1 & 0 & 1 & 2 & 1 \\
1 & 2 & 1 & 0 & 1 & 1 \\
2 & 3 & 2 & 1 & 0 & 1 \\
2 & 2 & 1 & 1 & 1 & 0
\end{pmatrix}
\end{array}
$$

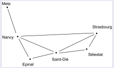
Écartement d'un sommet : $e(i) = \max\limits_{j \in E} d(i,j)$

Rayon d'un graphe : $\rho(\mathcal{G}) = \min\limits_{i \in E} e(i)$

Centre : tout sommet d'écartement minimal.

Diamètre d'un graphe : $\delta(\mathcal{G}) = \max\limits_{i \in E} e(i)$

$\rho(\mathcal{G}) = 2$ $\delta(\mathcal{G}) = 3$ Centres : Epinal, Nancy, Saint-Dié, Strasbourg.



Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Arbre recouvrant de poids minimum

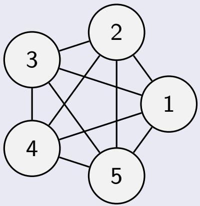
Le problème de l'arbre de poids minimum

Un arbre recouvrant d'un graphe connexe $\mathcal{G} = (E, \mathcal{A})$ est un graphe partiel $\mathcal{T} = (E, \mathcal{A}')$ de $\mathcal{G}$ qui soit un arbre.

Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

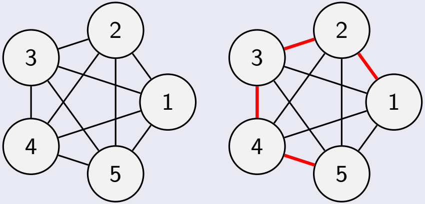
Un arbre recouvrant d'un graphe connexe $\mathcal{G} = (E, \mathcal{A})$ est un graphe partiel $\mathcal{T} = (E, \mathcal{A}')$ de $\mathcal{G}$ qui soit un arbre.
Exemple :



Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Un arbre recouvrant d'un graphe connexe $\mathcal{G} = (E, \mathcal{A})$ est un graphe partiel $\mathcal{T} = (E, \mathcal{A}')$ de $\mathcal{G}$ qui soit un arbre.
Exemple :



Arbre recouvrant de poids minimum

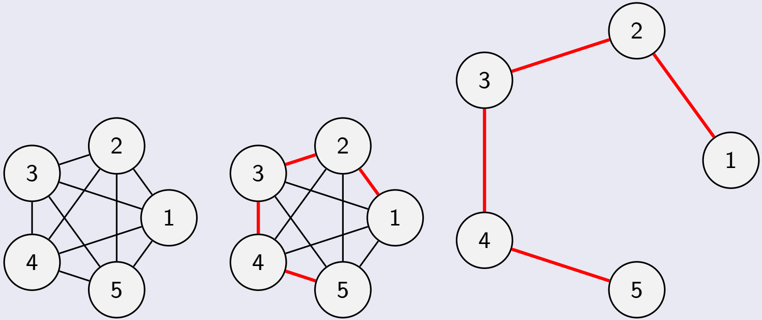
M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Le problème de l'arbre de poids minimum

Un arbre recouvrant d'un graphe connexe $\mathcal{G} = (E, \mathcal{A})$ est un graphe partiel $\mathcal{T} = (E, \mathcal{A}')$ de $\mathcal{G}$ qui soit un arbre.

Exemple :



Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
A chaque arête $\alpha \in \mathcal{A}$ on associe un nombre $w(\alpha)$ appelé le poids de l'arête.

Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
A chaque arête $\alpha \in \mathcal{A}$ on associe un nombre $w(\alpha)$ appelé le poids de l'arête.
En notant $\mathcal{G}' = (E, \mathcal{A}')$ un graphe partiel de $\mathcal{G}$, on appelle poids de $\mathcal{G}'$ le nombre $w(\mathcal{G}') = \sum_{\alpha \in \mathcal{A}'} w(\alpha)$.

Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
A chaque arête $\alpha \in \mathcal{A}$ on associe un nombre $w(\alpha)$ appelé le poids de l'arête.
En notant $\mathcal{G}' = (E, \mathcal{A}')$ un graphe partiel de $\mathcal{G}$, on appelle poids de $\mathcal{G}'$ le nombre $w(\mathcal{G}') = \sum_{\alpha \in \mathcal{A}'} w(\alpha)$.
On recherche un arbre $\mathcal{T}^*$ de $\mathcal{G}$ tel que $w(\mathcal{T}^*) = \min_{\mathcal{T}} (w(\mathcal{T}))$ minimum pris sur l'ensemble de tous les arbres possibles.

Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
A chaque arête $\alpha \in \mathcal{A}$ on associe un nombre $w(\alpha)$ appelé le poids de l'arête.
En notant $\mathcal{G}' = (E, \mathcal{A}')$ un graphe partiel de $\mathcal{G}$, on appelle poids de $\mathcal{G}'$ le nombre $w(\mathcal{G}') = \sum_{\alpha \in \mathcal{A}'} w(\alpha)$.
On recherche un arbre $\mathcal{T}^*$ de $\mathcal{G}$ tel que $w(\mathcal{T}^*) = \min_{\mathcal{T}} (w(\mathcal{T}))$ minimum pris sur l'ensemble de tous les arbres possibles.
On démontre qu'un tel arbre existe : il est dit arbre recouvrant de poids minimum.

Arbre recouvrant de poids minimum

Le problème de l'arbre de poids minimum

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
A chaque arête $\alpha \in \mathcal{A}$ on associe un nombre $w(\alpha)$ appelé le poids de l'arête.
En notant $\mathcal{G}' = (E, \mathcal{A}')$ un graphe partiel de $\mathcal{G}$, on appelle poids de $\mathcal{G}'$ le nombre $w(\mathcal{G}') = \sum\limits_{\alpha \in \mathcal{A}'} w(\alpha)$.
On recherche un arbre $\mathcal{T}^*$ de $\mathcal{G}$ tel que $w(\mathcal{T}^*) = \min\limits_{\mathcal{T}} (w(\mathcal{T}))$ minimum pris sur l'ensemble de tous les arbres possibles.
On démontre qu'un tel arbre existe : il est dit arbre recouvrant de poids minimum.
Applications : simplification de câblages, de dessertes (transports)...

Arbre recouvrant de poids minimum

M2201
Graphes et langages

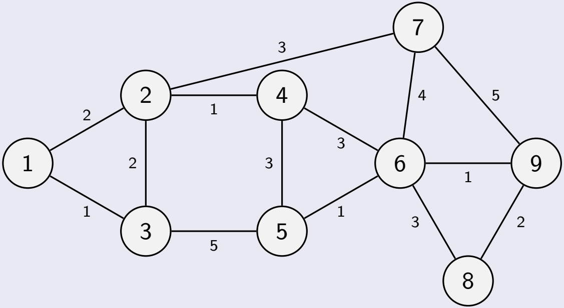
Département Informatique IUT de Saint-Dié

Algorithme de Kruskal : exemple

Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple

On veut équiper un pays en accès internet à haut débit. Pour cela, on doit relier les 9 plus grandes villes avec des câbles de fibres optiques. Après une étude préliminaire, une estimation des coûts de connexion entre les villes est donnée par le graphe suivant :



Arbre recouvrant de poids minimum

Algorithme de Kruskal (1956)

Arbre recouvrant de poids minimum

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

Arbre recouvrant de poids minimum

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
Donnée : la liste des arêtes dans l'ordre des poids croissants.

Arbre recouvrant de poids minimum

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
Donnée : la liste des arêtes dans l'ordre des poids croissants.

1. Initialisation : $T = \{\alpha_1\}$

Arbre recouvrant de poids minimum

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
Donnée : la liste des arêtes dans l'ordre des poids croissants.

1. Initialisation : $T = \{\alpha_1\}$
2. Lecture de la liste des arêtes

Arbre recouvrant de poids minimum

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
Donnée : la liste des arêtes dans l'ordre des poids croissants.

1. Initialisation : $T = \{\alpha_1\}$
2. Lecture de la liste des arêtes
   A l'étape k, si α_k ne forme pas de cycle avec les arêtes de T alors l'arête est prise : $T \leftarrow T \cup \{\alpha_k\}$.

Arbre recouvrant de poids minimum

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
Donnée : la liste des arêtes dans l'ordre des poids croissants.

1. Initialisation : $T = \{\alpha_1\}$
2. Lecture de la liste des arêtes
   A l'étape k, si α_k ne forme pas de cycle avec les arêtes de T alors l'arête est prise : $T \leftarrow T \cup \{\alpha_k\}$.
   Passage à l'arête suivante.

Arbre recouvrant de poids minimum

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
Donnée : la liste des arêtes dans l'ordre des poids croissants.

1. Initialisation : $T = \{\alpha_1\}$
2. Lecture de la liste des arêtes
   A l'étape k, si α_k ne forme pas de cycle avec les arêtes de T alors l'arête est prise : $T \leftarrow T \cup \{\alpha_k\}$.
   Passage à l'arête suivante.

L'arbre $\mathcal{T} = (E, T)$ obtenu est de poids minimum.

Arbre recouvrant de poids minimum

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Algorithme de Kruskal (1956)

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.
Donnée : la liste des arêtes dans l'ordre des poids croissants.

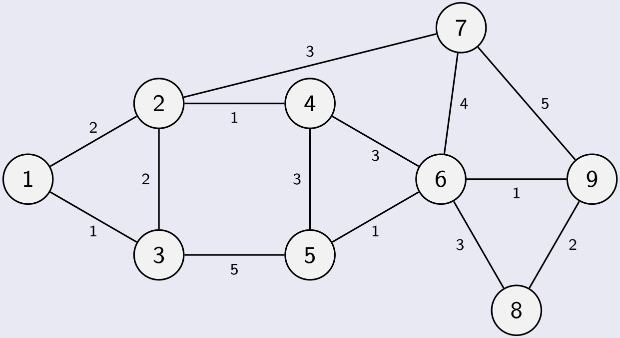
1. Initialisation : $T = \{\alpha_1\}$
2. Lecture de la liste des arêtes
   A l'étape k, si α_k ne forme pas de cycle avec les arêtes de T alors l'arête est prise : $T \leftarrow T \cup \{\alpha_k\}$.
   Passage à l'arête suivante.

L'arbre $\mathcal{T} = (E, T)$ obtenu est de poids minimum.

Joseph Kruskal : mathématicien américain (1928-2010).

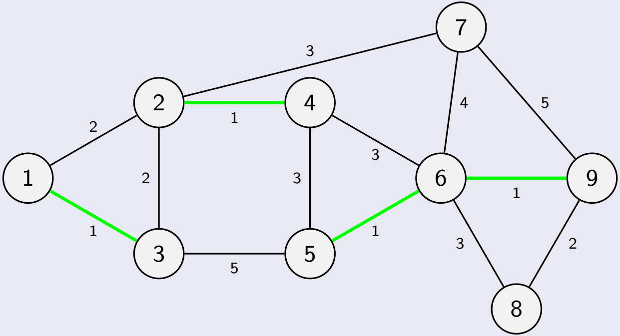
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple

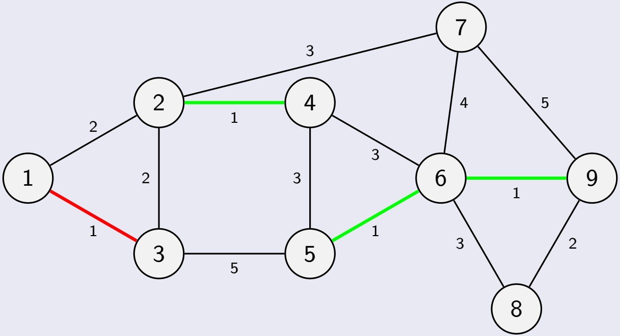


Arbre recouvrant de poids minimum

M2201
Graphes et langages

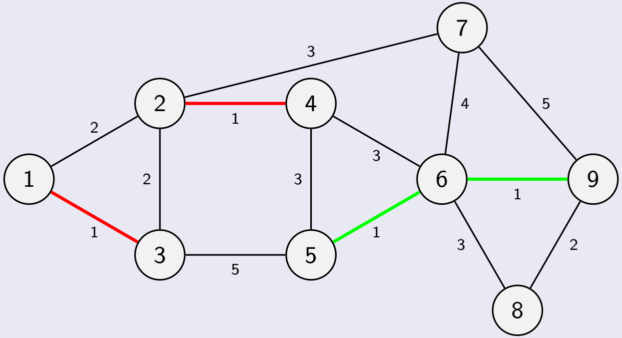
Département Informatique
IUT de Saint-Dié

Algorithme de Kruskal : exemple



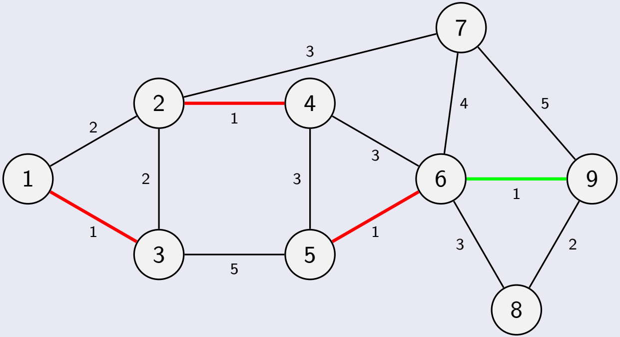
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



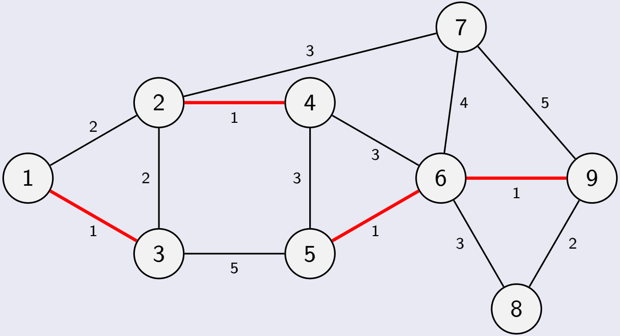
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple

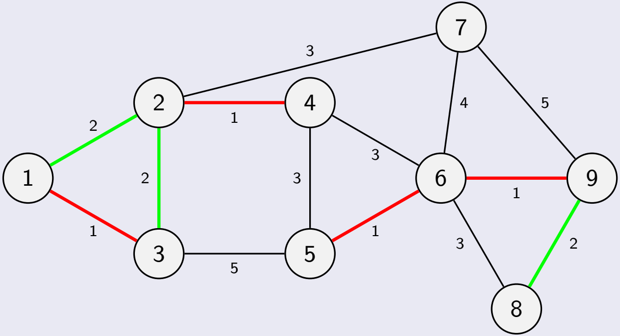


Arbre recouvrant de poids minimum

M2201
Graphes et langages

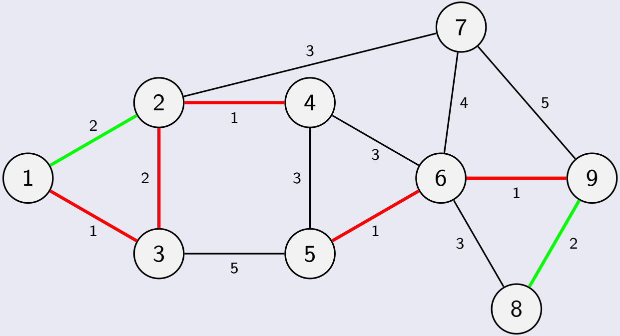
Département Informatique IUT de Saint-Dié

Algorithme de Kruskal : exemple



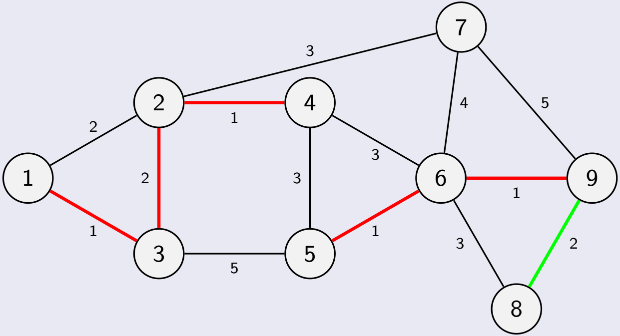
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



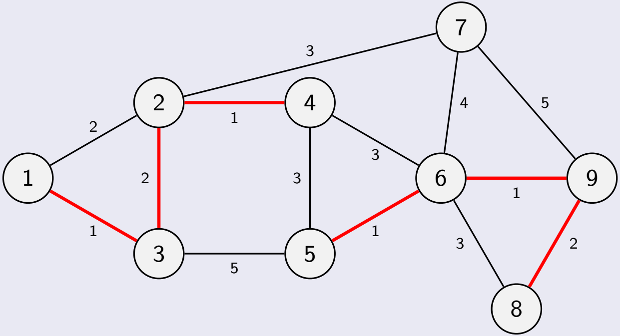
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



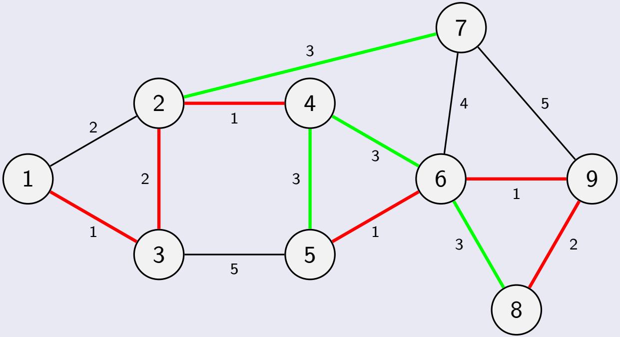
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



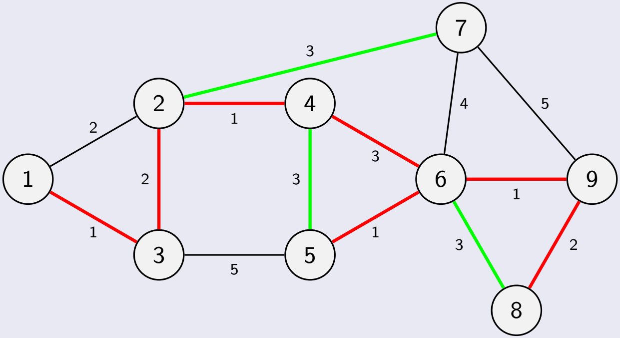
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple

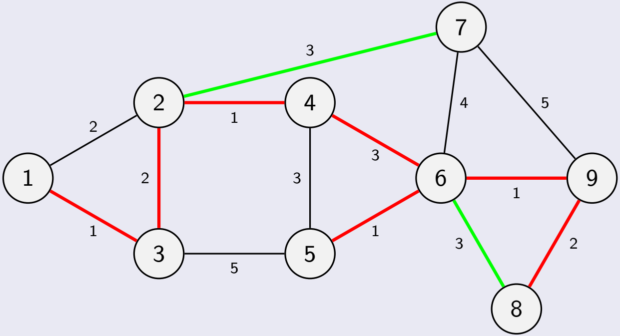


Arbre recouvrant de poids minimum

M2201
Graphes et langages

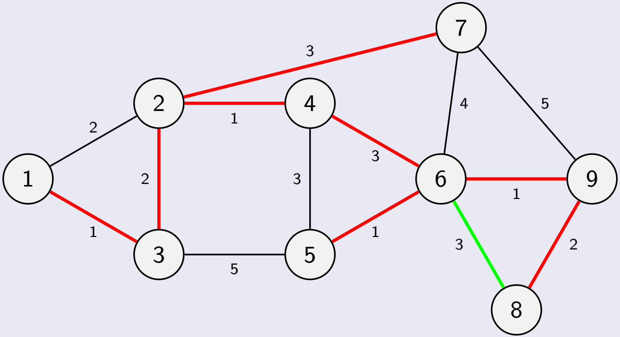
Département Informatique IUT de Saint-Dié

Algorithme de Kruskal : exemple



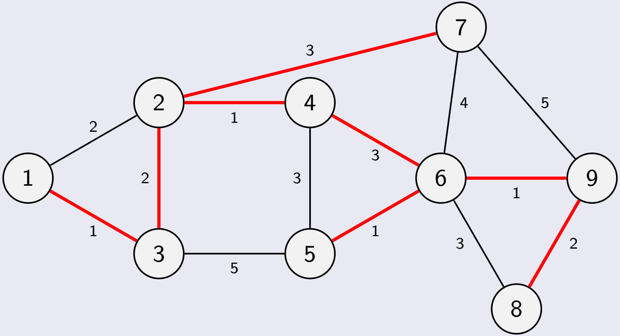
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



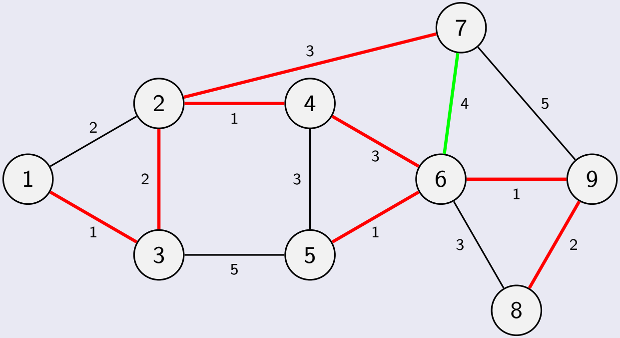
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple

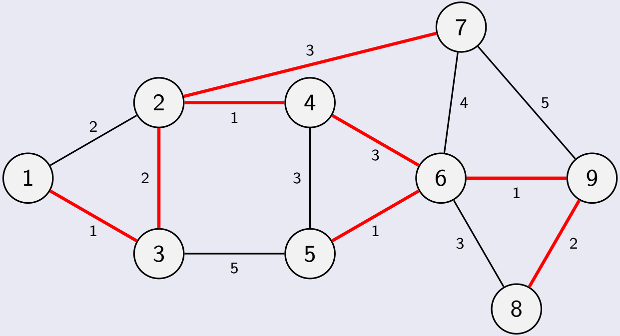


M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

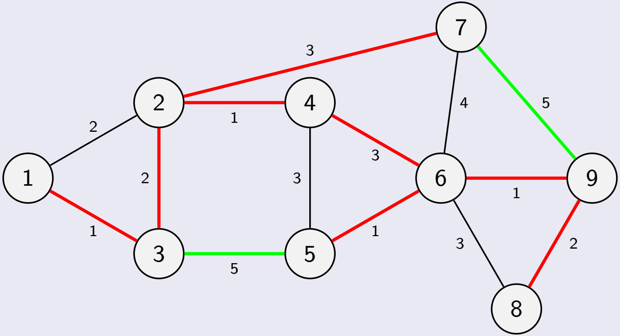
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



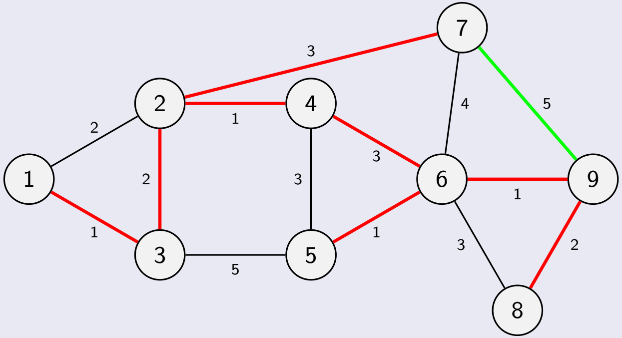
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



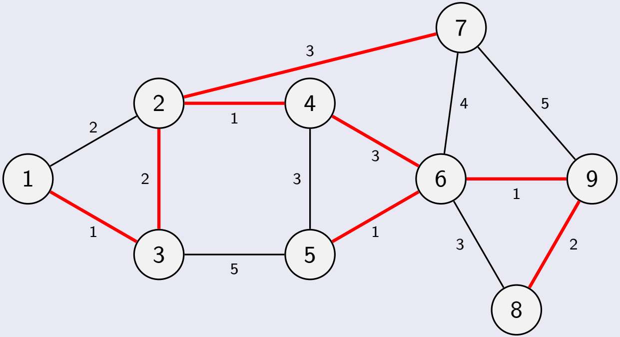
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



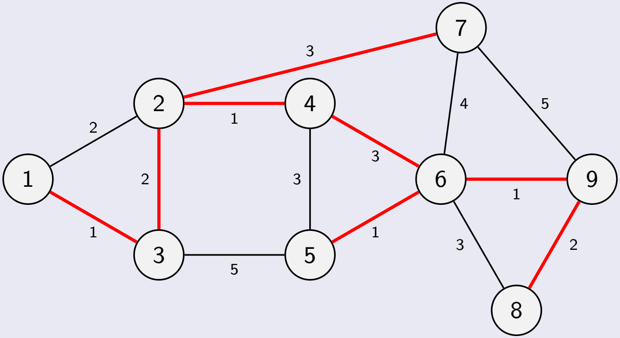
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



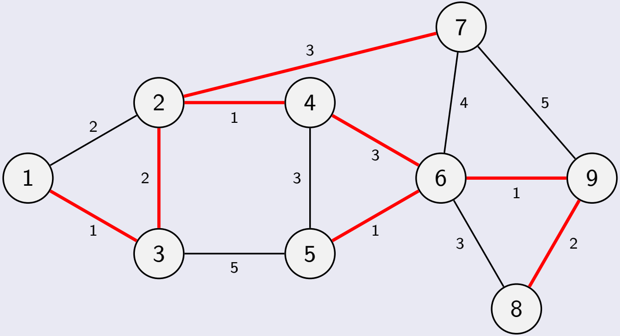
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



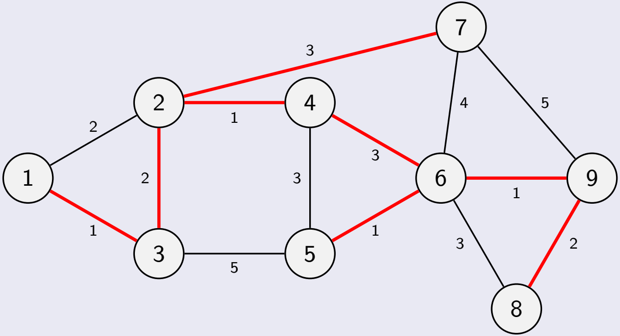
Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



Arbre recouvrant de poids minimum

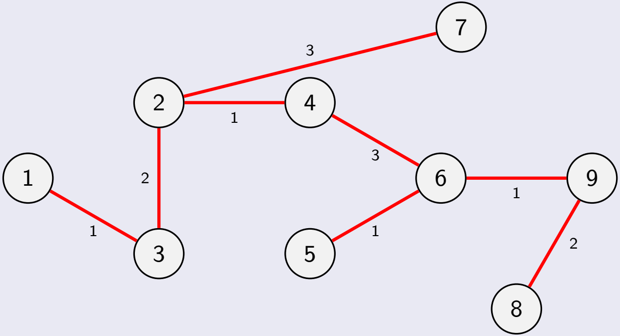
Algorithme de Kruskal : exemple



L'arbre $\mathcal{T} = (E, T)$ obtenu est un arbre recouvrant de poids minimum de valeur $1 + 1 + 1 + 1 + 2 + 2 + 3 + 3 = 14$.

Arbre recouvrant de poids minimum

Algorithme de Kruskal : exemple



Arbre recouvrant de poids minimum

Algorithme de Prim

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)

Arbre recouvrant de poids minimum

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter
    1. Trouver toutes les arêtes de $\mathcal{A}$ qui relient un sommet de S et un sommet de $\overline{S}$.

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter
   1. Trouver toutes les arêtes de $\mathcal{A}$ qui relient un sommet de S et un sommet de $\overline{S}$.
   2. Parmi celles-ci, choisir une arête de poids le plus petit possible.

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter
   1. Trouver toutes les arêtes de $\mathcal{A}$ qui relient un sommet de S et un sommet de $\overline{S}$.
   2. Parmi celles-ci, choisir une arête de poids le plus petit possible.
   3. Ajouter à T cette arête et le sommet correspondant à S.

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter
   1. Trouver toutes les arêtes de $\mathcal{A}$ qui relient un sommet de S et un sommet de $\overline{S}$.
   2. Parmi celles-ci, choisir une arête de poids le plus petit possible.
   3. Ajouter à T cette arête et le sommet correspondant à S.
3. S'arrêter dès que tous les sommets de E sont dans S.

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter
    1. Trouver toutes les arêtes de $\mathcal{A}$ qui relient un sommet de S et un sommet de $\overline{S}$.
    2. Parmi celles-ci, choisir une arête de poids le plus petit possible.
    3. Ajouter à T cette arête et le sommet correspondant à S.
3. S'arrêter dès que tous les sommets de E sont dans S.
4. Retourner T.

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter
   1. Trouver toutes les arêtes de $\mathcal{A}$ qui relient un sommet de S et un sommet de $\overline{S}$.
   2. Parmi celles-ci, choisir une arête de poids le plus petit possible.
   3. Ajouter à T cette arête et le sommet correspondant à S.
3. S'arrêter dès que tous les sommets de E sont dans S.
4. Retourner T.

L'arbre $\mathcal{T} = (E, T)$ obtenu est de poids minimum.

Arbre recouvrant de poids minimum

Algorithme de Prim

Soit $\mathcal{G} = (E, \mathcal{A})$ un graphe connexe.

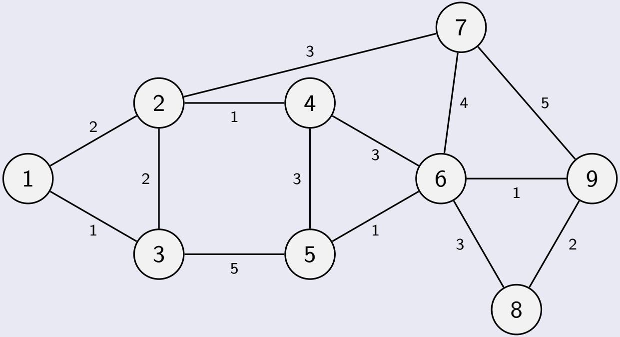
1. Initialisation : $T = \emptyset$ et $S = \{i_1\}$ (un sommet)
2. Répéter
   1. Trouver toutes les arêtes de $\mathcal{A}$ qui relient un sommet de S et un sommet de $\overline{S}$.
   2. Parmi celles-ci, choisir une arête de poids le plus petit possible.
   3. Ajouter à T cette arête et le sommet correspondant à S.
3. S'arrêter dès que tous les sommets de E sont dans S.
4. Retourner T.

L'arbre $\mathcal{T} = (E, T)$ obtenu est de poids minimum.

Algorithme développé en 1930 par le mathématicien tchèque Vojtech Jarnik puis redécouvert et republié par Robert C. Prim (mathématicien américain né en 1921) et Edsger W. Dijkstra en 1959.

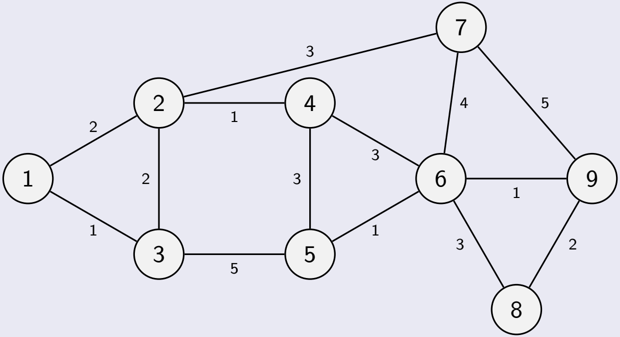
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



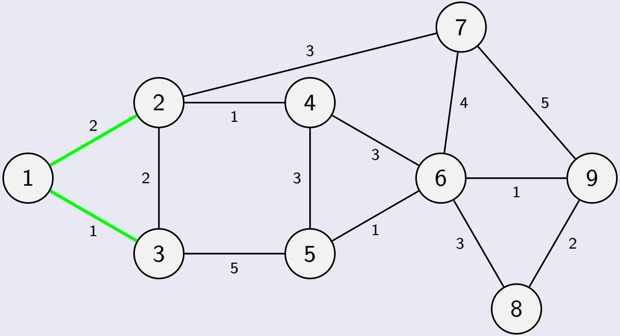
Sommet de départ : 1

Arbre recouvrant de poids minimum

M2201
Graphes et langages

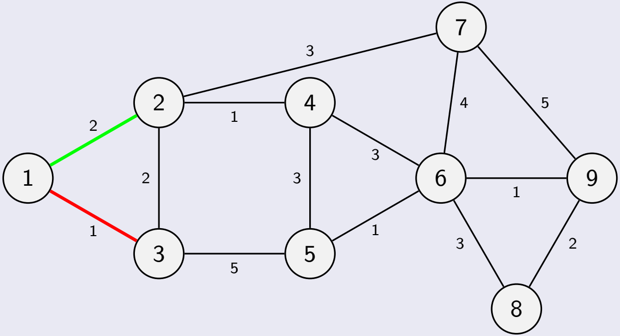
Département Informatique
IUT de Saint-Dié

Algorithme de Prim : exemple



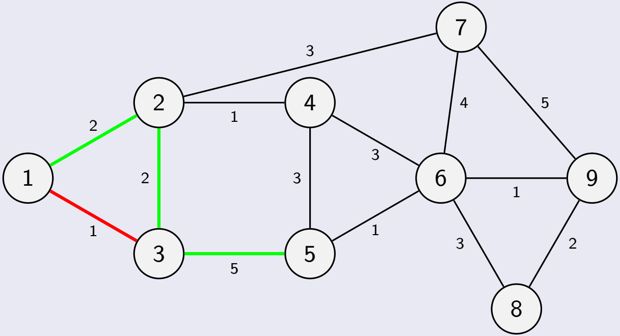
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



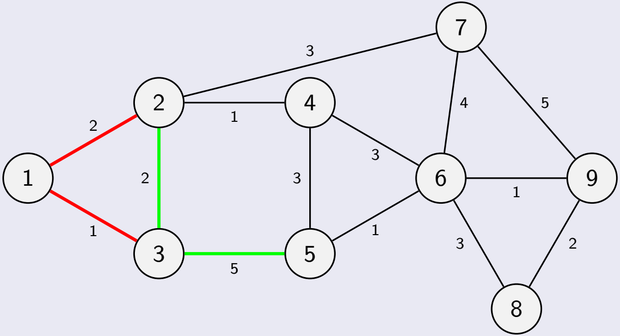
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



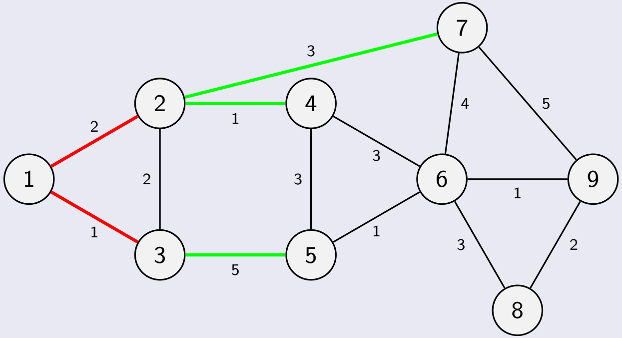
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



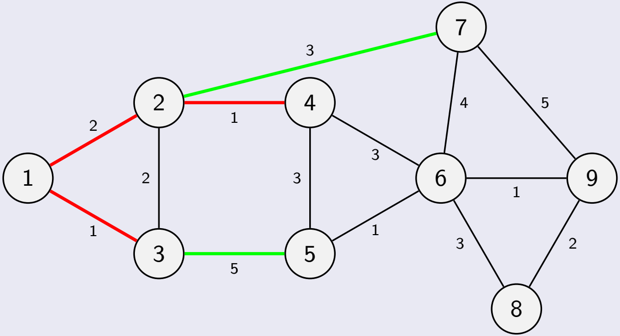
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



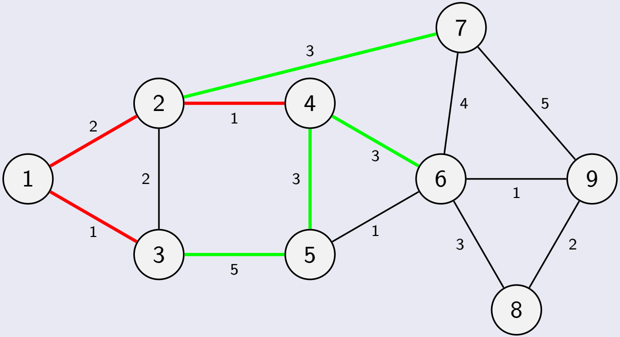
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



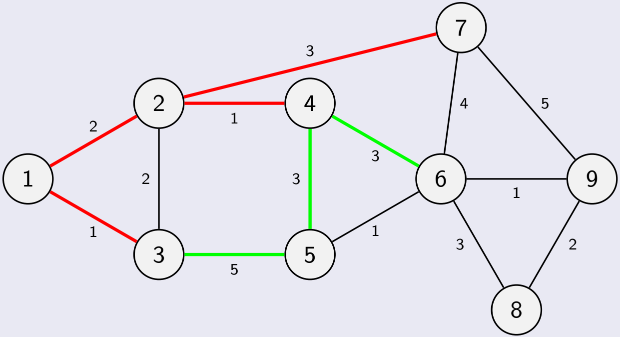
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



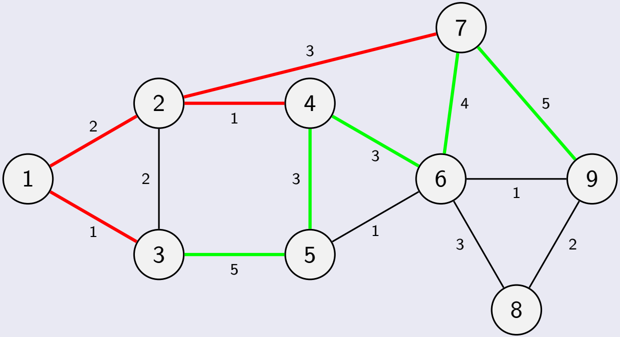
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



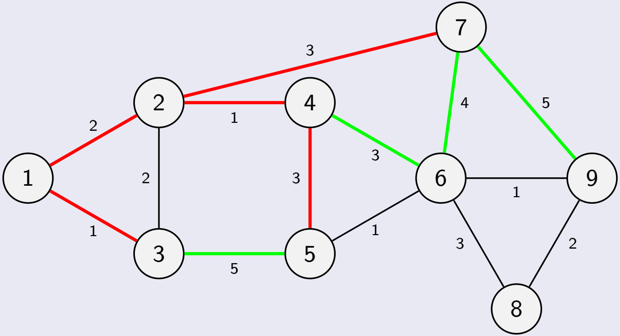
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



Arbre recouvrant de poids minimum

M2201 Graphes et langages

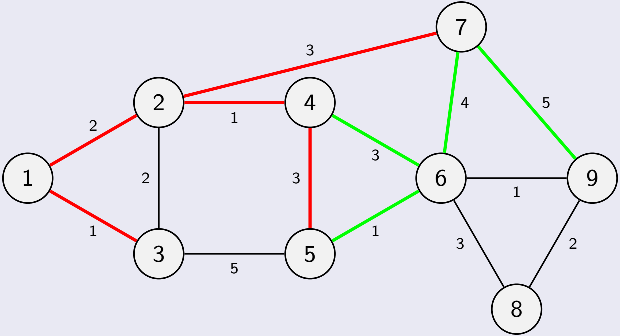
Département Informatique IUT de Saint-Dié

Graphes
- Généralités
- Arbre
- Cycle eulérien, hamiltonien
- Coloration
- Le problème du plus court chemin
- Arbre recouvrant de poids minimal

Langages et automates
- Langages
- Automates

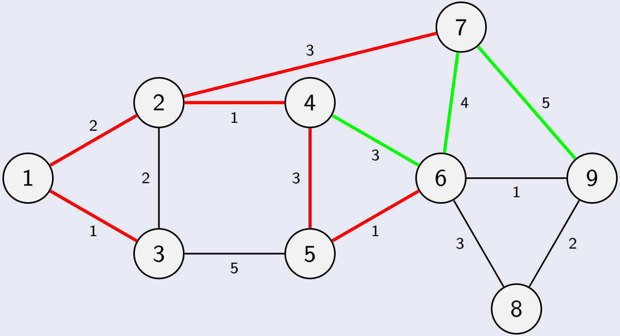
Annexes

Algorithme de Prim : exemple



Arbre recouvrant de poids minimum

Algorithme de Prim : exemple

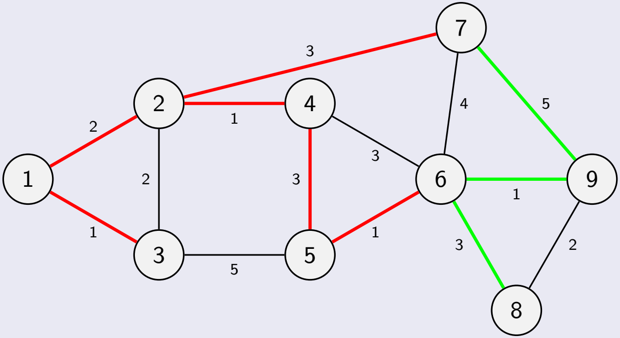


Arbre recouvrant de poids minimum

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Algorithme de Prim : exemple

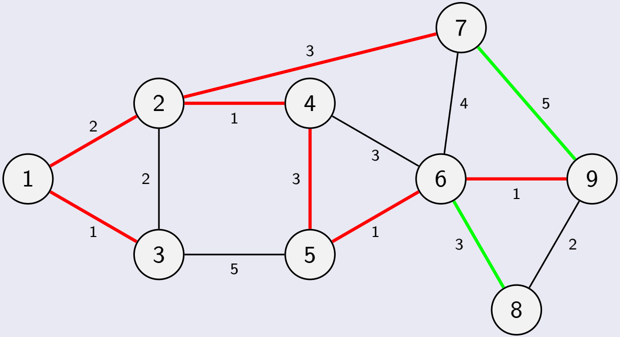


Arbre recouvrant de poids minimum

M2201
Graphes et langages

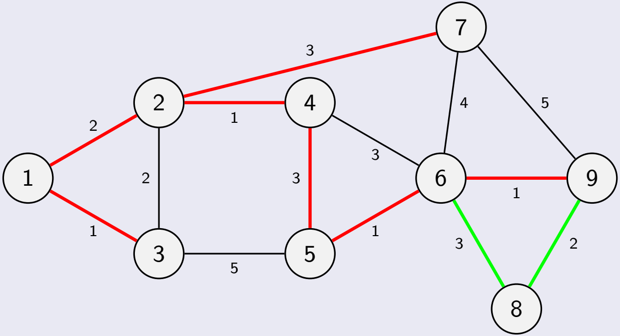
Département Informatique IUT de Saint-Dié

Algorithme de Prim : exemple



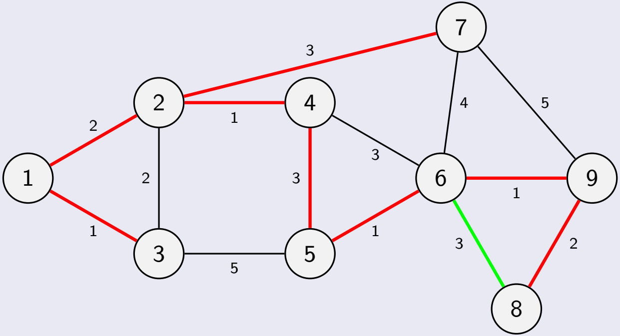
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



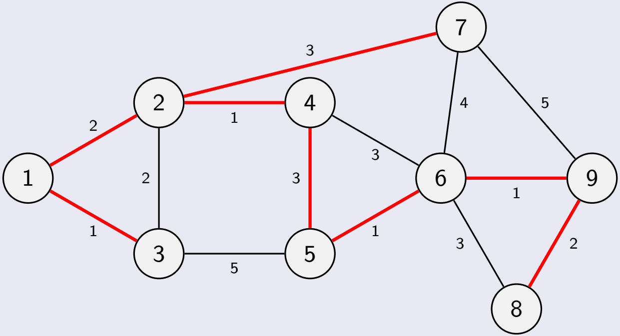
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



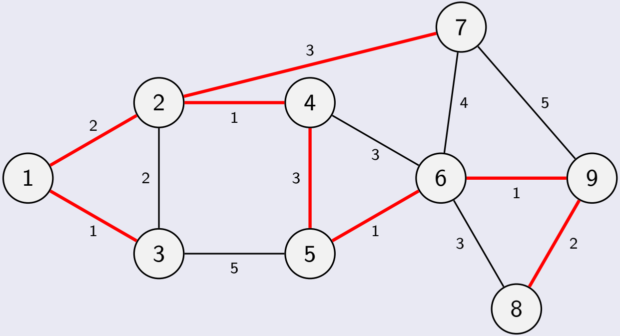
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



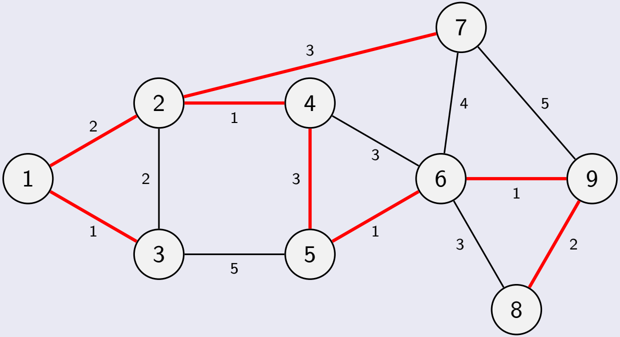
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



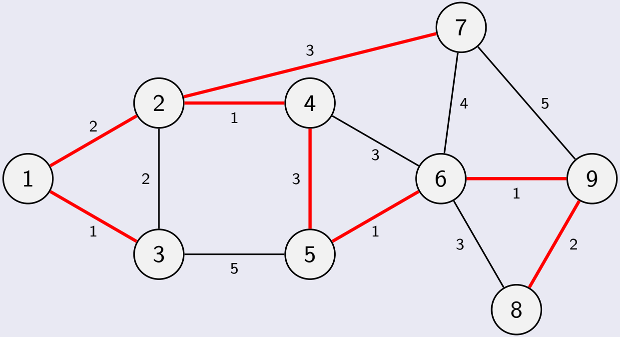
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



Arbre recouvrant de poids minimum

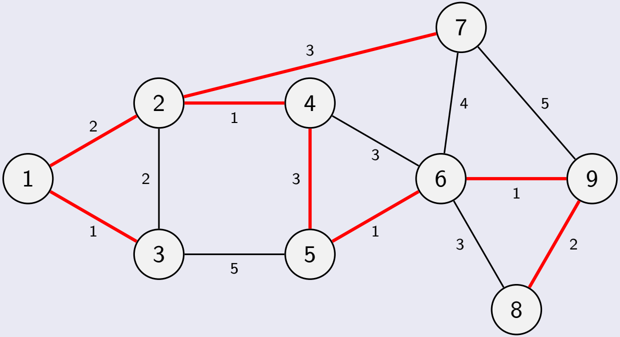
Algorithme de Prim : exemple



L'arbre $\mathcal{T} = (E, T)$ obtenu est un arbre recouvrant de poids minimum de valeur $1+2+3+1+3+1+1+2=14$.

Arbre recouvrant de poids minimum

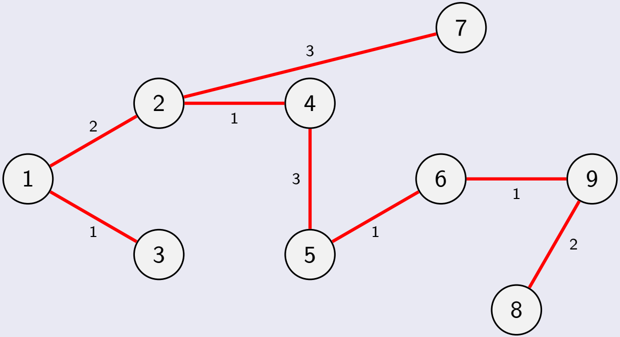
Algorithme de Prim : exemple



L'arbre $\mathcal{T} = (E, T)$ obtenu est un arbre recouvrant de poids minimum de valeur 1+2+3+1+3+1+1+2=14.
L'arbre recouvant est de même poids que celui obtenu par l'algorithme de Kruskal, mais différent.

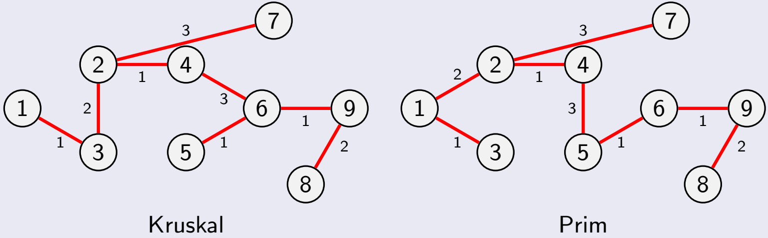
Arbre recouvrant de poids minimum

Algorithme de Prim : exemple



Arbre recouvrant de poids minimum

Algorithmes de Kruskal et de Prim : exemple



Langages

Alphabet

Langages

Alphabet

Un ensemble $\mathcal{A}$ fini non vide de caractères ou de symboles est appelé un alphabet.

Langages

Alphabet

Un ensemble $\mathcal{A}$ fini non vide de caractères ou de symboles est appelé un alphabet.
Ses éléments sont appelés lettres.

Langages

Alphabet

Un ensemble $\mathcal{A}$ fini non vide de caractères ou de symboles est appelé un alphabet.
Ses éléments sont appelés lettres.

Exemples

Langages

Alphabet

Un ensemble $\mathcal{A}$ fini non vide de caractères ou de symboles est appelé un alphabet.
Ses éléments sont appelés lettres.

Exemples

1. $\mathcal{A} = \{a\}$,

Langages

Alphabet

Un ensemble $\mathcal{A}$ fini non vide de caractères ou de symboles est appelé un alphabet.
Ses éléments sont appelés lettres.

Exemples

1. $\mathcal{A} = \{a\}$,
2. $\mathcal{A} = \{a, b\}$,

Langages

Alphabet

Un ensemble $\mathcal{A}$ fini non vide de caractères ou de symboles est appelé un alphabet.
Ses éléments sont appelés lettres.

Exemples

1. $\mathcal{A} = \{a\}$,
2. $\mathcal{A} = \{a, b\}$,
3. $\mathcal{A} = \{0, 1\}$.

Langages

Mots

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

Langages

Mots

Une suite de n lettres $(n \geq 1)$ est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

Langages

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$:

Langages

Mots

Une suite de n lettres $(n \geq 1)$ est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: a,

Langages

Mots

Une suite de n lettres $(n \geq 1)$ est un **mot** m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: a, aa

Langages

Mots

Une suite de n lettres $(n \geq 1)$ est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: a, aa
   $|a| = 1$,

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: a, aa
   $|a| = 1$, $|aa| = 2$

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: a, aa
   $|a| = 1$, $|aa| = 2$
2. Pour $\mathcal{A} = \{a, b\}$: a,

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: a, aa
   $|a| = 1$, $|aa| = 2$
2. Pour $\mathcal{A} = \{a, b\}$: a, b,

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: $a, \ aa$
   $|a| = 1, \ |aa| = 2$
2. Pour $\mathcal{A} = \{a, b\}$: $a, \ b, \ baba$

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: $a,\ aa$
   $|a| = 1,\ |aa| = 2$
2. Pour $\mathcal{A} = \{a, b\}$: $a,\ b,\ baba$
   $|a| = |b| = 1,$

Langages

Mots

Une suite de n lettres ($n \geq 1$) est un mot m de longueur n sur $\mathcal{A}$.

Notation

$l(m)$ ou $|m|$

Exemples

1. Pour $\mathcal{A} = \{a\}$: $a,\ aa$
   $|a| = 1,\ |aa| = 2$
2. Pour $\mathcal{A} = \{a, b\}$: $a,\ b,\ baba$
   $|a| = |b| = 1,\ |baba| = 4$

Langages

Concaténation

Langages

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.

Langages

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.
Il s'agit du plus petit ensemble tel que :

Langages

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.
Il s'agit du plus petit ensemble tel que :

1. pour toute lettre x de $\mathcal{A}$, x appartient à $\mathcal{A}^+$,

Langages

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.
Il s'agit du plus petit ensemble tel que :

1. pour toute lettre x de $\mathcal{A}$, x appartient à $\mathcal{A}^+$,
2. pour tout mot m de $\mathcal{A}^+$ et toute lettre x de $\mathcal{A}$: mx appartient à $\mathcal{A}^+$.

Langages

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.
Il s'agit du plus petit ensemble tel que :

1. pour toute lettre x de $\mathcal{A}$, x appartient à $\mathcal{A}^+$,
2. pour tout mot m de $\mathcal{A}^+$ et toute lettre x de $\mathcal{A}$: mx appartient à $\mathcal{A}^+$.

mx est la concaténation du mot m et de la lettre x, ou du mot m et du mot x.

Langages

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.
Il s'agit du plus petit ensemble tel que :

1. pour toute lettre x de $\mathcal{A}$, x appartient à $\mathcal{A}^+$,
2. pour tout mot m de $\mathcal{A}^+$ et toute lettre x de $\mathcal{A}$: mx appartient à $\mathcal{A}^+$.

mx est la concaténation du mot m et de la lettre x, ou du mot m et du mot x.

Exemple

Langages

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.
Il s'agit du plus petit ensemble tel que :

1. pour toute lettre x de $\mathcal{A}$, x appartient à $\mathcal{A}^+$,
2. pour tout mot m de $\mathcal{A}^+$ et toute lettre x de $\mathcal{A}$: mx appartient à $\mathcal{A}^+$.

mx est la concaténation du mot m et de la lettre x, ou du mot m et du mot x.

Exemple

Pour l'alphabet $\mathcal{A} = \{a, b\}$, la concaténation de ab et a

Concaténation

$\mathcal{A}^+$ est l'ensemble des mots de longueur $n \geq 1$ sur $\mathcal{A}$.
Il s'agit du plus petit ensemble tel que :

1. pour toute lettre x de $\mathcal{A}$, x appartient à $\mathcal{A}^+$,
2. pour tout mot m de $\mathcal{A}^+$ et toute lettre x de $\mathcal{A}$: mx appartient à $\mathcal{A}^+$.

mx est la concaténation du mot m et de la lettre x, ou du mot m et du mot x.

Exemple

Pour l'alphabet $\mathcal{A} = \{a, b\}$, la concaténation de ab et a est le mot aba.

Langages

Propriétés

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$,

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$, alors mm' est un mot sur $\mathcal{A}$ de longueur

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$, alors mm' est un mot sur $\mathcal{A}$ de longueur $|mm'| = |m| + |m'|$

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$, alors mm' est un mot sur $\mathcal{A}$ de longueur $|mm'| = |m| + |m'|$
2. $\forall m, m', m'' \in \mathcal{A}^+ \ (mm')m'' = m(m'm'')$

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$, alors mm' est un mot sur $\mathcal{A}$ de longueur $|mm'| = |m| + |m'|$
2. $\forall m, m', m'' \in \mathcal{A}^+ \; (mm')m'' = m(m'm'')$

Exemple

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$, alors mm' est un mot sur $\mathcal{A}$ de longueur $|mm'| = |m| + |m'|$
2. $\forall m, m', m'' \in \mathcal{A}^+ \ (mm')m'' = m(m'm'')$

Exemple

Pour l'alphabet $\mathcal{A} = \{a, b\}$, la concaténation de *aba* et *ba* est le mot

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$, alors mm' est un mot sur $\mathcal{A}$ de longueur $|mm'| = |m| + |m'|$
2. $\forall m, m', m'' \in \mathcal{A}^+ \ (mm')m'' = m(m'm'')$

Exemple

Pour l'alphabet $\mathcal{A} = \{a, b\}$, la concaténation de *aba* et *ba* est le mot *ababa*.

Langages

Propriétés

1. si m et m' sont deux mots sur $\mathcal{A}$ de longueur $|m|$ et $|m'|$, alors mm' est un mot sur $\mathcal{A}$ de longueur $|mm'| = |m| + |m'|$
2. $\forall m, m', m'' \in \mathcal{A}^+ \ (mm')m" = m(m'm")$

Exemple

Pour l'alphabet $\mathcal{A} = \{a, b\}$, la concaténation de *aba* et *ba* est le mot *ababa*.
$|ababa| = |aba| + |ba| = 3 + 2 = 5$.

Langages

Notation

Langages

Notation

Pour tout mot m, on note $m^1 = m$, $m^2 = mm$, ..., $m^{n+1} = mm^n$.

Langages

Notation

Pour tout mot m, on note $m^1 = m$, $m^2 = mm$, ..., $m^{n+1} = mm^n$.

Mot vide

Langages

Notation

Pour tout mot m, on note $m^1 = m$, $m^2 = mm$, ..., $m^{n+1} = mm^n$.

Mot vide

On note

Langages

Notation

Pour tout mot m, on note $m^1 = m$, $m^2 = mm$, ..., $m^{n+1} = mm^n$.

Mot vide

On note

1. ϵ le symbole représentant le mot vide.

Langages

Notation

Pour tout mot m, on note $m^1 = m$, $m^2 = mm$, ..., $m^{n+1} = mm^n$.

Mot vide

On note

1. ϵ le symbole représentant le mot vide.
2. $\mathcal{A}^* = \{\epsilon\} \cup \mathcal{A}^+$ (étoile de $\mathcal{A}$).

Langages

Propriétés

Langages

Propriétés

1. $\forall m \in \mathcal{A}^* \quad m\epsilon = \epsilon m = m$

Langages

Propriétés

1. $\forall m \in \mathcal{A}^* \quad m\epsilon = \epsilon m = m$
2. Longueur du mot vide : $|\epsilon| = 0$

Langages

Propriétés

1. $\forall m \in \mathcal{A}^* \quad m\epsilon = \epsilon m = m$
2. Longueur du mot vide : $|\epsilon| = 0$
3. $\epsilon \notin \mathcal{A}^+$.

Langages

Propriétés

1. $\forall m \in \mathcal{A}^* \quad m\epsilon = \epsilon m = m$
2. Longueur du mot vide : $|\epsilon| = 0$
3. $\epsilon \notin \mathcal{A}^+$.

Notation

Langages

Propriétés

1. $\forall m \in \mathcal{A}^* \quad m\epsilon = \epsilon m = m$
2. Longueur du mot vide : $|\epsilon| = 0$
3. $\epsilon \notin \mathcal{A}^+$.

Notation

Pour tout mot m, $m^0 = \epsilon$.

Langages

Langage

Langages

Langage

Un langage $\mathcal{L}$ sur l'alphabet $\mathcal{A}$ est un sous-ensemble de $\mathcal{A}^*$.

Langages

Langage

Un langage $\mathcal{L}$ sur l'alphabet $\mathcal{A}$ est un sous-ensemble de $\mathcal{A}^*$.

Exemples

Langages

Langage

Un langage $\mathcal{L}$ sur l'alphabet $\mathcal{A}$ est un sous-ensemble de $\mathcal{A}^*$.

Exemples

$\mathcal{L} = \{\ \}$ ou $\mathcal{L} = \{a\}$ sur l'alphabet $\mathcal{A} = \{a\}$.

Langages

Notations

Langages

Notations

Pour tout langage $\mathcal{L}$, on note

Langages

Notations

Pour tout langage $\mathcal{L}$, on note

1. $\mathcal{L}^+$ l'ensemble des mots de $\mathcal{A}^+$ obtenus par concaténation d'éléments de $\mathcal{L}$,

Langages

Notations

Pour tout langage $\mathcal{L}$, on note

1. $\mathcal{L}^+$ l'ensemble des mots de $\mathcal{A}^+$ obtenus par concaténation d'éléments de $\mathcal{L}$,
2. $\mathcal{L}^* = \{\epsilon\} \cup \mathcal{L}^+$.

Langages

Notations

Pour tout langage $\mathcal{L}$, on note

1. $\mathcal{L}^+$ l'ensemble des mots de $\mathcal{A}^+$ obtenus par concaténation d'éléments de $\mathcal{L}$,
2. $\mathcal{L}^* = \{\epsilon\} \cup \mathcal{L}^+$.

Exemple

Langages

Notations

Pour tout langage $\mathcal{L}$, on note

1. $\mathcal{L}^+$ l'ensemble des mots de $\mathcal{A}^+$ obtenus par concaténation d'éléments de $\mathcal{L}$,
2. $\mathcal{L}^* = \{\epsilon\} \cup \mathcal{L}^+$.

Exemple

Pour $\mathcal{L} = \{a\}$ sur l'alphabet $\mathcal{A} = \{a\}$,

Langages

Notations

Pour tout langage $\mathcal{L}$, on note

1. $\mathcal{L}^+$ l'ensemble des mots de $\mathcal{A}^+$ obtenus par concaténation d'éléments de $\mathcal{L}$,
2. $\mathcal{L}^* = \{\epsilon\} \cup \mathcal{L}^+$.

Exemple

Pour $\mathcal{L} = \{a\}$ sur l'alphabet $\mathcal{A} = \{a\}$,
$\mathcal{L}^+ = \{a^n,\ n \geq 1\}$

Langages

Notations

Pour tout langage $\mathcal{L}$, on note

1. $\mathcal{L}^+$ l'ensemble des mots de $\mathcal{A}^+$ obtenus par concaténation d'éléments de $\mathcal{L}$,
2. $\mathcal{L}^* = \{\epsilon\} \cup \mathcal{L}^+$.

Exemple

Pour $\mathcal{L} = \{a\}$ sur l'alphabet $\mathcal{A} = \{a\}$,
$\mathcal{L}^+ = \{a^n,\ n \geq 1\}$
et $\mathcal{L}^* = \{a^n,\ n \geq 0\}$ en posant $a^0 = \epsilon$.

Langages

Notations

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$
4. $a^+ = aa^*$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$
4. $a^+ = aa^*$

Propriétés

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$
4. $a^+ = aa^*$

Propriétés

1. $\mathcal{L}\emptyset = \emptyset\mathcal{L} = \emptyset$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$
4. $a^+ = aa^*$

Propriétés

1. $\mathcal{L}\emptyset = \emptyset\mathcal{L} = \emptyset$
2. $\mathcal{L}\epsilon = \epsilon\mathcal{L} = \mathcal{L}$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$
4. $a^+ = aa^*$

Propriétés

1. $\mathcal{L}\emptyset = \emptyset\mathcal{L} = \emptyset$
2. $\mathcal{L}\epsilon = \epsilon\mathcal{L} = \mathcal{L}$
3. $(\mathcal{L}_1\mathcal{L}_2)\mathcal{L}_3 = \mathcal{L}_1(\mathcal{L}_2\mathcal{L}_3)$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$
4. $a^+ = aa^*$

Propriétés

1. $\mathcal{L}\emptyset = \emptyset\mathcal{L} = \emptyset$
2. $\mathcal{L}\epsilon = \epsilon\mathcal{L} = \mathcal{L}$
3. $(\mathcal{L}_1\mathcal{L}_2)\mathcal{L}_3 = \mathcal{L}_1(\mathcal{L}_2\mathcal{L}_3)$
4. $\mathcal{L}(\mathcal{L}_1 \cup \mathcal{L}_2) = \mathcal{L}\mathcal{L}_1 \cup \mathcal{L}\mathcal{L}_2$

Langages

Notations

1. $\mathcal{L}_1\mathcal{L}_2 = \{m_1m_2, m_1 \in \mathcal{L}_1, m_2 \in \mathcal{L}_2\}$
2. $\{a\}\mathcal{L} = a\mathcal{L}$
3. $\{a\}^* = a^*$
4. $a^+ = aa^*$

Propriétés

1. $\mathcal{L}\emptyset = \emptyset\mathcal{L} = \emptyset$
2. $\mathcal{L}\epsilon = \epsilon\mathcal{L} = \mathcal{L}$
3. $(\mathcal{L}_1\mathcal{L}_2)\mathcal{L}_3 = \mathcal{L}_1(\mathcal{L}_2\mathcal{L}_3)$
4. $\mathcal{L}(\mathcal{L}_1 \cup \mathcal{L}_2) = \mathcal{L}\mathcal{L}_1 \cup \mathcal{L}\mathcal{L}_2$
5. $(\mathcal{L}_1 \cup \mathcal{L}_2)\mathcal{L} = \mathcal{L}_1\mathcal{L} \cup \mathcal{L}_2\mathcal{L}$

Langages

Remarque

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Exemple

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Exemple

$\mathcal{L}_1 = \{\epsilon, a\}, \mathcal{L}_2 = \{b, ab\}$.

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Exemple

$\mathcal{L}_1 = \{\epsilon, a\}, \mathcal{L}_2 = \{b, ab\}$.
$\mathcal{L}_1\mathcal{L}_2 = \{b, ab, aab\} = \{b, ab, a^2b\}$

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Exemple

$\mathcal{L}_1 = \{\epsilon, a\}, \mathcal{L}_2 = \{b, ab\}$.
$\mathcal{L}_1\mathcal{L}_2 = \{b, ab, aab\} = \{b, ab, a^2b\}$
et $\mathcal{L}_2\mathcal{L}_1 = \{b, ab, ba, aba\}$.

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Exemple

$\mathcal{L}_1 = \{\epsilon, a\}, \mathcal{L}_2 = \{b, ab\}$.
$\mathcal{L}_1\mathcal{L}_2 = \{b, ab, aab\} = \{b, ab, a^2b\}$
et $\mathcal{L}_2\mathcal{L}_1 = \{b, ab, ba, aba\}$.

Définition

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Exemple

$\mathcal{L}_1 = \{\epsilon, a\}, \mathcal{L}_2 = \{b, ab\}$.
$\mathcal{L}_1\mathcal{L}_2 = \{b, ab, aab\} = \{b, ab, a^2b\}$
et $\mathcal{L}_2\mathcal{L}_1 = \{b, ab, ba, aba\}$.

Définition

L'associativité permet de définir :
$\mathcal{L}^0 = \{\epsilon\}$, $\mathcal{L}^1 = \mathcal{L}$ et pour $n \geq 2$: $\mathcal{L}^n = \{m_1m_2\cdots m_n,\ m_i \in \mathcal{L}\}$

Langages

Remarque

En général, $\mathcal{L}_1\mathcal{L}_2 \neq \mathcal{L}_2\mathcal{L}_1$.

Exemple

$\mathcal{L}_1 = \{\epsilon, a\}, \mathcal{L}_2 = \{b, ab\}$.
$\mathcal{L}_1\mathcal{L}_2 = \{b, ab, aab\} = \{b, ab, a^2b\}$
et $\mathcal{L}_2\mathcal{L}_1 = \{b, ab, ba, aba\}$.

Définition

L'associativité permet de définir :
$\mathcal{L}^0 = \{\epsilon\}$, $\mathcal{L}^1 = \mathcal{L}$ et pour $n \geq 2$: $\mathcal{L}^n = \{m_1 m_2 \cdots m_n,\ m_i \in \mathcal{L}\}$
Conséquence : $\mathcal{L}^* = \mathcal{L}^0 \cup \mathcal{L}^1 \cup \mathcal{L}^2 \cup \cdots$

Langages

Préfixe, suffixe

Langages

Préfixe, suffixe

Soit $u,\ w \in \mathcal{A}^*$. u est un préfixe de w si $\exists v \in \mathcal{A}^*\ w = uv$.

Langages

Préfixe, suffixe

Soit $u, w \in \mathcal{A}^*$. u est un préfixe de w si $\exists v \in \mathcal{A}^*\ w = uv$.
u est un suffixe de w si $\exists v \in \mathcal{A}^*\ w = vu$.

Langages

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
- Généralités
- Arbre
- Cycle eulérien, hamiltonien
- Coloration
- Le problème du plus court chemin
- Arbre recouvrant de poids minimal

Langages et automates
- Langages
- Automates

Annexes

Préfixe, suffixe

Soit $u, \ w \in \mathcal{A}^*$. u est un préfixe de w si $\exists v \in \mathcal{A}^* \ w = uv$.
u est un suffixe de w si $\exists v \in \mathcal{A}^* \ w = vu$.

Exemples

Langages

Préfixe, suffixe

Soit $u,\ w \in \mathcal{A}^*$. u est un préfixe de w si $\exists v \in \mathcal{A}^*\ w = uv$.
u est un suffixe de w si $\exists v \in \mathcal{A}^*\ w = vu$.

Exemples

$\mathcal{A} = \{a, b\}$. $w = babb$

Langages

Préfixe, suffixe

Soit $u,\ w \in \mathcal{A}^*$. u est un préfixe de w si $\exists v \in \mathcal{A}^*\ w = uv$.
u est un suffixe de w si $\exists v \in \mathcal{A}^*\ w = vu$.

Exemples

$\mathcal{A} = \{a, b\}$. $w = babb$
Les préfixes de w sont : ϵ, b, ba, bab et $babb$.

Langages

Préfixe, suffixe

Soit $u,\ w \in \mathcal{A}^*$. u est un préfixe de w si $\exists v \in \mathcal{A}^*\ w = uv$.
u est un suffixe de w si $\exists v \in \mathcal{A}^*\ w = vu$.

Exemples

$\mathcal{A} = \{a, b\}$. $w = babb$
Les préfixes de w sont : ϵ, b, ba, bab et $babb$.
Les suffixes de w sont : ϵ, b, bb, abb et $babb$.

Langages

Autre définition : nombre de lettres α dans un mot

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1x_2\ldots x_n$ un mot de $\mathcal{A}^*$.

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1 x_2 \dots x_n$ un mot de $\mathcal{A}^*$.
On note $|m|_\alpha = |\{i \in [\![1, n]\!] \; x_i = \alpha\}|$.

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1x_2 \ldots x_n$ un mot de $\mathcal{A}^*$.
On note $|m|_\alpha = |\{i \in [\![1, n]\!] \; x_i = \alpha\}|$.
Il s'agit du nombre de lettres α apparaissant dans le mot m.

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1 x_2 \dots x_n$ un mot de $\mathcal{A}^*$.
On note $|m|_\alpha = |\{i \in [\![1, n]\!]\ x_i = \alpha\}|$.
Il s'agit du nombre de lettres α apparaissant dans le mot m.

Exemple

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1 x_2 \ldots x_n$ un mot de $\mathcal{A}^*$.
On note $|m|_\alpha = |\{i \in [\![1, n]\!] \; x_i = \alpha\}|$.
Il s'agit du nombre de lettres α apparaissant dans le mot m.

Exemple

$\mathcal{A} = \{I, U, T\}$.

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1 x_2 \dots x_n$ un mot de $\mathcal{A}^*$.
On note $|m|_\alpha = |\{i \in [\![1, n]\!] \; x_i = \alpha\}|$.
Il s'agit du nombre de lettres α apparaissant dans le mot m.

Exemple

$\mathcal{A} = \{I, U, T\}$.
$|IUT|_I = 1$

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1 x_2 \dots x_n$ un mot de $\mathcal{A}^*$.
On note $|m|_\alpha = |\{i \in [\![1, n]\!] \; x_i = \alpha\}|$.
Il s'agit du nombre de lettres α apparaissant dans le mot m.

Exemple

$\mathcal{A} = \{I, U, T\}$.
$|IUT|_I = 1$
$|UT|_I = 0$

Langages

Autre définition : nombre de lettres α dans un mot

Soit $\alpha \in \mathcal{A}$ et $m = x_1 x_2 \ldots x_n$ un mot de $\mathcal{A}^*$.
On note $|m|_\alpha = |\{i \in [\![1, n]\!] \; x_i = \alpha\}|$.
Il s'agit du nombre de lettres α apparaissant dans le mot m.

Exemple

$\mathcal{A} = \{I, U, T\}$.
$|IUT|_I = 1$
$|UT|_I = 0$
$|ITT|_T = 2$

Langages

Langages

Expression régulière ou rationnelle

Langages

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

1. ϵ et ( ) sont des expressions régulières (( ) est parfois noté ($\emptyset$)).

Langages

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

1. ϵ et ( ) sont des expressions régulières (( ) est parfois noté $(\emptyset)$).
2. si a est une lettre, a est une expression régulière.

Langages

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

1. ϵ et ( ) sont des expressions régulières (( ) est parfois noté ($\emptyset$)).
2. si a est une lettre, a est une expression régulière.
3. si r est une régulière, (r^*) est une expression régulière.

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

1. ϵ et ( ) sont des expressions régulières (( ) est parfois noté ($\emptyset$)).
2. si a est une lettre, a est une expression régulière.
3. si r est une régulière, (r^*) est une expression régulière.
4. si r_1 et r_2 sont des expressions régulières, $(r_1 \vee r_2)$ et $(r_1 r_2)$ sont des expressions régulières.

Langages

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

1. ϵ et ( ) sont des expressions régulières (( ) est parfois noté $(\emptyset)$).
2. si a est une lettre, a est une expression régulière.
3. si r est une régulière, (r^*) est une expression régulière.
4. si r_1 et r_2 sont des expressions régulières, $(r_1 \vee r_2)$ et $(r_1 r_2)$ sont des expressions régulières.

Les expressions régulières sont formées à partir de ces règles.

Langages

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

1. ϵ et ( ) sont des expressions régulières (( ) est parfois noté ($\emptyset$)).
2. si a est une lettre, a est une expression régulière.
3. si r est une régulière, (r^*) est une expression régulière.
4. si r_1 et r_2 sont des expressions régulières, $(r_1 \vee r_2)$ et $(r_1 r_2)$ sont des expressions régulières.

Les expressions régulières sont formées à partir de ces règles.
Elles sont construites à partir de lettres de l'alphabet et des cinq symboles :

Langages

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Expression régulière ou rationnelle

Définition : Une expression régulière ou rationnelle sur un alphabet $\mathcal{A}$ est définie par

1. ϵ et ( ) sont des expressions régulières (( ) est parfois noté ($\emptyset$)).
2. si a est une lettre, a est une expression régulière.
3. si r est une régulière, (r^*) est une expression régulière.
4. si r_1 et r_2 sont des expressions régulières, $(r_1 \vee r_2)$ et $(r_1 r_2)$ sont des expressions régulières.

Les expressions régulières sont formées à partir de ces règles.
Elles sont construites à partir de lettres de l'alphabet et des cinq symboles : ( ) * $\vee$ ϵ

Langages

Langages

Expression régulière

Langages

Expression régulière

Exemples sur $\mathcal{A} = \{a, b\}$:

Langages

Expression régulière

Exemples sur $\mathcal{A} = \{a, b\}$:

1. $(a \vee b)$,

Langages

Expression régulière

Exemples sur $\mathcal{A} = \{a, b\}$:

1. $(a \vee b)$,
2. (ab),

Langages

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Expression régulière

Exemples sur $\mathcal{A} = \{a, b\}$:

1. $(a \vee b)$,
2. (ab),
3. (a^*),

Langages

Expression régulière

Exemples sur $\mathcal{A} = \{a, b\}$:

1. $(a \vee b)$,
2. (ab),
3. (a^*),
4. $a((a \vee ab)^* b)$.

Langages

Expression régulière

Exemples sur $\mathcal{A} = \{a, b\}$:

1. $(a \vee b)$,
2. (ab),
3. (a^*),
4. $a((a \vee ab)^* b)$.
5. $((a \vee b(a)^*)^*(a^2 \vee ab \vee aba))^*$.

Langages

Langages

Expression régulière

Langages

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

Langages

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

1. $\mathcal{L}(\epsilon) = \{\epsilon\}$

Langages

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

1. $\mathcal{L}(\epsilon) = \{\epsilon\}$
2. $\mathcal{L}((\ )) = \emptyset$

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

1. $\mathcal{L}(\epsilon) = \{\epsilon\}$
2. $\mathcal{L}((\ )) = \emptyset$
3. a est une lettre de $\mathcal{A}$, $\mathcal{L}(a) = \{a\}$

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

1. $\mathcal{L}(\epsilon) = \{\epsilon\}$
2. $\mathcal{L}((\ )) = \emptyset$
3. a est une lettre de $\mathcal{A}$, $\mathcal{L}(a) = \{a\}$
4. $\mathcal{L}(r^*) = (\mathcal{L}(r))^*$

Langages

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

1. $\mathcal{L}(\epsilon) = \{\epsilon\}$
2. $\mathcal{L}((\ )) = \emptyset$
3. a est une lettre de $\mathcal{A}$, $\mathcal{L}(a) = \{a\}$
4. $\mathcal{L}(r^*) = (\mathcal{L}(r))^*$
5. $\mathcal{L}(r_1 \vee r_2) = \mathcal{L}(r_1) \cup \mathcal{L}(r_2)$

Langages

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

1. $\mathcal{L}(\epsilon) = \{\epsilon\}$
2. $\mathcal{L}((\ )) = \emptyset$
3. a est une lettre de $\mathcal{A}$, $\mathcal{L}(a) = \{a\}$
4. $\mathcal{L}(r^*) = (\mathcal{L}(r))^*$
5. $\mathcal{L}(r_1 \vee r_2) = \mathcal{L}(r_1) \cup \mathcal{L}(r_2)$
6. $\mathcal{L}(r_1 r_2) = \mathcal{L}(r_1)\mathcal{L}(r_2)$

Langages

Expression régulière

Définition : Le langage $\mathcal{L}(r)$ associé à une expression régulière r sur un alphabet $\mathcal{A}$ est défini par

1. $\mathcal{L}(\epsilon) = \{\epsilon\}$
2. $\mathcal{L}((\ )) = \emptyset$
3. a est une lettre de $\mathcal{A}$, $\mathcal{L}(a) = \{a\}$
4. $\mathcal{L}\left(r^*\right) = \left(\mathcal{L}(r)\right)^*$
5. $\mathcal{L}\left(r_1 \vee r_2\right) = \mathcal{L}\left(r_1\right) \cup \mathcal{L}\left(r_2\right)$
6. $\mathcal{L}\left(r_1 r_2\right) = \mathcal{L}\left(r_1\right) \mathcal{L}\left(r_2\right)$

Des parenthèses peuvent être omises du fait de l'associativité de la concaténation et de l'union, ainsi que des priorités : étoile prioritaire sur la concaténation qui l'est sur l'union.

Langages

Langages

Langage régulier ou rationnel

Langages

Langage régulier ou rationnel

Définition : Un langage $\mathcal{L}$ sur un alphabet $\mathcal{A}$ est appelé un langage régulier ou rationnel s'il existe une expression régulière r sur $\mathcal{A}$ telle que $\mathcal{L} = \mathcal{L}(r)$.

Langages

Langages

Exemples de langage régulier

Exemples de langage régulier

$\mathcal{A} = \{a, b\}$.

1. $r = a^* \ : \ \mathcal{L}(r) = a^*$

Langages

Exemples de langage régulier

$\mathcal{A} = \{a, b\}$.

1. $r = a^* \ : \ \mathcal{L}(r) = a^*$
2. $r = aa^* \ : \ \mathcal{L}(r) = aa^*$

Langages

Exemples de langage régulier

$\mathcal{A} = \{a, b\}$.

1. $r = a^* \, : \, \mathcal{L}(r) = a^*$
2. $r = aa^* \, : \, \mathcal{L}(r) = aa^*$
3. $r = a \vee b^* \, : \, \mathcal{L}(a \vee b^*) = \mathcal{L}(a) \cup \mathcal{L}(b^*) = \{a\} \cup b^* = \{a, \epsilon, b, b^2, \ldots\}$

Langages

Exemples de langage régulier

$\mathcal{A} = \{a, b\}$.

1. $r = a^* \; : \; \mathcal{L}(r) = a^*$
2. $r = aa^* \; : \; \mathcal{L}(r) = aa^*$
3. $r = a \vee b^* \; : \; \mathcal{L}(a \vee b^*) = \mathcal{L}(a) \cup \mathcal{L}(b^*) = \{a\} \cup b^* = \{a, \epsilon, b, b^2, \ldots\}$
4. $r = (a \vee b)^* \; : \; \mathcal{L}((a \vee b)^*) = (\mathcal{L}(a \vee b))^* = (\mathcal{L}(a) \cup \mathcal{L}(b))^* = (\{a\} \cup \{b\})^* = \{a, b\}^* = \mathcal{A}^*$

Langages

Exemples de langage régulier

$\mathcal{A} = \{a, b\}$.

1. $r = a^*$: $\mathcal{L}(r) = a^*$
2. $r = aa^*$: $\mathcal{L}(r) = aa^*$
3. $r = a \vee b^*$: $\mathcal{L}(a \vee b^*) = \mathcal{L}(a) \cup \mathcal{L}(b^*) = \{a\} \cup b^* = \{a, \epsilon, b, b^2, \dots\}$
4. $r = (a \vee b)^*$: $\mathcal{L}((a \vee b)^*) = (\mathcal{L}(a \vee b))^* = (\mathcal{L}(a) \cup \mathcal{L}(b))^* = (\{a\} \cup \{b\})^* = \{a, b\}^* = \mathcal{A}^*$
5. $r = (a \vee b)^* bbb = \mathcal{A}^* bbb$: $\mathcal{L}(r)$ est l'ensemble des mots se terminant par bbb.

Automates

Définition

Automates

Définition

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est la donnée de

Automates

M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Définition

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est la donnée de

1. $\mathcal{A}$: un alphabet

Automates

Définition

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est la donnée de

1. $\mathcal{A}$: un alphabet
2. $\mathcal{E}$: un ensemble fini non vide, ensemble des états de l'automate

Automates

Définition

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est la donnée de

1. $\mathcal{A}$: un alphabet
2. $\mathcal{E}$: un ensemble fini non vide, ensemble des états de l'automate
3. i_0 : un état particulier (état initial)

Automates

Définition

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est la donnée de

1. $\mathcal{A}$: un alphabet
2. $\mathcal{E}$: un ensemble fini non vide, ensemble des états de l'automate
3. i_0 : un état particulier (état initial)
4. $\mathcal{F} \subset \mathcal{E}$, ensemble des états finaux, terminaux ou acceptants

Automates

Définition

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est la donnée de

1. $\mathcal{A}$: un alphabet
2. $\mathcal{E}$: un ensemble fini non vide, ensemble des états de l'automate
3. i_0 : un état particulier (état initial)
4. $\mathcal{F} \subset \mathcal{E}$, ensemble des états finaux, terminaux ou acceptants
5. $\delta : \mathcal{E} \times \mathcal{A} \to \mathcal{E}$ une application dite de transition.

Automates

Exemple

Automates

Exemple

$\mathcal{A} = \{0, 1\}$,

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$,

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$,

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 0) = P$, $\delta(P, 1) = I$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 0) = P$, $\delta(P, 1) = I$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.

Représentation graphique

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 0) = P$, $\delta(P, 1) = I$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.

Représentation graphique

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est représenté par un diagramme : graphe orienté dont les sommets sont les états et les arcs les transitions.

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 0) = P$, $\delta(P, 1) = I$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.

Représentation graphique

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est représenté par un diagramme : graphe orienté dont les sommets sont les états et les arcs les transitions.

1. Un état est repéré par un cercle,

Automates

Exemple

$\mathcal{A} = \{0,1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P,0) = P$, $\delta(P,1) = I$, $\delta(I,0) = I$, $\delta(I,1) = P$.

Représentation graphique

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est représenté par un diagramme : graphe orienté dont les sommets sont les états et les arcs les transitions.

1. Un état est repéré par un cercle,
2. un état final est repéré par un double cercle,

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 0) = P$, $\delta(P, 1) = I$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.

Représentation graphique

Un automate fini $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ est représenté par un diagramme : graphe orienté dont les sommets sont les états et les arcs les transitions.

1. Un état est repéré par un cercle,
2. un état final est repéré par un double cercle,
3. l'état initial est repéré par une flèche.

Automates

Automates

Exemple

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,

Automates

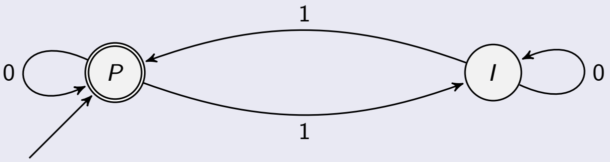
Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 1) = I$, $\delta(P, 0) = P$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.

Automates

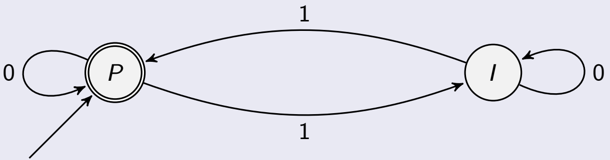
Exemple

$\mathcal{A} = \{0,1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P,1) = I$, $\delta(P,0) = P$, $\delta(I,0) = I$, $\delta(I,1) = P$.



Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 1) = I$, $\delta(P, 0) = P$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.

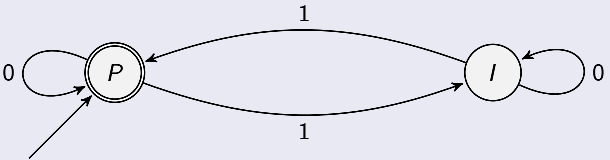


P est un état acceptant

Automates

Exemple

$\mathcal{A} = \{0, 1\}$, $\mathcal{E} = \{P, I\}$, $i_0 = P$, $\mathcal{F} = \{P\}$,
$\delta(P, 1) = I$, $\delta(P, 0) = P$, $\delta(I, 0) = I$, $\delta(I, 1) = P$.



P est un état acceptant , I est un état refusant (non-acceptant).

Automates

Table des transitions

Automates

Table des transitions

T	P	I
0	P	I
1	I	P

Automates

Table des transitions

T	P	I
0	P	I
1	I	P

Relation

Automates

Table des transitions

T	P	I
0	P	I
1	I	P

Relation

$$\mathcal{R} = \{(P, 0, P), (P, 1, I), (I, 0, I), (I, 1, P)\}$$

Automates

Définitions

Automates

Définitions

i est un état quelconque. On pose :

Automates

Définitions

i est un état quelconque. On pose :

1. $\delta(i, \epsilon) = i$.

Automates

Définitions

i est un état quelconque. On pose :

1. $\delta(i, \epsilon) = i$.
2. pour toutes lettres a_1, a_2, $\delta(i, a_1 a_2) = \delta(\delta(i, a_1), a_2)$.

Automates

Définitions

i est un état quelconque. On pose :

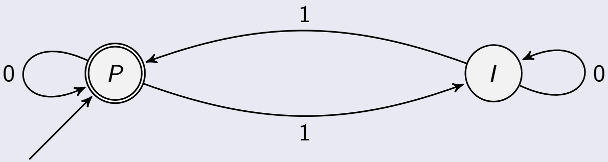
1. $\delta(i, \epsilon) = i$.
2. pour toutes lettres a_1, a_2, $\delta(i, a_1 a_2) = \delta(\delta(i, a_1), a_2)$.
3. pour tous mots m_1, m_2, $\delta(i, m_1 m_2) = \delta(\delta(i, m_1), m_2)$.

Automates

Exemple

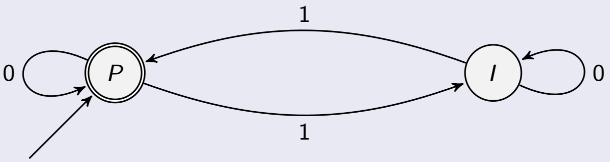
Automates

Exemple



Automates

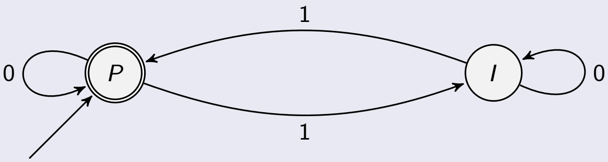
Exemple



1. $\delta(P, \epsilon) = P$.

Automates

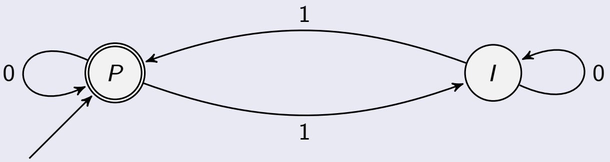
Exemple



1. $\delta(P, \epsilon) = P$.
2. $\delta(P, 01) = \delta(\delta(P, 0), 1) = \delta(P, 1) = I$.

Automates

Exemple



1. $\delta(P, \epsilon) = P$.
2. $\delta(P, 01) = \delta(\delta(P, 0), 1) = \delta(P, 1) = I$.
3. $\delta(P, 011) = \delta(\delta(P, 01), 1) = \delta(I, 1) = P$.

Automates

Automates

Langage accepté ou reconnu par un automate

Automates

Langage accepté ou reconnu par un automate

Un mot est accepté ou reconnu par un automate s'il permet de passer de l'état initial à un état final.

Automates

Langage accepté ou reconnu par un automate

Un mot est accepté ou reconnu par un automate s'il permet de passer de l'état initial à un état final.

L'ensemble des mots acceptés par un automate est appelé le langage accepté par l'automate ou langage de l'automate.

Automates

Langage accepté ou reconnu par un automate

Un mot est accepté ou reconnu par un automate s'il permet de passer de l'état initial à un état final.
L'ensemble des mots acceptés par un automate est appelé le langage accepté par l'automate ou langage de l'automate.

Exemple

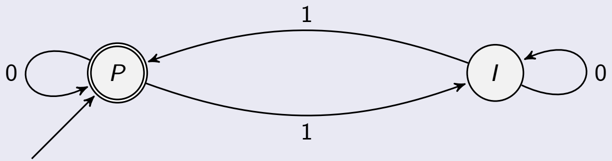
Automates

Langage accepté ou reconnu par un automate

Un mot est accepté ou reconnu par un automate s'il permet de passer de l'état initial à un état final.
L'ensemble des mots acceptés par un automate est appelé le langage accepté par l'automate ou langage de l'automate.

Exemple

ϵ, 0000, 01000100 sont des mots acceptés par l'automate :



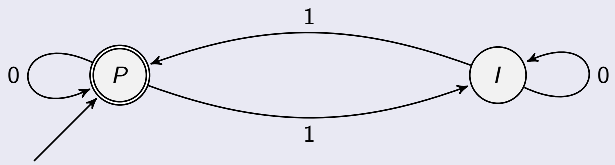
Automates

Langage accepté ou reconnu par un automate

Un mot est accepté ou reconnu par un automate s'il permet de passer de l'état initial à un état final.
L'ensemble des mots acceptés par un automate est appelé le langage accepté par l'automate ou langage de l'automate.

Exemple

ϵ, 0000, 01000100 sont des mots acceptés par l'automate :



010, 111011 sont des mots non acceptés ou refusés.

Automates

Automates

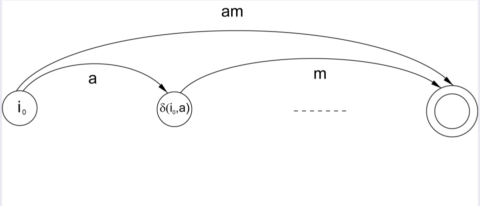
Détermination du langage d'un automate

Détermination du langage d'un automate

Un mot am est accepté par un automate si et seulement si m est accepté par l'automate dont l'état initial est $\delta(i_0, a)$.

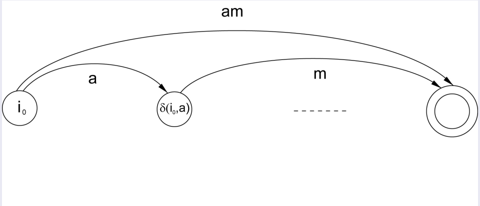
Détermination du langage d'un automate

Un mot *am* est accepté par un automate si et seulement si *m* est accepté par l'automate dont l'état initial est $\delta(i_0, a)$.



Détermination du langage d'un automate

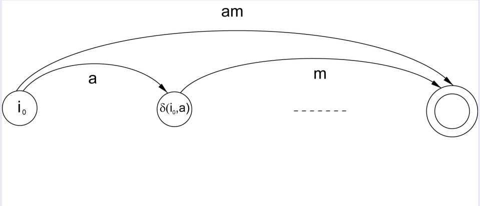
Un mot *am* est accepté par un automate si et seulement si *m* est accepté par l'automate dont l'état initial est $\delta(i_0, a)$.



Cas particulier

Détermination du langage d'un automate

Un mot *am* est accepté par un automate si et seulement si *m* est accepté par l'automate dont l'état initial est $\delta(i_0, a)$.



Cas particulier

L'état i est un état final ou acceptant si et seulement si ϵ est accepté par l'automate dont l'état initial est i : $\delta(i, \epsilon) = i$.

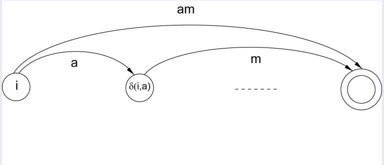
Automates

Automates

Méthode du départ

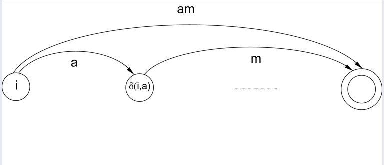
Automates

Méthode du départ



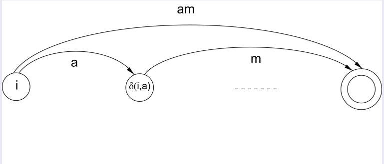
Automates

Méthode du départ



On associe à chaque état i le langage X_i accepté par l'automate si i était son état initial.

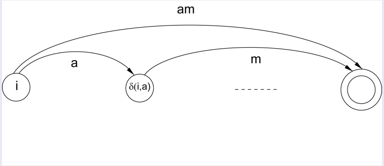
Méthode du départ



On associe à chaque état i le langage X_i accepté par l'automate si i était son état initial.
On en déduit pour chaque état i l'équation :

Automates

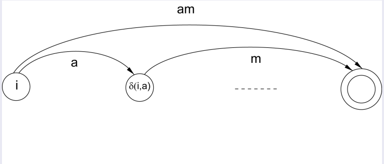
Méthode du départ



On associe à chaque état i le langage X_i accepté par l'automate si i était son état initial.
On en déduit pour chaque état i l'équation :

$$X_i = \varphi_i \cup aX_{\delta(i,a)} \cup bX_{\delta(i,b)} \cup cX_{\delta(i,c)} \cup \cdots$$

Méthode du départ



On associe à chaque état i le langage X_i accepté par l'automate si i était son état initial.
On en déduit pour chaque état i l'équation :

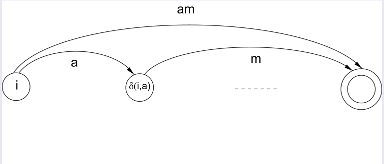
$$X_i = \varphi_i \cup aX_{\delta(i,a)} \cup bX_{\delta(i,b)} \cup cX_{\delta(i,c)} \cup \cdots$$

soit

$$X_i = \varphi_i \cup \bigcup_{a \in A} aX_{\delta(i,a)}$$

Automates

Méthode du départ



On associe à chaque état i le langage X_i accepté par l'automate si i était son état initial.
On en déduit pour chaque état i l'équation :

$$X_i = \varphi_i \cup aX_{\delta(i,a)} \cup bX_{\delta(i,b)} \cup cX_{\delta(i,c)} \cup \cdots$$

soit

$$X_i = \varphi_i \cup \bigcup_{a \in A} aX_{\delta(i,a)}$$

avec $\varphi_i = \epsilon$ si i est acceptant et $\varphi_i = \emptyset$ sinon.

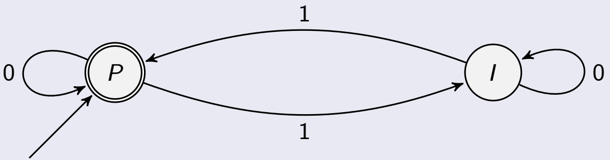
Automates

Automates

Exemple

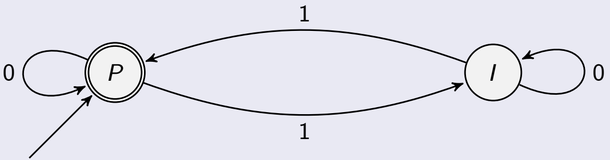
Automates

Exemple



Automates

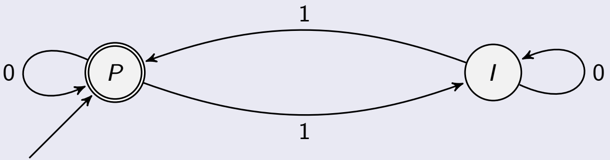
Exemple



$$\begin{cases} X_P = \epsilon \cup 0X_{\delta(P,0)} \cup 1X_{\delta(P,1)} \\ X_I = \emptyset \cup 0X_{\delta(I,0)} \cup 1X_{\delta(I,1)} \end{cases}$$

Automates

Exemple



$$\begin{cases} X_P = \epsilon \cup 0X_{\delta(P,0)} \cup 1X_{\delta(P,1)} \\ X_I = \emptyset \cup 0X_{\delta(I,0)} \cup 1X_{\delta(I,1)} \end{cases}$$

$$\begin{cases} X_P = \epsilon \cup 0X_P \cup 1X_I \\ X_I = 0X_I \cup 1X_P \end{cases}$$

Automates

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup 0X_P \cup 1X_I \\ X_I = 0X_I \cup 1X_P \end{cases}$$

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup 0X_P \cup 1X_I \\ X_I = 0X_I \cup 1X_P \end{cases} \qquad \begin{cases} X_I = 0X_I \cup 1X_P \\ X_P = 1X_I \cup 0X_P \cup \epsilon \end{cases}$$

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup 0X_P \cup 1X_I \\ X_I = 0X_I \cup 1X_P \end{cases} \qquad \begin{cases} X_I = 0X_I \cup 1X_P \\ X_P = 1X_I \cup 0X_P \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = 0^*1X_P \\ X_P = 10^*1X_P \cup 0X_P \cup \epsilon \end{cases}$$

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup 0X_P \cup 1X_I \\ X_I = 0X_I \cup 1X_P \end{cases} \qquad \begin{cases} X_I = 0X_I \cup 1X_P \\ X_P = 1X_I \cup 0X_P \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = 0^*1X_P \\ X_P = 10^*1X_P \cup 0X_P \cup \epsilon \end{cases} \qquad \begin{cases} X_I = 0^*1X_P \\ X_P = (10^*1 \cup 0)X_P \cup \epsilon \end{cases}$$

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup 0X_P \cup 1X_I \\ X_I = 0X_I \cup 1X_P \end{cases} \quad \begin{cases} X_I = 0X_I \cup 1X_P \\ X_P = 1X_I \cup 0X_P \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = 0^*1X_P \\ X_P = 10^*1X_P \cup 0X_P \cup \epsilon \end{cases} \quad \begin{cases} X_I = 0^*1X_P \\ X_P = (10^*1 \cup 0)X_P \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = 0^*1X_P \\ X_P = (10^*1 \cup 0)^*\epsilon \end{cases}$$

Automates

Résolution du système (lemme d'Arden - Dean N. Arden 1960)

L'équation $X = \mathcal{L}X \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}^*\mathcal{L}'$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup 0X_P \cup 1X_I \\ X_I = 0X_I \cup 1X_P \end{cases} \qquad \begin{cases} X_I = 0X_I \cup 1X_P \\ X_P = 1X_I \cup 0X_P \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = 0^*1X_P \\ X_P = 10^*1X_P \cup 0X_P \cup \epsilon \end{cases} \qquad \begin{cases} X_I = 0^*1X_P \\ X_P = (10^*1 \cup 0)X_P \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = 0^*1X_P \\ X_P = (10^*1 \cup 0)^*\epsilon \end{cases} \qquad \begin{cases} X_P = (10^*1 \cup 0)^* \\ X_I = 0^*1(10^*1 \cup 0)^* \end{cases}$$

Automates

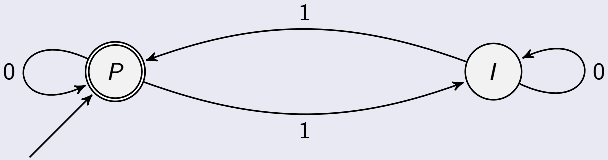
Automates

Exemple

Automates

Exemple

Conclusion : le langage de l'automate est $X_P = (10^*1 \cup 0)^*$



Automates

Automates

Méthode de l'arrivée

Méthode de l'arrivée

On associe à l'état j le langage d'arrivée A_j de l'état j :

Méthode de l'arrivée

On associe à l'état j le langage d'arrivée A_j de l'état j : ensemble des mots qui font arriver à l'état j en étant parti de l'état initial.

Automates

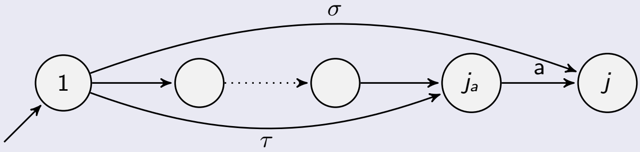
Méthode de l'arrivée

On associe à l'état j le langage d'arrivée A_j de l'état j : ensemble des mots qui font arriver à l'état j en étant parti de l'état initial.
Le langage de l'automate est alors l'union des langages A_j pour tous les états j acceptants.

Automates

Méthode de l'arrivée

On associe à l'état j le langage d'arrivée A_j de l'état j : ensemble des mots qui font arriver à l'état j en étant parti de l'état initial.
Le langage de l'automate est alors l'union des langages A_j pour tous les états j acceptants.



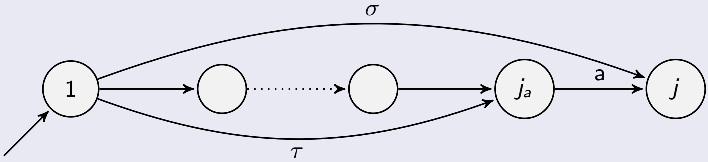
Automates

Automates

Méthode de l'arrivée

Automates

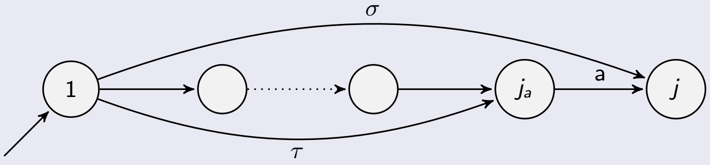
Méthode de l'arrivée



Les langages d'arrivée sont régis par les équations
$X_j = \varphi_j \cup X_{j_a} a \cup X_{j_b} b \cup X_{j_c} c \cup \cdots$ avec $\delta(j_a, a) = \delta(j_b, b) = \delta(j_c, c) = \ldots = j$

Automates

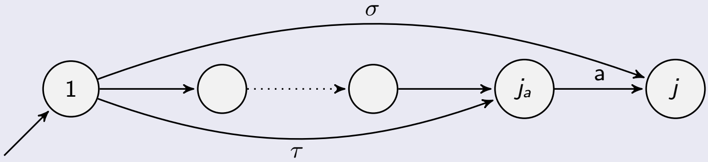
Méthode de l'arrivée



Les langages d'arrivée sont régis par les équations
$X_j = \varphi_j \cup X_{j_a} a \cup X_{j_b} b \cup X_{j_c} c \cup \cdots$ avec $\delta(j_a, a) = \delta(j_b, b) = \delta(j_c, c) = \ldots = j$
soit $X_j = \varphi_j \cup \bigcup_{a \in \mathcal{A},\ \delta(j_a, a) = j} X_{j_a} a$

Automates

Méthode de l'arrivée



Les langages d'arrivée sont régis par les équations
$X_j = \varphi_j \cup X_{j_a} a \cup X_{j_b} b \cup X_{j_c} c \cup \cdots$ avec $\delta(j_a, a) = \delta(j_b, b) = \delta(j_c, c) = \ldots = j$
soit $X_j = \varphi_j \cup \bigcup_{a \in \mathcal{A},\ \delta(j_a, a) = j} X_{j_a} a$
avec $\varphi_j = \epsilon$ si $j = 1$, $\varphi_j = \emptyset$ sinon.

Automates

M2201
Graphes et langages

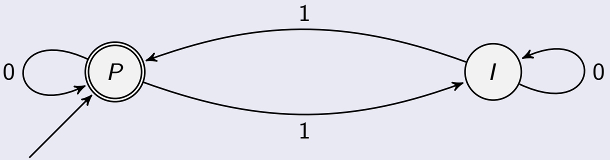
Département Informatique IUT de Saint-Dié

Automates

Exemple

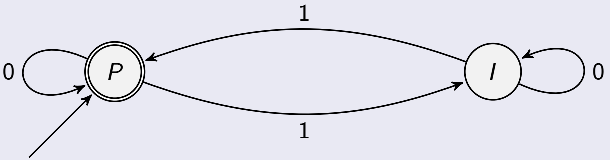
Automates

Exemple



Automates

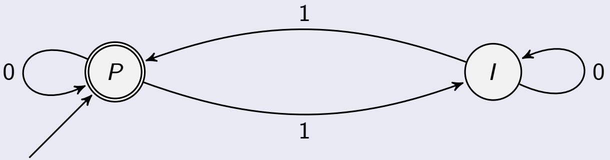
Exemple



$$\begin{cases} X_P = \epsilon \cup X_{j_0}0 \cup X_{j_1}1 & (1) \\ X_I = \emptyset \cup X_{j'_0}0 \cup X_{j'_1}1 & (2) \end{cases}$$

Automates

Exemple

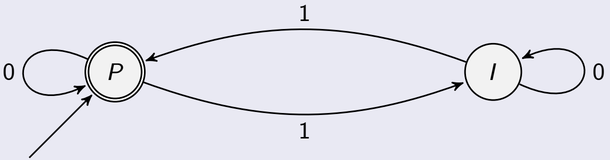


$$\begin{cases} X_P = \epsilon \cup X_{j_0}0 \cup X_{j_1}1 \ (1) \\ X_I = \emptyset \cup X_{j_0'}0 \cup X_{j_1'}1 \ (2) \end{cases}$$

(1) $\delta(j_0, 0) = P$ pour $j_0 = P$ et $\delta(j_1, 1) = P$ pour $j_1 = I$

Automates

Exemple



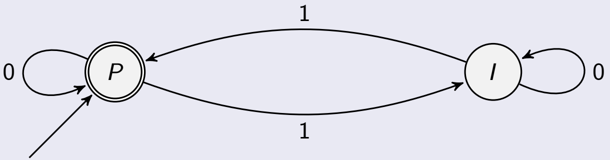
$$\begin{cases} X_P = \epsilon \cup X_{j_0}0 \cup X_{j_1}1 \ (1) \\ X_I = \emptyset \cup X_{j_0'}0 \cup X_{j_1'}1 \ (2) \end{cases}$$

(1) $\delta(j_0, 0) = P$ pour $j_0 = P$ et $\delta(j_1, 1) = P$ pour $j_1 = I$

(2) $\delta(j_0', 0) = I$ pour $j_0' = I$ et $\delta(j_1', 1) = I$ pour $j_1' = P$

Automates

Exemple



$$\begin{cases} X_P = \epsilon \cup X_{j_0}0 \cup X_{j_1}1 \ (1) \\ X_I = \emptyset \cup X_{j'_0}0 \cup X_{j'_1}1 \ (2) \end{cases}$$

(1) $\delta(j_0, 0) = P$ pour $j_0 = P$ et $\delta(j_1, 1) = P$ pour $j_1 = I$

(2) $\delta(j'_0, 0) = I$ pour $j'_0 = I$ et $\delta(j'_1, 1) = I$ pour $j'_1 = P$

$$\begin{cases} X_P = \epsilon \cup X_P0 \cup X_I1 \\ X_I = X_I0 \cup X_P1 \end{cases}$$

Automates

Automates

Lemme d'Arden : version 2

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup X_P 0 \cup X_I 1 \\ X_I = X_I 0 \cup X_P 1 \end{cases}$$

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup X_P 0 \cup X_I 1 \\ X_I = X_I 0 \cup X_P 1 \end{cases} \qquad \begin{cases} X_I = X_I 0 \cup X_P 1 \\ X_P = X_I 1 \cup X_P 0 \cup \epsilon \end{cases}$$

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup X_P0 \cup X_I1 \\ X_I = X_I0 \cup X_P1 \end{cases} \qquad \begin{cases} X_I = X_I0 \cup X_P1 \\ X_P = X_I1 \cup X_P0 \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = X_P10^* \\ X_P = X_P10^*1 \cup X_P0 \cup \epsilon \end{cases}$$

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup X_P 0 \cup X_I 1 \\ X_I = X_I 0 \cup X_P 1 \end{cases} \qquad \begin{cases} X_I = X_I 0 \cup X_P 1 \\ X_P = X_I 1 \cup X_P 0 \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = X_P 10^* \\ X_P = X_P 10^* 1 \cup X_P 0 \cup \epsilon \end{cases} \qquad \begin{cases} X_P = X_P(0 \cup 10^* 1) \cup \epsilon \\ X_I = X_P 10^* \end{cases}$$

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup X_P 0 \cup X_I 1 \\ X_I = X_I 0 \cup X_P 1 \end{cases} \qquad \begin{cases} X_I = X_I 0 \cup X_P 1 \\ X_P = X_I 1 \cup X_P 0 \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = X_P 10^* \\ X_P = X_P 10^* 1 \cup X_P 0 \cup \epsilon \end{cases} \qquad \begin{cases} X_P = X_P(0 \cup 10^* 1) \cup \epsilon \\ X_I = X_P 10^* \end{cases}$$

$$\begin{cases} X_P = \epsilon(0 \cup 10^* 1)^* \\ X_I = X_P 10^* \end{cases}$$

Automates

Lemme d'Arden : version 2

L'équation $X = X\mathcal{L} \cup \mathcal{L}'$ a

1. une solution unique $X = \mathcal{L}'\mathcal{L}^*$ si $\epsilon \notin \mathcal{L}$
2. pour solution $X = (\mathcal{L}' \cup \mathcal{L}'')\mathcal{L}^*$ où $\mathcal{L}''$ est un langage quelconque, si $\epsilon \in \mathcal{L}$.

Exemple

$$\begin{cases} X_P = \epsilon \cup X_P 0 \cup X_I 1 \\ X_I = X_I 0 \cup X_P 1 \end{cases} \qquad \begin{cases} X_I = X_I 0 \cup X_P 1 \\ X_P = X_I 1 \cup X_P 0 \cup \epsilon \end{cases}$$

$$\begin{cases} X_I = X_P 10^* \\ X_P = X_P 10^* 1 \cup X_P 0 \cup \epsilon \end{cases} \qquad \begin{cases} X_P = X_P(0 \cup 10^* 1) \cup \epsilon \\ X_I = X_P 10^* \end{cases}$$

$$\begin{cases} X_P = \epsilon(0 \cup 10^* 1)^* \\ X_I = X_P 10^* \end{cases} \qquad \begin{cases} \color{red}{X_P = (0 \cup 10^* 1)^*} \\ X_I = (10^* 1 \cup 0)^* 10^* \end{cases}$$

Automates

Automates

États inaccessibles

Automates

États inaccessibles

Définition :

Automates

États inaccessibles

Définition : Les états j tels que $A_j = \emptyset$ sont dits inaccessibles.

Automates

États inaccessibles

Définition : Les états j tels que $A_j = \emptyset$ sont dits inaccessibles.
Les autres sont dits accessibles.

Automates

États inaccessibles

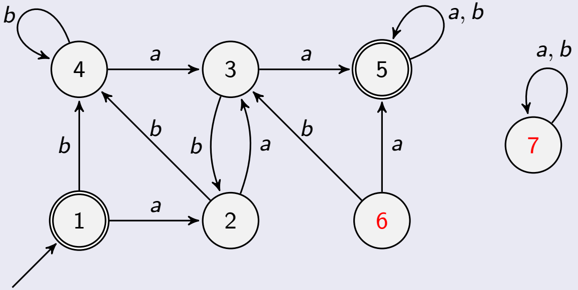
Définition : Les états j tels que $A_j = \emptyset$ sont dits inaccessibles.
Les autres sont dits accessibles.

Exemple

États inaccessibles

Définition : Les états j tels que $A_j = \emptyset$ sont dits inaccessibles.
Les autres sont dits accessibles.

Exemple



Automates

Automates

Automate accessible

Automates

Automate accessible

Définition :

Automates

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Automates

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Résiduel

Automates

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Résiduel

Définition :

Automates

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Résiduel

Définition : Le résiduel d'un langage $\mathcal{L}$ par rapport à un mot σ de $\mathcal{A}^*$ est le langage formé de tous les mots τ tels que $\sigma\tau \in \mathcal{L}$.

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Résiduel

Définition : Le résiduel d'un langage $\mathcal{L}$ par rapport à un mot σ de $\mathcal{A}^*$ est le langage formé de tous les mots τ tels que $\sigma\tau \in \mathcal{L}$.
Notation : $\sigma^{-1}\mathcal{L}$.

Automates

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Résiduel

Définition : Le résiduel d'un langage $\mathcal{L}$ par rapport à un mot σ de $\mathcal{A}^*$ est le langage formé de tous les mots τ tels que $\sigma\tau \in \mathcal{L}$.
Notation : $\sigma^{-1}\mathcal{L}$.
$\sigma^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \sigma\tau \in \mathcal{L}\}$.

Automates

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Résiduel

Définition : Le résiduel d'un langage $\mathcal{L}$ par rapport à un mot σ de $\mathcal{A}^*$ est le langage formé de tous les mots τ tels que $\sigma\tau \in \mathcal{L}$.
Notation : $\sigma^{-1}\mathcal{L}$.
$\sigma^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \sigma\tau \in \mathcal{L}\}$.
L'ensemble des résiduels est noté $Q(\mathcal{L})$ ou $res(\mathcal{L})$.

Automates

Automate accessible

Définition : Un automate ne comportant que des états accessibles est dit accessible.

Résiduel

Définition : Le résiduel d'un langage $\mathcal{L}$ par rapport à un mot σ de $\mathcal{A}^*$ est le langage formé de tous les mots τ tels que $\sigma\tau \in \mathcal{L}$.
Notation : $\sigma^{-1}\mathcal{L}$.
$\sigma^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \sigma\tau \in \mathcal{L}\}$.
L'ensemble des résiduels est noté $Q(\mathcal{L})$ ou $res(\mathcal{L})$.
$Q(\mathcal{L}) = \{\sigma^{-1}\mathcal{L} \ , \ \sigma \in \mathcal{A}^*\}$

Automates

Automates

Exemple

Automates

Exemple

$\mathcal{A} = \{a, b\}$.

Automates

Exemple

$\mathcal{A} = \{a, b\}$.
$\mathcal{L} = a^*$.

Automates

Exemple

$\mathcal{A} = \{a, b\}$.
$\mathcal{L} = a^*$.
σ est un mot de $\mathcal{A}^*$.

Automates

Exemple

$\mathcal{A} = \{a, b\}$.
$\mathcal{L} = a^*$.
σ est un mot de $\mathcal{A}^*$.

1. Si σ contient la lettre b alors $\sigma^{-1}\mathcal{L} = \emptyset$.

Automates

Exemple

$\mathcal{A} = \{a, b\}$.
$\mathcal{L} = a^*$.
σ est un mot de $\mathcal{A}^*$.

1. Si σ contient la lettre b alors $\sigma^{-1}\mathcal{L} = \emptyset$.
   Aucun mot du langage $\mathcal{L} = a^*$ ne commence par σ.

Exemple

$\mathcal{A} = \{a, b\}$.
$\mathcal{L} = a^*$.
σ est un mot de $\mathcal{A}^*$.

1. Si σ contient la lettre b alors $\sigma^{-1}\mathcal{L} = \emptyset$.
   Aucun mot du langage $\mathcal{L} = a^*$ ne commence par σ.
2. Sinon, $\exists n \geq 0$ $\sigma = a^n$ et $\sigma^{-1}\mathcal{L} = \mathcal{L}$.

Automates

Exemple

$\mathcal{A} = \{a, b\}$.
$\mathcal{L} = a^*$.
σ est un mot de $\mathcal{A}^*$.

1. Si σ contient la lettre b alors $\sigma^{-1}\mathcal{L} = \emptyset$.
   Aucun mot du langage $\mathcal{L} = a^*$ ne commence par σ.
2. Sinon, $\exists n \geq 0$ $\sigma = a^n$ et $\sigma^{-1}\mathcal{L} = \mathcal{L}$.
   Les mots du langage $\mathcal{L} = a^*$ commençant par $\sigma = a^n$ sont $a^n\epsilon$, $a^n a$, $a^n a^2$...

Automates

Exemple

$\mathcal{A} = \{a, b\}$.
$\mathcal{L} = a^*$.
σ est un mot de $\mathcal{A}^*$.

1. Si σ contient la lettre b alors $\sigma^{-1}\mathcal{L} = \emptyset$.
   Aucun mot du langage $\mathcal{L} = a^*$ ne commence par σ.
2. Sinon, $\exists n \geq 0$ $\sigma = a^n$ et $\sigma^{-1}\mathcal{L} = \mathcal{L}$.
   Les mots du langage $\mathcal{L} = a^*$ commençant par $\sigma = a^n$ sont $a^n\epsilon$, $a^n a$, $a^n a^2$...
   $Q(\mathcal{L}) = \{\emptyset, \mathcal{L}\}$

Automates

Propriétés

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

   Démonstration :

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

   Démonstration :
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \sigma\tau u \in \mathcal{L}\}$

Automates

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

   Démonstration :
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \sigma\tau u \in \mathcal{L}\}$
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \tau u \in \sigma^{-1}\mathcal{L}\}$

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

   Démonstration :
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \sigma\tau u \in \mathcal{L}\}$
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \tau u \in \sigma^{-1}\mathcal{L}\}$
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ u \in \tau^{-1}\left(\sigma^{-1}\mathcal{L}\right)\}$

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

   Démonstration :
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \sigma\tau u \in \mathcal{L}\}$
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \tau u \in \sigma^{-1}\mathcal{L}\}$
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ u \in \tau^{-1}\left(\sigma^{-1}\mathcal{L}\right)\}$

4. $\sigma^{-1}(\mathcal{L} \cup \mathcal{L}') = \sigma^{-1}\mathcal{L} \cup \sigma^{-1}\mathcal{L}'$

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

   Démonstration :

   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \sigma\tau u \in \mathcal{L}\}$

   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \tau u \in \sigma^{-1}\mathcal{L}\}$

   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ u \in \tau^{-1}\left(\sigma^{-1}\mathcal{L}\right)\}$

4. $\sigma^{-1}(\mathcal{L} \cup \mathcal{L}') = \sigma^{-1}\mathcal{L} \cup \sigma^{-1}\mathcal{L}'$

   Démonstration :

Automates

Propriétés

1. $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma \in \mathcal{L}$

   Démonstration :

   $\epsilon \in \sigma^{-1}\mathcal{L}$ si et seulement si $\sigma\epsilon = \sigma \in \mathcal{L}$

2. $\epsilon^{-1}\mathcal{L} = \mathcal{L}$

   Démonstration :

   $\epsilon^{-1}\mathcal{L} = \{\tau \in \mathcal{A}^* \ , \ \epsilon\tau \in \mathcal{L}\} = \{\tau \in \mathcal{A}^* \ , \ \tau \in \mathcal{L}\} = \mathcal{L}.$

3. $\tau^{-1}\left(\sigma^{-1}\mathcal{L}\right) = (\sigma\tau)^{-1}\mathcal{L}$

   Démonstration :
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \sigma\tau u \in \mathcal{L}\}$
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ \tau u \in \sigma^{-1}\mathcal{L}\}$
   $(\sigma\tau)^{-1}\mathcal{L} = \{u \in \mathcal{A}^* \ , \ u \in \tau^{-1}\left(\sigma^{-1}\mathcal{L}\right)\}$

4. $\sigma^{-1}(\mathcal{L} \cup \mathcal{L}') = \sigma^{-1}\mathcal{L} \cup \sigma^{-1}\mathcal{L}'$
   Démonstration :
   Double incusion.

Automates

Automates

Théorème

Automates

Théorème

On note D_i le langage de départ du sommet i.

Automates

Théorème

On note D_i le langage de départ du sommet i.

- Si σ est un mot et si $\delta(i, \sigma) = j$ alors $D_j = \sigma^{-1} D_i$.

Automates

Théorème

On note D_i le langage de départ du sommet i.

- Si σ est un mot et si $\delta(i, \sigma) = j$ alors $D_j = \sigma^{-1} D_i$.
  Signification : si σ fait passer le repère de l'état i à l'état j alors $D_j = \sigma^{-1} D_i$.

Automates

Théorème

On note D_i le langage de départ du sommet i.

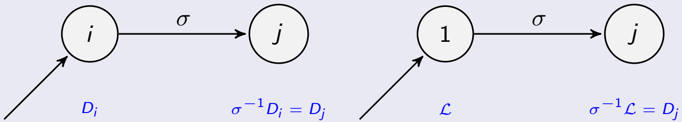
- Si σ est un mot et si $\delta(i,\sigma) = j$ alors $D_j = \sigma^{-1} D_i$.
  Signification : si σ fait passer le repère de l'état i à l'état j alors $D_j = \sigma^{-1} D_i$.
- En particulier, si j est l'état tel que $\sigma \in A_j$ alors $D_j = \sigma^{-1} D_1 = \sigma^{-1} \mathcal{L}$

Automates

Théorème

On note D_i le langage de départ du sommet i.

- Si σ est un mot et si $\delta(i,\sigma) = j$ alors $D_j = \sigma^{-1}D_i$.
  Signification : si σ fait passer le repère de l'état i à l'état j alors $D_j = \sigma^{-1}D_i$.
- En particulier, si j est l'état tel que $\sigma \in A_j$ alors $D_j = \sigma^{-1}D_1 = \sigma^{-1}\mathcal{L}$
  $\sigma^{-1}\mathcal{L} = D_j$



Automates

Détermination des résiduels du langage d'un automate : méthode 1

Automates

Détermination des résiduels du langage d'un automate : méthode 1

1. 1 étant l'état initial, on détermine l'état $i = \delta(1, \sigma)$,

Détermination des résiduels du langage d'un automate : méthode 1

1. 1 étant l'état initial, on détermine l'état $i = \delta(1, \sigma)$,
2. on précise le résiduel $\sigma^{-1}\mathcal{L} = D_i$ (suivre le mouvement du repère),

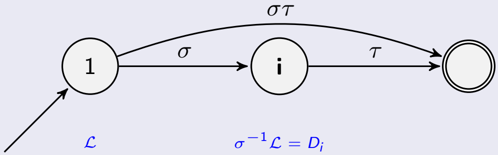
Automates

Détermination des résiduels du langage d'un automate : méthode 1

1. 1 étant l'état initial, on détermine l'état $i = \delta(1, \sigma)$,
2. on précise le résiduel $\sigma^{-1}\mathcal{L} = D_i$ (suivre le mouvement du repère),
3. on détermine une expression du résiduel en utilisant la méthode du départ.

Détermination des résiduels du langage d'un automate : méthode 1

1. 1 étant l'état initial, on détermine l'état $i = \delta(1, \sigma)$,
2. on précise le résiduel $\sigma^{-1}\mathcal{L} = D_i$ (suivre le mouvement du repère),
3. on détermine une expression du résiduel en utilisant la méthode du départ.



Automates

Détermination des résiduels du langage d'un automate : méthode 1

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.
Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

Automates

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

$Q(\mathcal{L}) \subset \{D_i\, , \; i \in \mathcal{E}\}$

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

$Q(\mathcal{L}) \subset \{D_i\,,\, i \in \mathcal{E}\}$

Si l'automate est accessible alors $\{D_i\,,\, i \in \mathcal{E}\} = Q(\mathcal{L})$

Automates

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

$Q(\mathcal{L}) \subset \{D_i \, , \, i \in \mathcal{E}\}$

Si l'automate est accessible alors $\{D_i \, , \, i \in \mathcal{E}\} = Q(\mathcal{L})$

Démonstration :

Automates

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

$Q(\mathcal{L}) \subset \{D_i \text{ , } i \in \mathcal{E}\}$

Si l'automate est accessible alors $\{D_i \text{ , } i \in \mathcal{E}\} = Q(\mathcal{L})$

Démonstration :

Soit $D_i \in \mathcal{D} = \{D_j \text{ , } j \in \mathcal{E}\}$

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

$Q\left(\mathcal{L}\right) \subset \left\{D_i\,,\; i \in \mathcal{E}\right\}$

Si l'automate est accessible alors $\left\{D_i\,,\; i \in \mathcal{E}\right\} = Q\left(\mathcal{L}\right)$

Démonstration :

Soit $D_i \in \mathcal{D} = \{D_j\,,\; j \in \mathcal{E}\}$

$\exists\, \sigma \in \mathcal{A}^* \;\; \delta(1, \sigma) = i.$

Automates

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

$Q(\mathcal{L}) \subset \{D_i\, ,\, i \in \mathcal{E}\}$

Si l'automate est accessible alors $\{D_i\, ,\, i \in \mathcal{E}\} = Q(\mathcal{L})$

Démonstration :

Soit $D_i \in \mathcal{D} = \{D_j\, ,\, j \in \mathcal{E}\}$

$\exists\, \sigma \in \mathcal{A}^* \;\; \delta(1, \sigma) = i$.

On en déduit $D_i = \sigma^{-1}\mathcal{L} \in Q(\mathcal{L})$.

Détermination des résiduels du langage d'un automate : méthode 1

Conséquence : Les résiduels du langage $\mathcal{L}$ d'un automate sont des langages de départ de l'automate.

Il y a donc un nombre fini de résiduels du langage $\mathcal{L}$ d'un automate.

$Q(\mathcal{L}) \subset \{D_i\,,\, i \in \mathcal{E}\}$

Si l'automate est accessible alors $\{D_i\,,\, i \in \mathcal{E}\} = Q(\mathcal{L})$

Démonstration :

Soit $D_i \in \mathcal{D} = \{D_j\,,\, j \in \mathcal{E}\}$

$\exists\, \sigma \in \mathcal{A}^* \;\; \delta(1, \sigma) = i$.

On en déduit $D_i = \sigma^{-1}\mathcal{L} \in Q(\mathcal{L})$.

Et donc $\{D_i\,,\, i \in \mathcal{E}\} \subset Q(\mathcal{L})$.

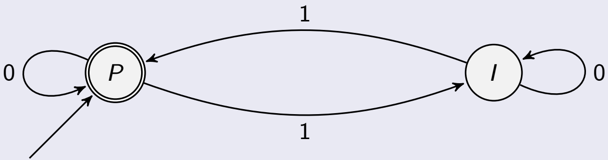
Automates

Automates

Exemple

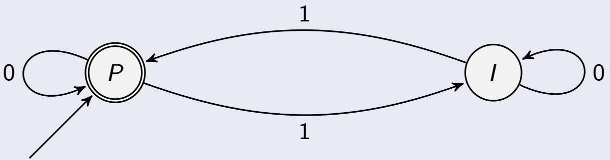
Automates

Exemple



Automates

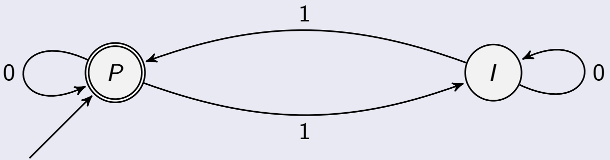
Exemple



- Résiduel de 0 :

Automates

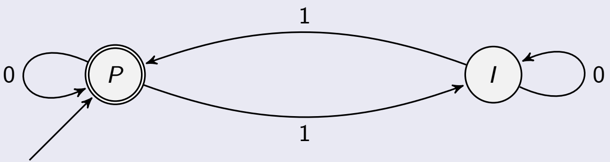
Exemple



- Résiduel de 0 :
  1. $\delta(P, 0) = P,$

Automates

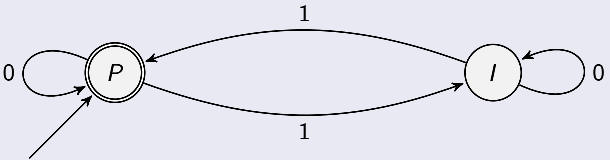
Exemple



- Résiduel de 0 :
  1. $\delta(P, 0) = P$,
  2. $0^{-1}\mathcal{L} = D_P$

Automates

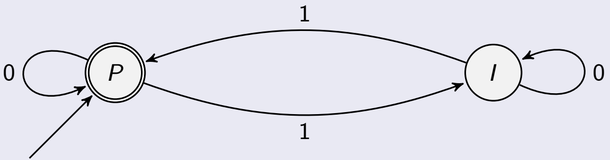
Exemple



- Résiduel de 0 :
  1. $\delta(P, 0) = P$,
  2. $0^{-1}\mathcal{L} = D_P = \mathcal{L} = (10^*1 \cup 0)^*$.

Automates

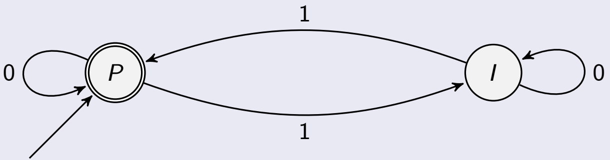
Exemple



- Résiduel de 0 :
  1. $\delta(P,0) = P$,
  2. $0^{-1}\mathcal{L} = D_P = \mathcal{L} = (10^*1 \cup 0)^*$.
- Résiduel de 1 :

Automates

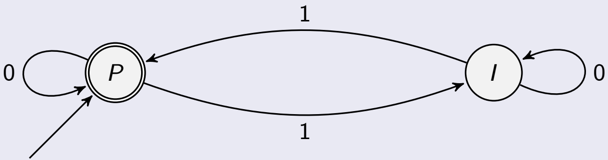
Exemple



- Résiduel de 0 :
  1. $\delta(P, 0) = P$,
  2. $0^{-1}\mathcal{L} = D_P = \mathcal{L} = (10^*1 \cup 0)^*$.
- Résiduel de 1 :
  1. $\delta(P, 1) = I$,

Automates

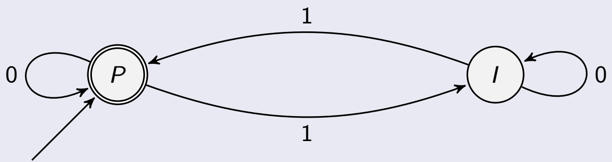
Exemple



- Résiduel de 0 :
  1. $\delta(P,0) = P$,
  2. $0^{-1}\mathcal{L} = D_P = \mathcal{L} = (10^*1 \cup 0)^*$.
- Résiduel de 1 :
  1. $\delta(P,1) = I$,
  2. $1^{-1}\mathcal{L} = D_I$

Automates

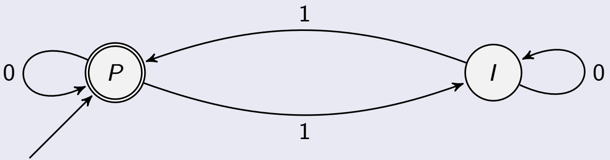
Exemple



- Résiduel de 0 :
  1. $\delta(P,0) = P$,
  2. $0^{-1}\mathcal{L} = D_P = \mathcal{L} = (10^*1 \cup 0)^*$.
- Résiduel de 1 :
  1. $\delta(P,1) = I$,
  2. $1^{-1}\mathcal{L} = D_I = 0^*1(10^*1 \cup 0)^*$.

Automates

Exemple



- Résiduel de 0 :
  1. $\delta(P, 0) = P$,
  2. $0^{-1}\mathcal{L} = D_P = \mathcal{L} = (10^*1 \cup 0)^*$.
- Résiduel de 1 :
  1. $\delta(P, 1) = I$,
  2. $1^{-1}\mathcal{L} = D_I = 0^*1(10^*1 \cup 0)^*$.

Ici $\{D_P\ ,\ D_I\} = Q\,(\mathcal{L})$

Automates

Automates

Détermination des résiduels : méthode 2 (algorithme de Nerode)

Automates

Détermination des résiduels : méthode 2 (algorithme de Nerode)

Dans l'ensemble $\mathcal{D}$ des langages de départ d'un automate, on note $\equiv_d$ la relation d'équivalence définie par $D_i \equiv_d D_j$ si les langages D_i et D_j ont les mêmes mots de longueur $0, 1, 2, ..., d$.

Détermination des résiduels : méthode 2 (algorithme de Nerode)

Dans l'ensemble $\mathcal{D}$ des langages de départ d'un automate, on note $\equiv_d$ la relation d'équivalence définie par $D_i \equiv_d D_j$ si les langages D_i et D_j ont les mêmes mots de longueur $0, 1, 2, ..., d$.
On pose $D_i^{(d)} = \{\sigma \in D_i \,,\, |\sigma| \leq d\}$.

Automates

Détermination des résiduels : méthode 2 (algorithme de Nerode)

Dans l'ensemble $\mathcal{D}$ des langages de départ d'un automate, on note $\equiv_d$ la relation d'équivalence définie par $D_i \equiv_d D_j$ si les langages D_i et D_j ont les mêmes mots de longueur $0, 1, 2, ..., d$.

On pose $D_i^{(d)} = \{\sigma \in D_i\, ,\, |\sigma| \leq d\}$.

Autre notation : $i \equiv_d j \Leftrightarrow D_i^{(d)} = D_j^{(d)}$

Automates

Détermination des résiduels : méthode 2 (algorithme de Nerode)

Dans l'ensemble $\mathcal{D}$ des langages de départ d'un automate, on note $\equiv_d$ la relation d'équivalence définie par $D_i \equiv_d D_j$ si les langages D_i et D_j ont les mêmes mots de longueur $0, 1, 2, ..., d$.
On pose $D_i^{(d)} = \{\sigma \in D_i\ ,\ |\sigma| \leq d\}$.
Autre notation : $i \equiv_d j \Leftrightarrow D_i^{(d)} = D_j^{(d)}$
Propriété : Si les classes d'équivalence sont les mêmes deux fois de suite pour d et $d+1$, $i \equiv_d j \Rightarrow D_i = D_j$.

Automates

Détermination des résiduels : méthode 2 (algorithme de Nerode)

Dans l'ensemble $\mathcal{D}$ des langages de départ d'un automate, on note $\equiv_d$ la relation d'équivalence définie par $D_i \equiv_d D_j$ si les langages D_i et D_j ont les mêmes mots de longueur $0, 1, 2, ..., d$.
On pose $D_i^{(d)} = \{\sigma \in D_i\ ,\ |\sigma| \leq d\}$.
Autre notation : $i \equiv_d j \Leftrightarrow D_i^{(d)} = D_j^{(d)}$
Propriété : Si les classes d'équivalence sont les mêmes deux fois de suite pour d et $d+1$, $i \equiv_d j \Rightarrow D_i = D_j$.

Anil Nerode, né en 1932 à Los Angeles, est un mathématicien américain.

Automates

Automates

Détermination des résiduels : méthode 2

Automates

Détermination des résiduels : méthode 2

1. On partage $\mathcal{D}$ en deux classes : les langages de départ qui contiennent ϵ et les autres ($d = 0$).

Détermination des résiduels : méthode 2

1. On partage $\mathcal{D}$ en deux classes : les langages de départ qui contiennent ϵ et les autres ($d = 0$).
2. Par ordre de longueur croissante, les différents mots séparent les langages de départ qui contiennent le mot et les autres.

Automates

Détermination des résiduels : méthode 2

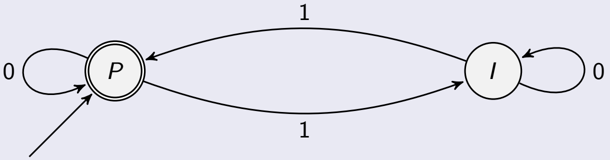
1. On partage $\mathcal{D}$ en deux classes : les langages de départ qui contiennent ϵ et les autres ($d = 0$).
2. Par ordre de longueur croissante, les différents mots séparent les langages de départ qui contiennent le mot et les autres.
3. Si les classes ne changent plus, on a le partage définitif en langages de départ égaux.

Automates

Méthode 2 : exemple

Automates

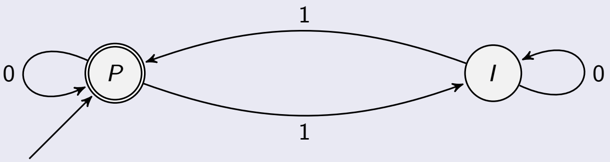
Méthode 2 : exemple



$\mathcal{D} = \{D_P, D_I\}$

Automates

Méthode 2 : exemple

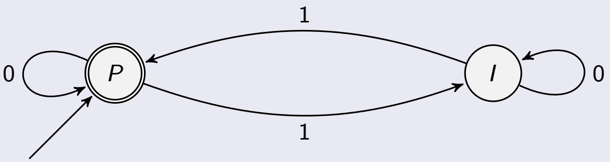


$\mathcal{D} = \{D_P, D_I\}$

d	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
D_P	ϵ	$\{D_P\}$	0	$\{D_P\}$	00,11	$\{D_P\}$
D_I		$\{D_I\}$	1	$\{D_I\}$	10,01	$\{D_I\}$

Automates

Méthode 2 : exemple



$\mathcal{D} = \{D_P, D_I\}$

d	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
D_P	ϵ	$\{D_P\}$	0	$\{D_P\}$	00,11	$\{D_P\}$
D_I		$\{D_I\}$	1	$\{D_I\}$	10,01	$\{D_I\}$

$\{\{D_P\}, \{D_I\}\}$ est une partition de $\mathcal{D}$.

Automates

Automates

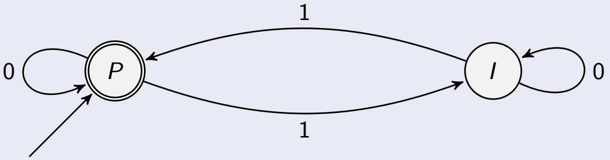
M2201
Graphes et langages

Département Informatique IUT de Saint-Dié

Méthode 2 : exemple, autre présentation

Automates

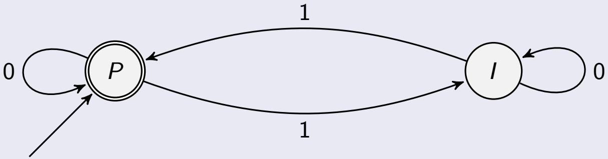
Méthode 2 : exemple, autre présentation



$\mathcal{D} = \{D_P, D_I\}$

Automates

Méthode 2 : exemple, autre présentation

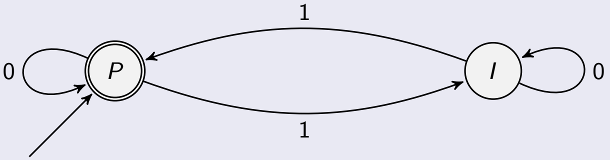


$\mathcal{D} = \{D_P, D_I\}$

d	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
P	ϵ	$\{P\}$	0	$\{P\}$	00,11	$\{P\}$
I		$\{I\}$	1	$\{I\}$	10,01	$\{I\}$

Automates

Méthode 2 : exemple, autre présentation



$\mathcal{D} = \{D_P, D_I\}$

d	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
P	ϵ	$\{P\}$	0	$\{P\}$	00,11	$\{P\}$
I		$\{I\}$	1	$\{I\}$	10,01	$\{I\}$

$\{\{D_P\}, \{D_I\}\}$ est une partition de $\mathcal{D}$.

Simplification

Automate minimal

Simplification

Automate minimal

Théorème : Un automate $\mathcal{A}ut$ étant donné, il existe un automate $\mathcal{M}$ ayant le même langage et dont le nombre d'états est minimal.

Simplification

Automate minimal

Théorème : Un automate $\mathcal{A}ut$ étant donné, il existe un automate $\mathcal{M}$ ayant le même langage et dont le nombre d'états est minimal.
Le nombre d'états de l'automate minimal $\mathcal{M}$ est le nombre de résiduels de l'automate.

Simplification

Construction de l'automate minimal : cas particulier

Simplification

Construction de l'automate minimal : cas particulier

Si l'automate présente autant de résiduels que d'états, l'automate est minimal.

Simplification

Construction de l'automate minimal : cas particulier

Si l'automate présente autant de résiduels que d'états, l'automate est minimal.

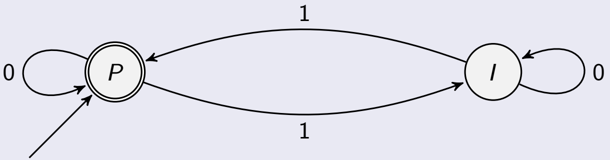
Exemple

Simplification

Construction de l'automate minimal : cas particulier

Si l'automate présente autant de résiduels que d'états, l'automate est minimal.

Exemple



Simplification

Construction de l'automate minimal

Simplification

Construction de l'automate minimal

1. On élimine tous les états inaccessibles.

Simplification

Construction de l'automate minimal

1. On élimine tous les états inaccessibles.
2. On calcule les langages de départ de chaque état accessible (il s'agit du langage de l'automate et des résiduels).

Simplification

Construction de l'automate minimal

1. On élimine tous les états inaccessibles.
2. On calcule les langages de départ de chaque état accessible (il s'agit du langage de l'automate et des résiduels).
3. L'automate minimal est défini par :

Simplification

Construction de l'automate minimal

1. On élimine tous les états inaccessibles.
2. On calcule les langages de départ de chaque état accessible (il s'agit du langage de l'automate et des résiduels).
3. L'automate minimal est défini par :
   - ses états sont les résiduels du langage de l'automate,

Simplification

Construction de l'automate minimal

1. On élimine tous les états inaccessibles.
2. On calcule les langages de départ de chaque état accessible (il s'agit du langage de l'automate et des résiduels).
3. L'automate minimal est défini par :
   - ses états sont les résiduels du langage de l'automate,
   - l'état initial est le langage de l'automate,

Simplification

Construction de l'automate minimal

1. On élimine tous les états inaccessibles.
2. On calcule les langages de départ de chaque état accessible (il s'agit du langage de l'automate et des résiduels).
3. L'automate minimal est défini par :
   - ses états sont les résiduels du langage de l'automate,
   - l'état initial est le langage de l'automate,
   - ses états acceptants sont les langages de départ des états acceptants de l'automate,

Simplification

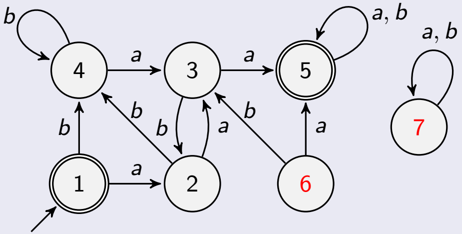
Construction de l'automate minimal

1. On élimine tous les états inaccessibles.
2. On calcule les langages de départ de chaque état accessible (il s'agit du langage de l'automate et des résiduels).
3. L'automate minimal est défini par :
   - ses états sont les résiduels du langage de l'automate,
   - l'état initial est le langage de l'automate,
   - ses états acceptants sont les langages de départ des états acceptants de l'automate,
   - les transitions sont définies par $\delta(D, \sigma) = \sigma^{-1} D$

Simplification

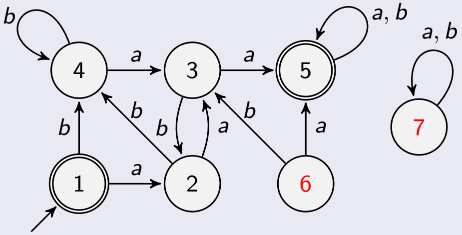
Construction de l'automate minimal : exemple

Construction de l'automate minimal : exemple



Simplification

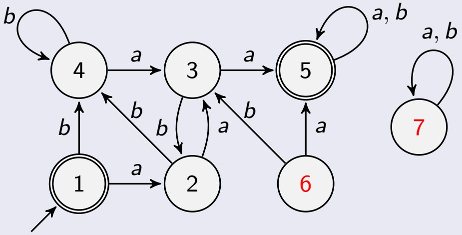
Construction de l'automate minimal : exemple



d	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
1	ϵ	$\{1,5\}$		$\{1\}$		$\{1\}$
2		$\{2,3,4\}$		$\{2,4\}$	aa	$\{2,4\}$
3		$\{2,3,4\}$	a	$\{3\}$	$aa,\ ab$	$\{3\}$
4		$\{2,3,4\}$		$\{2,4\}$	aa	$\{2,4\}$
5	ϵ	$\{1,5\}$	$a,\ b$	$\{5\}$	$aa,\ ab,\ ba,\ bb$	$\{5\}$

Simplification

Construction de l'automate minimal : exemple

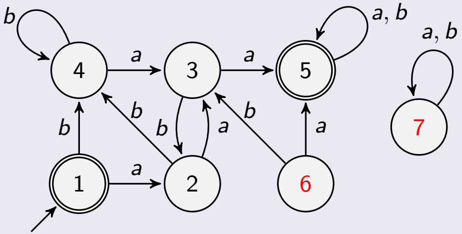


d	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
1	ϵ	$\{1, 5\}$		$\{1\}$		$\{1\}$
2		$\{2, 3, 4\}$		$\{2, 4\}$	aa	$\{2, 4\}$
3		$\{2, 3, 4\}$	a	$\{3\}$	aa, ab	$\{3\}$
4		$\{2, 3, 4\}$		$\{2, 4\}$	aa	$\{2, 4\}$
5	ϵ	$\{1, 5\}$	a, b	$\{5\}$	aa, ab, ba, bb	$\{5\}$

$\{\{D_1\}, \{D_2, D_4\}, \{D_3\}, \{D_5\}\}$ est une partition de $\mathcal{D}$.

Simplification

Construction de l'automate minimal : exemple



d	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
1	ϵ	$\{1,5\}$		$\{1\}$		$\{1\}$
2		$\{2,3,4\}$		$\{2,4\}$	aa	$\{2,4\}$
3		$\{2,3,4\}$	a	$\{3\}$	aa, ab	$\{3\}$
4		$\{2,3,4\}$		$\{2,4\}$	aa	$\{2,4\}$
5	ϵ	$\{1,5\}$	a, b	$\{5\}$	aa, ab, ba, bb	$\{5\}$

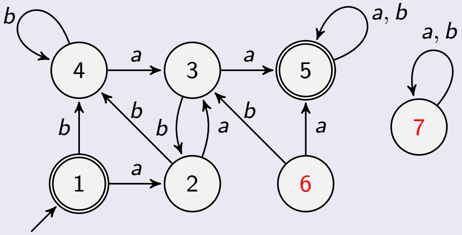
$\{\{D_1\}, \{D_2, D_4\}, \{D_3\}, \{D_5\}\}$ est une partition de $\mathcal{D}$.

Les classes sont notées $\{1\}$, $\{2,4\}$, $\{3\}$, $\{5\}$

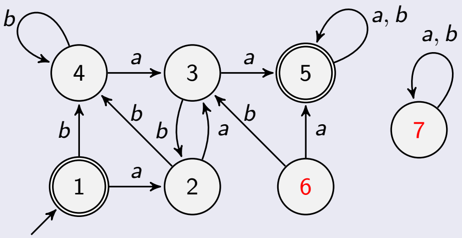
Construction de l'automate minimal : exemple

Simplification

Construction de l'automate minimal : exemple



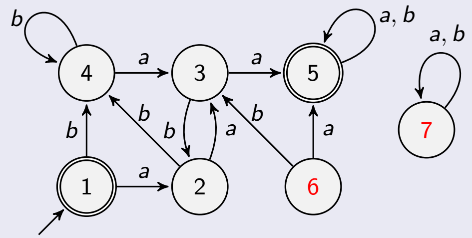
Construction de l'automate minimal : exemple



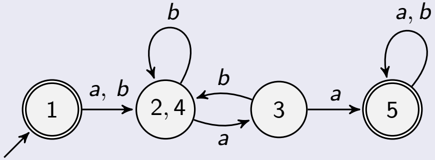
Classes : $\{1\}$, $\{2,4\}$, $\{3\}$, $\{5\}$

Simplification

Construction de l'automate minimal : exemple



Classes : $\{1\}, \{2,4\}, \{3\}, \{5\}$ Automate minimal :

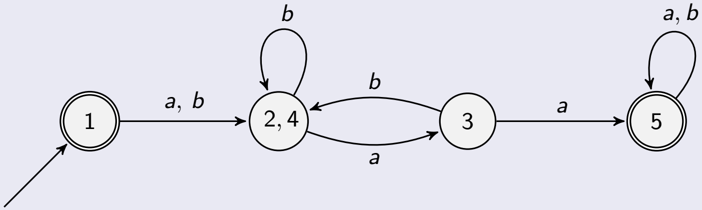


Simplification

Langage de l'automate minimal : méthode du départ

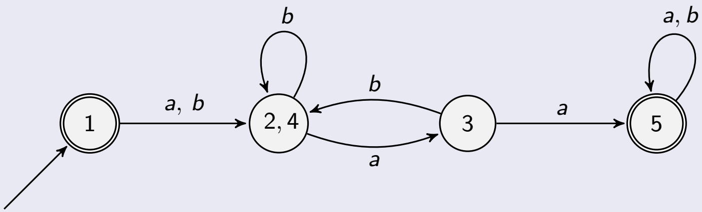
Simplification

Langage de l'automate minimal : méthode du départ



Simplification

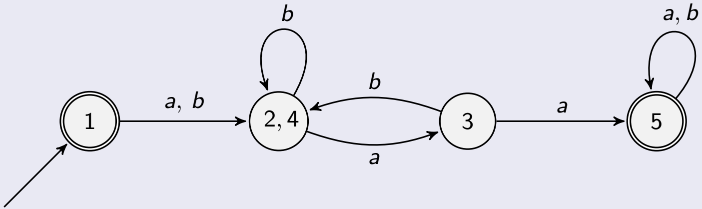
Langage de l'automate minimal : méthode du départ



$$\begin{cases} D_1 = \epsilon \cup aD_2 \cup bD_2 \\ D_2 = bD_2 \cup aD_3 \\ D_3 = bD_2 \cup aD_5 \\ D_5 = \epsilon \cup \{a, b\} D_5 \end{cases}$$

Simplification

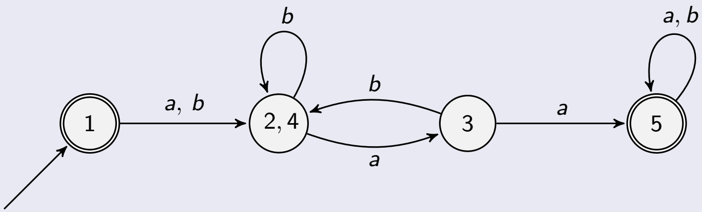
Langage de l'automate minimal : méthode du départ



$$\begin{cases} D_1 = \epsilon \cup aD_2 \cup bD_2 \\ D_2 = bD_2 \cup aD_3 \\ D_3 = bD_2 \cup aD_5 \\ D_5 = \epsilon \cup \{a, b\} D_5 \end{cases} \qquad \begin{cases} D_5 = \{a, b\}^* \epsilon = \{a, b\}^* \\ D_3 = bD_2 \cup a\{a, b\}^* \\ D_2 = bD_2 \cup a\left(bD_2 \cup a\{a, b\}^*\right) \\ D_1 = \epsilon \cup \{a, b\} D_2 \end{cases}$$

Simplification

Langage de l'automate minimal : méthode du départ

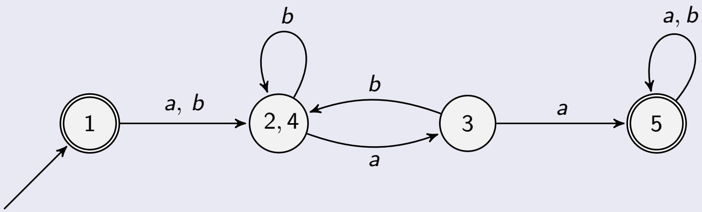


$$\begin{cases} D_1 = \epsilon \cup aD_2 \cup bD_2 \\ D_2 = bD_2 \cup aD_3 \\ D_3 = bD_2 \cup aD_5 \\ D_5 = \epsilon \cup \{a,b\}D_5 \end{cases} \qquad \begin{cases} D_5 = \{a,b\}^*\epsilon = \{a,b\}^* \\ D_3 = bD_2 \cup a\{a,b\}^* \\ D_2 = bD_2 \cup a\left(bD_2 \cup a\{a,b\}^*\right) \\ D_1 = \epsilon \cup \{a,b\}D_2 \end{cases}$$

$$\begin{cases} D_2 = (b \cup ab)\, D_2 \cup a^2\{a,b\}^* \\ D_1 = \epsilon \cup \{a,b\}D_2 \\ D_3 = bD_2 \cup a\{a,b\}^* \\ D_5 = \{a,b\}^*\epsilon = \{a,b\}^* \end{cases}$$

Simplification

Langage de l'automate minimal : méthode du départ



$$\begin{cases} D_1 = \epsilon \cup aD_2 \cup bD_2 \\ D_2 = bD_2 \cup aD_3 \\ D_3 = bD_2 \cup aD_5 \\ D_5 = \epsilon \cup \{a, b\} D_5 \end{cases} \qquad \begin{cases} D_5 = \{a, b\}^* \epsilon = \{a, b\}^* \\ D_3 = bD_2 \cup a\{a, b\}^* \\ D_2 = bD_2 \cup a\left(bD_2 \cup a\{a, b\}^*\right) \\ D_1 = \epsilon \cup \{a, b\} D_2 \end{cases}$$

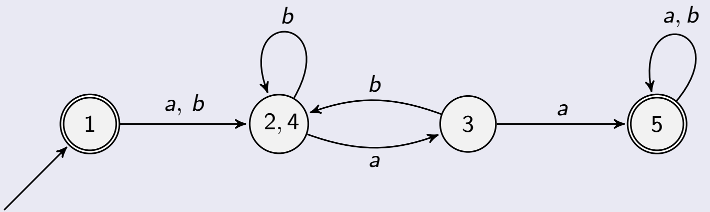
$$\begin{cases} D_2 = (b \cup ab)\, D_2 \cup a^2\{a, b\}^* \\ D_1 = \epsilon \cup \{a, b\} D_2 \\ D_3 = bD_2 \cup a\{a, b\}^* \\ D_5 = \{a, b\}^* \epsilon = \{a, b\}^* \end{cases} \qquad \begin{cases} D_2 = (b \cup ab)^*\, a^2\{a, b\}^* \\ D_1 = \epsilon \cup \{a, b\}\, \{b, ab\}^*\, a^2\{a, b\}^* \\ D_3 = b\, \{b, ab\}^*\, a^2\{a, b\}^* \cup a\{a, b\}^* \\ D_5 = \{a, b\}^* \end{cases}$$

Simplification

Langage de l'automate minimal : méthode du départ

Simplification

Langage de l'automate minimal : méthode du départ



$$\begin{cases} \mathcal{L} = D_1 = \epsilon \cup \{a, b\} \{b, ab\}^* \, a^2 \{a, b\}^* = \epsilon \cup \mathcal{A} \{b, ab\}^* \, a^2 \mathcal{A}^* \\ D_2 = \{b, ab\}^* \, a^2 \mathcal{A}^* = a^{-1} \mathcal{L} = b^{-1} \mathcal{L} \\ D_3 = b \{b, ab\}^* \, a^2 \mathcal{A}^* \cup a\mathcal{A}^* = (aa)^{-1} \mathcal{L} \\ D_5 = \mathcal{A}^* = (aaa)^{-1} \mathcal{L} \end{cases}$$

Simplification

M2201 Graphes et langages

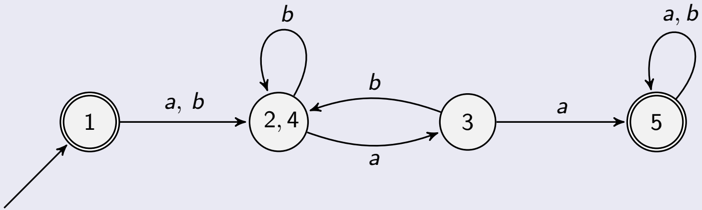
Département Informatique IUT de Saint-Dié

Graphes
- Généralités
- Arbre
- Cycle eulérien, hamiltonien
- Coloration
- Le problème du plus court chemin
- Arbre recouvrant de poids minimal

Langages et automates
- Langages
- Automates

Annexes

Langage de l'automate minimal : méthode du départ



$$\begin{cases} \mathcal{L} = D_1 = \epsilon \cup \{a, b\}\,\{b, ab\}^*\, a^2\{a, b\}^* = \epsilon \cup \mathcal{A}\,\{b, ab\}^*\, a^2\mathcal{A}^* \\ D_2 = \{b, ab\}^*\, a^2\mathcal{A}^* = a^{-1}\mathcal{L} = b^{-1}\mathcal{L} \\ D_3 = b\,\{b, ab\}^*\, a^2\mathcal{A}^* \cup a\mathcal{A}^* = (aa)^{-1}\,\mathcal{L} \\ D_5 = \mathcal{A}^* = (aaa)^{-1}\,\mathcal{L} \end{cases}$$

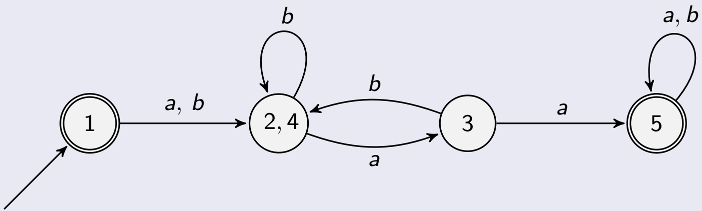
$$Q\,(\mathcal{L}) = \{\mathcal{L}\,, a^{-1}\mathcal{L}\,, aa^{-1}\mathcal{L}\,, aaa^{-1}\mathcal{L}\}$$

Simplification

Langage de l'automate minimal : méthode de l'arrivée

Simplification

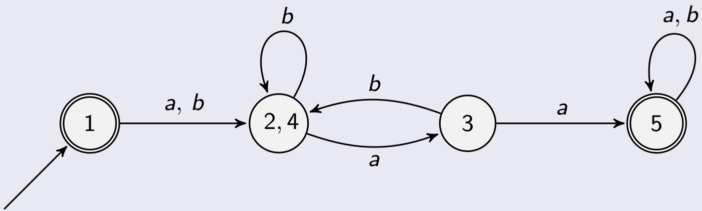
Langage de l'automate minimal : méthode de l'arrivée



$$\begin{cases} A_1 = \epsilon \\ A_2 = A_1\{a,b\} \cup A_2 b \cup A_3 b \\ A_3 = A_2 a \\ A_5 = A_3 a \cup A_5\{a,b\} \end{cases}$$

Simplification

Langage de l'automate minimal : méthode de l'arrivée



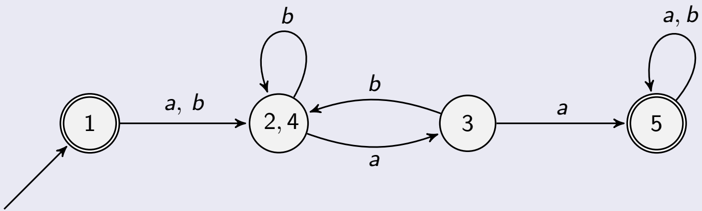
$$\begin{cases} A_1 = \epsilon \\ A_2 = A_1\{a,b\} \cup A_2 b \cup A_3 b \\ A_3 = A_2 a \\ A_5 = A_3 a \cup A_5\{a,b\} \end{cases} \qquad \begin{cases} A_2 = \epsilon\{a,b\} \cup A_2 b \cup A_2 ab \\ A_3 = A_2 a \\ A_5 = A_3 a\{a,b\}^* \\ A_1 = \epsilon \end{cases}$$

Simplification

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Langage de l'automate minimal : méthode de l'arrivée



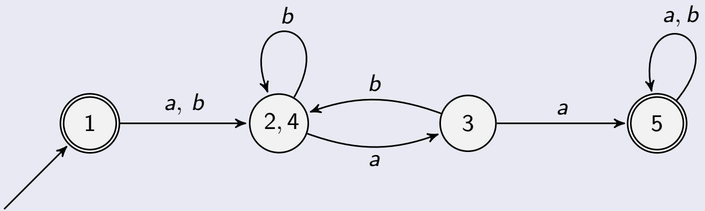
$$\begin{cases} A_1 = \epsilon \\ A_2 = A_1\{a,b\} \cup A_2 b \cup A_3 b \\ A_3 = A_2 a \\ A_5 = A_3 a \cup A_5\{a,b\} \end{cases}$$

$$\begin{cases} A_2 = A_2\,(b \cup ab) \cup \{a,b\} \\ A_3 = A_2 a \\ A_5 = A_3 a\{a,b\}^* \\ A_1 = \epsilon \end{cases}$$

$$\begin{cases} A_2 = \epsilon\{a,b\} \cup A_2 b \cup A_2 ab \\ A_3 = A_2 a \\ A_5 = A_3 a\{a,b\}^* \\ A_1 = \epsilon \end{cases}$$

Simplification

Langage de l'automate minimal : méthode de l'arrivée

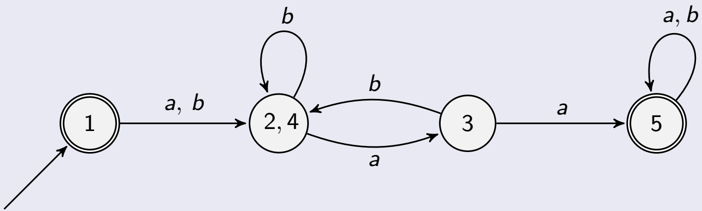


$$\begin{cases} A_1 = \epsilon \\ A_2 = A_1\{a,b\} \cup A_2 b \cup A_3 b \\ A_3 = A_2 a \\ A_5 = A_3 a \cup A_5\{a,b\} \end{cases} \quad \begin{cases} A_2 = \epsilon\{a,b\} \cup A_2 b \cup A_2 ab \\ A_3 = A_2 a \\ A_5 = A_3 a\{a,b\}^* \\ A_1 = \epsilon \end{cases}$$

$$\begin{cases} A_2 = A_2\,(b \cup ab) \cup \{a,b\} \\ A_3 = A_2 a \\ A_5 = A_3 a\{a,b\}^* \\ A_1 = \epsilon \end{cases} \quad \begin{cases} A_2 = \{a,b\}\,(b \cup ab)^* \\ A_3 = \{a,b\}\,(b \cup ab)^*\, a \\ A_5 = \{a,b\}\,(b \cup ab)^*\, a^2\{a,b\}^* \\ A_1 = \epsilon \end{cases}$$

Simplification

Langage de l'automate minimal : méthode de l'arrivée



$$\begin{cases} A_1 = \epsilon \\ A_2 = A_1\{a,b\} \cup A_2 b \cup A_3 b \\ A_3 = A_2 a \\ A_5 = A_3 a \cup A_5\{a,b\} \end{cases} \qquad \begin{cases} A_2 = \epsilon\{a,b\} \cup A_2 b \cup A_2 ab \\ A_3 = A_2 a \\ A_5 = A_3 a\{a,b\}^* \\ A_1 = \epsilon \end{cases}$$

$$\begin{cases} A_2 = A_2\,(b \cup ab) \cup \{a,b\} \\ A_3 = A_2 a \\ A_5 = A_3 a\{a,b\}^* \\ A_1 = \epsilon \end{cases} \qquad \begin{cases} A_2 = \{a,b\}\,(b \cup ab)^* \\ A_3 = \{a,b\}\,(b \cup ab)^*\, a \\ A_5 = \{a,b\}\,(b \cup ab)^*\, a^2\{a,b\}^* \\ A_1 = \epsilon \end{cases}$$

On obtient $\mathcal{L} = A_1 \cup A_5 = \epsilon \cup \{a,b\}\,(b \cup ab)^*\, a^2\{a,b\}^*$

Automates

Automates

Exemples : remarques

Automates

Exemples : remarques

- Les deux méthodes conduisent bien au même langage de l'automate $\mathcal{L}$.

Automates

Exemples : remarques

- Les deux méthodes conduisent bien au même langage de l'automate $\mathcal{L}$.
- Le passage par l'automate minimal peut rendre plus facile la détermination du langage de l'automate.

Automates

Exemples : remarques

- Les deux méthodes conduisent bien au même langage de l'automate $\mathcal{L}$.
- Le passage par l'automate minimal peut rendre plus facile la détermination du langage de l'automate.
- On peut préciser $D_6 = aD_4 \cup bD_3 = a\{b, ab\}^* a^2\{a, b\}^* \cup b^2 \{b, ab\}^* a^2\{a, b\}^* \cup a\{a, b\}^*$

Exemples : remarques

- Les deux méthodes conduisent bien au même langage de l'automate $\mathcal{L}$.
- Le passage par l'automate minimal peut rendre plus facile la détermination du langage de l'automate.
- On peut préciser $D_6 = aD_4 \cup bD_3 =$ $a\{b, ab\}^* a^2\{a, b\}^* \cup b^2 \{b, ab\}^* a^2\{a, b\}^* \cup a\{a, b\}^*$ et $D_7 = \{a, b\}D_7 = \{a, b\}^*\emptyset = \emptyset$.

Automates

Exemples : remarques

- Les deux méthodes conduisent bien au même langage de l'automate $\mathcal{L}$.
- Le passage par l'automate minimal peut rendre plus facile la détermination du langage de l'automate.
- On peut préciser $D_6 = aD_4 \cup bD_3 =$
  $a\{b, ab\}^* a^2\{a, b\}^* \cup b^2\{b, ab\}^* a^2\{a, b\}^* \cup a\{a, b\}^*$
  et $D_7 = \{a, b\}D_7 = \{a, b\}^*\emptyset = \emptyset$.
  De manière générale, les langages de départ des états inaccessibles dépendent des états et transitions associés.
- $A_6 = A_7 = \emptyset$ (états inaccessibles).

Automates

Automates

Langage régulier

Automates

Langage régulier

Théorème : Le langage accepté par un automate fini est régulier.

Automates

Langage régulier

Théorème : Le langage accepté par un automate fini est régulier.

Démonstration : la résolution des systèmes des méthodes du départ ou de l'arrivée avec le lemme d'Arden, conduit à une expression régulière décrivant le langage de l'automate.

Automates

Langage régulier

Théorème : Le langage accepté par un automate fini est régulier.

Démonstration : la résolution des systèmes des méthodes du départ ou de l'arrivée avec le lemme d'Arden, conduit à une expression régulière décrivant le langage de l'automate.

Exemple de langage qui n'est pas le langage d'un automate

Automates

Langage régulier

Théorème : Le langage accepté par un automate fini est régulier.

Démonstration : la résolution des systèmes des méthodes du départ ou de l'arrivée avec le lemme d'Arden, conduit à une expression régulière décrivant le langage de l'automate.

Exemple de langage qui n'est pas le langage d'un automate

En notant $\mathcal{A} = \{a, b\}$, $\mathcal{L} = \{a^k b^k,\ k \in \mathbb{N}\}$ n'est pas le langage d'un automate fini.

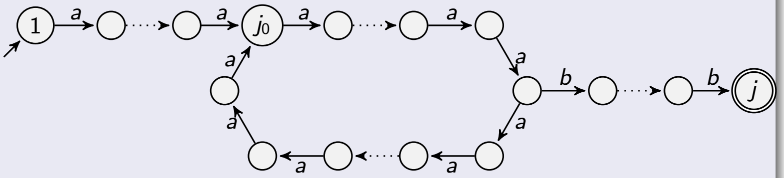
Automates

Automates

$\mathcal{L} = \{a^k b^k,\ k \in \mathbb{N}\}$

Automates

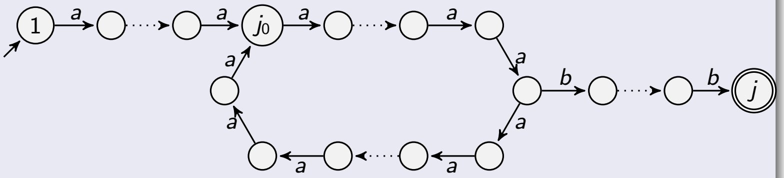
$\mathcal{L} = \{a^k b^k, \ k \in \mathbb{N}\}$



Supposons que $\mathcal{A}ut$ soit un automate à n états de langage $\mathcal{L} = \{a^k b^k, \ k \geq 0\}$.

Automates

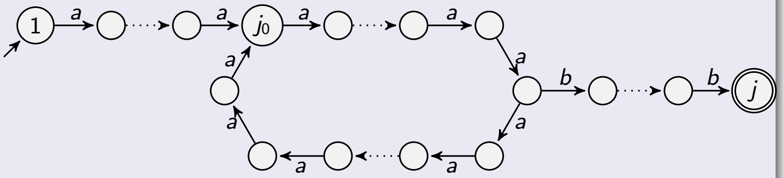
$\mathcal{L} = \{a^k b^k,\ k \in \mathbb{N}\}$



Supposons que $\mathcal{A}ut$ soit un automate à n états de langage $\mathcal{L} = \{a^k b^k,\ k \geq 0\}$.
Soit $\sigma = a^{n+1} b^{n+1}$, tel que $\delta(1, \sigma) = j$ acceptant.

Automates

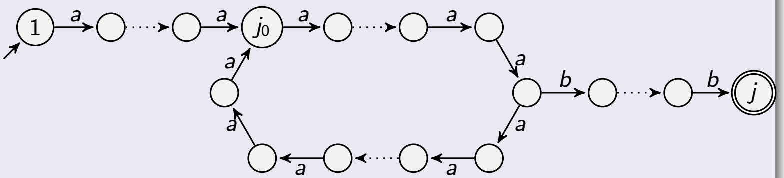
$\mathcal{L} = \{a^k b^k,\ k \in \mathbb{N}\}$



Supposons que $\mathcal{A}ut$ soit un automate à n états de langage $\mathcal{L} = \{a^k b^k,\ k \geq 0\}$.
Soit $\sigma = a^{n+1} b^{n+1}$, tel que $\delta(1, \sigma) = j$ acceptant.
$\exists m\,,\ 1 \leq m \leq n\,,\ \exists j_0\ \delta(j_0, a^m) = j_0$ (boucle).

Automates

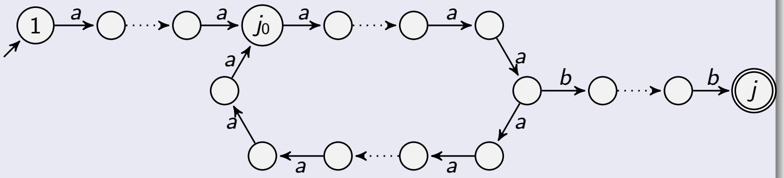
$\mathcal{L} = \{a^k b^k,\ k \in \mathbb{N}\}$



Supposons que $\mathcal{A}ut$ soit un automate à n états de langage $\mathcal{L} = \{a^k b^k,\ k \geq 0\}$.
Soit $\sigma = a^{n+1} b^{n+1}$, tel que $\delta(1, \sigma) = j$ acceptant.
$\exists m\,,\ 1 \leq m \leq n\,,\ \exists j_0\ \delta(j_0, a^m) = j_0$ (boucle).
On a alors $\delta(1, a^{m+n+1} b^{n+1}) = j$ et $a^{m+n+1} b^{n+1}$ est reconnu par $\mathcal{A}ut$.

Automates

$\mathcal{L} = \{a^k b^k,\ k \in \mathbb{N}\}$



Supposons que $\mathcal{A}ut$ soit un automate à n états de langage
$\mathcal{L} = \{a^k b^k,\ k \geq 0\}$.
Soit $\sigma = a^{n+1} b^{n+1}$, tel que $\delta(1, \sigma) = j$ acceptant.
$\exists m\,,\ 1 \leq m \leq n\,,\ \exists j_0\ \delta(j_0, a^m) = j_0$ (boucle).
On a alors $\delta(1, a^{m+n+1} b^{n+1}) = j$ et $a^{m+n+1} b^{n+1}$ est reconnu par $\mathcal{A}ut$.
Contradiction : $a^{m+n+1} b^{n+1} \notin \mathcal{L} = \{a^n b^n,\ n \geq 0\}$.

Automates

Automates

Langage régulier : caractérisation

Langage régulier : caractérisation

Théorème de Kleene : Tout langage régulier est le langage d'un automate fini.

Automates

Langage régulier : caractérisation

Théorème de Kleene : Tout langage régulier est le langage d'un automate fini.
On a alors : un langage peut être décrit par une expression régulière (ou rationnelle) si et seulement si il est reconnu par un automate fini.

Automates

Langage régulier : caractérisation

Théorème de Kleene : Tout langage régulier est le langage d'un automate fini.
On a alors : un langage peut être décrit par une expression régulière (ou rationnelle) si et seulement si il est reconnu par un automate fini.

Stephen Cole Kleene (1909-1994) : mathématicien américain.

Automates

Automates

Automate associé à un langage régulier

Automates

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.

Automates

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.
Exemple : $\mathcal{L} = \left(a \cup b^2\right)^* b$.

Automates

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.
Exemple : $\mathcal{L} = \left(a \cup b^2\right)^* b$.
$R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}$

Automates

M2201 Graphes et langages

Département Informatique IUT de Saint-Dié

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.
Exemple : $\mathcal{L} = \left(a \cup b^2\right)^* b$.
$R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}$
$\mathcal{L} = \left(a \cup b^2\right)^* b = \left(\epsilon \cup \left(a \cup b^2\right)\left(a \cup b^2\right)^*\right) b$

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.
Exemple : $\mathcal{L} = \left(a \cup b^2\right)^* b$.
$R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}$
$\mathcal{L} = \left(a \cup b^2\right)^* b = \left(\epsilon \cup \left(a \cup b^2\right)\left(a \cup b^2\right)^*\right) b$
$\mathcal{L} = b \cup a\left(a \cup b^2\right)^* b \cup b^2\left(a \cup b^2\right)^* b$

Automates

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.

Exemple : $\mathcal{L} = \left(a \cup b^2\right)^* b$.

$R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}$

$\mathcal{L} = \left(a \cup b^2\right)^* b = \left(\epsilon \cup \left(a \cup b^2\right)\left(a \cup b^2\right)^*\right) b$

$\mathcal{L} = b \cup a\left(a \cup b^2\right)^* b \cup b^2\left(a \cup b^2\right)^* b$

$\mathcal{L} = a\mathcal{L} \cup b \cup b^2\mathcal{L}$

Automates

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.

Exemple : $\mathcal{L} = \left(a \cup b^2\right)^* b$.

$R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}$

$\mathcal{L} = \left(a \cup b^2\right)^* b = \left(\epsilon \cup \left(a \cup b^2\right)\left(a \cup b^2\right)^*\right) b$

$\mathcal{L} = b \cup a\left(a \cup b^2\right)^* b \cup b^2\left(a \cup b^2\right)^* b$

$\mathcal{L} = a\mathcal{L} \cup b \cup b^2\mathcal{L}$

$\mathcal{L} = a\mathcal{L} \cup \left(b \cup b^2\mathcal{L}\right)$

Automate associé à un langage régulier

La détermination des résiduels d'un langage permet d'obtenir l'automate minimal du langage.

Exemple : $\mathcal{L} = \left(a \cup b^2\right)^* b$.

$R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}$

$\mathcal{L} = \left(a \cup b^2\right)^* b = \left(\epsilon \cup \left(a \cup b^2\right)\left(a \cup b^2\right)^*\right) b$

$\mathcal{L} = b \cup a\left(a \cup b^2\right)^* b \cup b^2\left(a \cup b^2\right)^* b$

$\mathcal{L} = a\mathcal{L} \cup b \cup b^2\mathcal{L}$

$\mathcal{L} = a\mathcal{L} \cup \left(b \cup b^2\mathcal{L}\right)$

$\mathcal{L} = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

Automates

Automates

Automate associé à un langage régulier : exemple

Automate associé à un langage régulier : exemple

$$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$$

Automate associé à un langage régulier : exemple

$$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$$

$$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$$

Automate associé à un langage régulier : exemple

$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\mathcal{L} = \mathcal{L}$

Automate associé à un langage régulier : exemple

$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\mathcal{L} = \mathcal{L}$

$b^{-1}R_1 = b^{-1}\mathcal{L} = \epsilon \cup b\mathcal{L}$

Automates

Automate associé à un langage régulier : exemple

$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\mathcal{L} = \mathcal{L}$

$b^{-1}R_1 = b^{-1}\mathcal{L} = \epsilon \cup b\mathcal{L}$

$\boxed{R_2 = \epsilon \cup b\mathcal{L}}$

Automates

Automate associé à un langage régulier : exemple

$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\mathcal{L} = \mathcal{L}$

$b^{-1}R_1 = b^{-1}\mathcal{L} = \epsilon \cup b\mathcal{L}$

$\boxed{R_2 = \epsilon \cup b\mathcal{L}}$

$a^{-1}R_2 = a^{-1}\left(\epsilon \cup b\mathcal{L}\right) = \emptyset$

Automate associé à un langage régulier : exemple

$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\mathcal{L} = \mathcal{L}$

$b^{-1}R_1 = b^{-1}\mathcal{L} = \epsilon \cup b\mathcal{L}$

$\boxed{R_2 = \epsilon \cup b\mathcal{L}}$

$a^{-1}R_2 = a^{-1}\left(\epsilon \cup b\mathcal{L}\right) = \emptyset$

$b^{-1}R_2 = b^{-1}\left(\epsilon \cup b\mathcal{L}\right) = \mathcal{L}$

Automates

Automate associé à un langage régulier : exemple

$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\mathcal{L} = \mathcal{L}$

$b^{-1}R_1 = b^{-1}\mathcal{L} = \epsilon \cup b\mathcal{L}$

$\boxed{R_2 = \epsilon \cup b\mathcal{L}}$

$a^{-1}R_2 = a^{-1}\left(\epsilon \cup b\mathcal{L}\right) = \emptyset$

$b^{-1}R_2 = b^{-1}\left(\epsilon \cup b\mathcal{L}\right) = \mathcal{L}$

$\boxed{R_3 = \emptyset}$

Automate associé à un langage régulier : exemple

$\mathcal{L} = \left(a \cup b^2\right)^* b = a\mathcal{L} \cup b\left(\epsilon \cup b\mathcal{L}\right)$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\mathcal{L} = \mathcal{L}$

$b^{-1}R_1 = b^{-1}\mathcal{L} = \epsilon \cup b\mathcal{L}$

$\boxed{R_2 = \epsilon \cup b\mathcal{L}}$

$a^{-1}R_2 = a^{-1}\left(\epsilon \cup b\mathcal{L}\right) = \emptyset$

$b^{-1}R_2 = b^{-1}\left(\epsilon \cup b\mathcal{L}\right) = \mathcal{L}$

$\boxed{R_3 = \emptyset}$

$a^{-1}R_3 = b^{-1}R_3 = \emptyset$

Automates

Automates

Automate associé à un langage régulier : exemple

Automate associé à un langage régulier : exemple

$R_1 = \mathcal{L} = \left(a \cup b^2\right)^* b$; $R_2 = \epsilon \cup b\mathcal{L}$; $R_3 = \emptyset$.

Automates

Automate associé à un langage régulier : exemple

$R_1 = \mathcal{L} = \left(a \cup b^2\right)^* b$; $R_2 = \epsilon \cup b\mathcal{L}$; $R_3 = \emptyset$.
$a^{-1}R_1 = R_1$; $b^{-1}R_1 = R_2$

Automates

Automate associé à un langage régulier : exemple

$R_1 = \mathcal{L} = \left(a \cup b^2\right)^* b$; $R_2 = \epsilon \cup b\mathcal{L}$; $R_3 = \emptyset$.
$a^{-1}R_1 = R_1$; $b^{-1}R_1 = R_2$
$a^{-1}R_2 = R_3$; $b^{-1}R_2 = R_1$

Automates

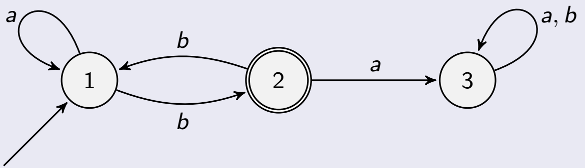
Automate associé à un langage régulier : exemple

$R_1 = \mathcal{L} = \left(a \cup b^2\right)^* b$; $R_2 = \epsilon \cup b\mathcal{L}$; $R_3 = \emptyset$.
$a^{-1}R_1 = R_1$; $b^{-1}R_1 = R_2$
$a^{-1}R_2 = R_3$; $b^{-1}R_2 = R_1$
$a^{-1}R_3 = b^{-1}R_3 = R_3$

Automates

Automate associé à un langage régulier : exemple

$R_1 = \mathcal{L} = \left(a \cup b^2\right)^* b$; $R_2 = \epsilon \cup b\mathcal{L}$; $R_3 = \emptyset$.
$a^{-1}R_1 = R_1$; $b^{-1}R_1 = R_2$
$a^{-1}R_2 = R_3$; $b^{-1}R_2 = R_1$
$a^{-1}R_3 = b^{-1}R_3 = R_3$



Automates

Automates

Automate associé à un langage régulier : exemple 2

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\,(\epsilon \cup ba\mathcal{L}) = \emptyset$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\left(\epsilon \cup ba\mathcal{L}\right) = \emptyset$

$b^{-1}R_1 = b^{-1}\left(\epsilon \cup ba\mathcal{L}\right) = a\mathcal{L}$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\,(\epsilon \cup ba\mathcal{L}) = \emptyset$

$b^{-1}R_1 = b^{-1}\,(\epsilon \cup ba\mathcal{L}) = a\mathcal{L}$

$\boxed{R_2 = \emptyset}$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\,(\epsilon \cup ba\mathcal{L}) = \emptyset$

$b^{-1}R_1 = b^{-1}\,(\epsilon \cup ba\mathcal{L}) = a\mathcal{L}$

$\boxed{R_2 = \emptyset}$ $\boxed{R_3 = a\mathcal{L}}$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\,(\epsilon \cup ba\mathcal{L}) = \emptyset$

$b^{-1}R_1 = b^{-1}\,(\epsilon \cup ba\mathcal{L}) = a\mathcal{L}$

$\boxed{R_2 = \emptyset}$ $\boxed{R_3 = a\mathcal{L}}$

$a^{-1}R_2 = b^{-1}R_2 = \emptyset = R_2$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}\,(\epsilon \cup ba\mathcal{L}) = \emptyset$

$b^{-1}R_1 = b^{-1}\,(\epsilon \cup ba\mathcal{L}) = a\mathcal{L}$

$\boxed{R_2 = \emptyset}$ $\boxed{R_3 = a\mathcal{L}}$

$a^{-1}R_2 = b^{-1}R_2 = \emptyset = R_2$

$a^{-1}R_3 = a^{-1}\,(a\mathcal{L}) = \mathcal{L} = R_1$

Automates

Automate associé à un langage régulier : exemple 2

$\mathcal{L} = (ba)^*$

$\mathcal{L} = \epsilon \cup ba\,(ba)^* = \epsilon \cup ba\mathcal{L}$

$\boxed{R_1 = \epsilon^{-1}\mathcal{L} = \mathcal{L}}$

$a^{-1}R_1 = a^{-1}(\epsilon \cup ba\mathcal{L}) = \emptyset$

$b^{-1}R_1 = b^{-1}(\epsilon \cup ba\mathcal{L}) = a\mathcal{L}$

$\boxed{R_2 = \emptyset}$ $\boxed{R_3 = a\mathcal{L}}$

$a^{-1}R_2 = b^{-1}R_2 = \emptyset = R_2$

$a^{-1}R_3 = a^{-1}(a\mathcal{L}) = \mathcal{L} = R_1$

$b^{-1}R_3 = b^{-1}(a\mathcal{L}) = \emptyset$

Automates

Automates

Automate associé à un langage régulier : exemple 2

Automates

Automate associé à un langage régulier : exemple 2

$R_1 = \mathcal{L} = (ba)^*$; $R_2 = \emptyset$; $R_3 = a\,(ba)^*$.

Automate associé à un langage régulier : exemple 2

$R_1 = \mathcal{L} = (ba)^*$; $R_2 = \emptyset$; $R_3 = a\,(ba)^*$.
$a^{-1}R_1 = R_2$; $b^{-1}R_1 = R_3$

Automate associé à un langage régulier : exemple 2

$R_1 = \mathcal{L} = (ba)^*$; $R_2 = \emptyset$; $R_3 = a\,(ba)^*$.
$a^{-1}R_1 = R_2$; $b^{-1}R_1 = R_3$
$a^{-1}R_2 = b^{-1}R_2 = R_2$

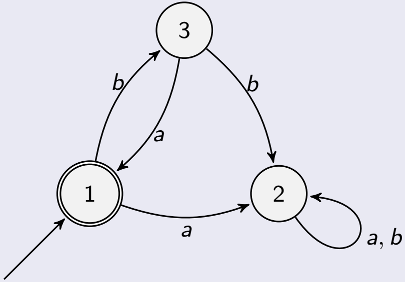
Automates

Automate associé à un langage régulier : exemple 2

$R_1 = \mathcal{L} = (ba)^*$; $R_2 = \emptyset$; $R_3 = a\,(ba)^*$.
$a^{-1}R_1 = R_2$; $b^{-1}R_1 = R_3$
$a^{-1}R_2 = b^{-1}R_2 = R_2$
$a^{-1}R_3 = R_1$; $b^{-1}R_3 = R_2$

Automate associé à un langage régulier : exemple 2

$R_1 = \mathcal{L} = (ba)^*$; $R_2 = \emptyset$; $R_3 = a\,(ba)^*$.
$a^{-1}R_1 = R_2$; $b^{-1}R_1 = R_3$
$a^{-1}R_2 = b^{-1}R_2 = R_2$
$a^{-1}R_3 = R_1$; $b^{-1}R_3 = R_2$



Automates

Automates

Exemple de synthèse

Automates

Exemple de synthèse

On considére $Aut(\mathcal{A}, \mathcal{E}, i_0, \mathcal{F}, \delta)$ avec $\mathcal{A} = \{a, b\}$, $\mathcal{E} = \{1, 2, 3, 4, 5, 6\}$, $i_0 = 1$, $\mathcal{F} = \{4\}$ et la table des transitions :

T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6

Automates

Automates

Exemple de synthèse : représentation graphique

Exemple de synthèse : représentation graphique

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

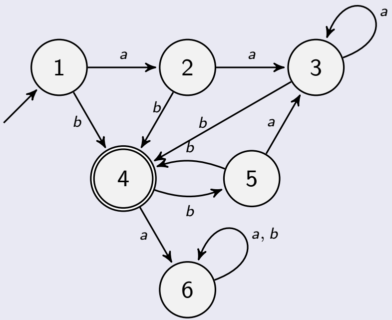
T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6

Automates

Exemple de synthèse : représentation graphique

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6



Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

Exemple de synthèse : langage de l'automate (méthode du départ)

$\boxed{\text{Méthode du départ}}$

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

$\boxed{\text{Méthode du départ}}$

$$\begin{cases} D_1 = aD_2 \cup bD_4 \\ D_2 = aD_3 \cup bD_4 \\ D_3 = aD_3 \cup bD_4 \\ D_4 = \epsilon \cup aD_6 \cup bD_5 \\ D_5 = aD_3 \cup bD_4 \\ D_6 = \{a, b\}D_6 \end{cases}$$

Exemple de synthèse : langage de l'automate (méthode du départ)

Méthode du départ

$$\begin{cases} D_1 = aD_2 \cup bD_4 \\ D_2 = aD_3 \cup bD_4 \\ D_3 = aD_3 \cup bD_4 \\ D_4 = \epsilon \cup aD_6 \cup bD_5 \\ D_5 = aD_3 \cup bD_4 \\ D_6 = \{a, b\}D_6 \end{cases} \qquad \begin{cases} D_6 = \{a, b\}D_6 \cup \emptyset = \{a, b\}^*\emptyset = \emptyset \\ D_3 = D_2 \\ D_5 = D_2 \\ D_4 = a\emptyset \cup bD_5 \cup \epsilon \\ D_2 = aD_2 \cup bD_4 \\ D_1 = aD_2 \cup bD_4 \end{cases}$$

Automates

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

$$\begin{cases} D_4 = bD_2 \cup \epsilon \\ D_2 = aD_2 \cup b\left(bD_2 \cup \epsilon\right) \\ D_1 = D_2 \\ D_3 = D_2 \\ D_5 = D_2 \\ D_6 = \emptyset \end{cases}$$

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

$$\begin{cases} D_4 = bD_2 \cup \epsilon \\ D_2 = aD_2 \cup b\left(bD_2 \cup \epsilon\right) \\ D_1 = D_2 \\ D_3 = D_2 \\ D_5 = D_2 \\ D_6 = \emptyset \end{cases} \qquad \begin{cases} D_2 = \left(a \cup b^2\right) D_2 \cup b \\ D_1 = D_2 \\ D_3 = D_2 \\ D_5 = D_2 \\ D_4 = bD_2 \cup \epsilon \\ D_6 = \emptyset \end{cases}$$

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

$$\begin{cases} D_2 = \left(a \cup b^2\right)^* b \\ D_1 = D_2 \\ D_3 = D_2 \\ D_4 = b\left(a \cup b^2\right)^* b \cup \epsilon \\ D_5 = D_2 \\ D_6 = \emptyset \end{cases}$$

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

$$\begin{cases} D_2 = \left(a \cup b^2\right)^* b \\ D_1 = D_2 \\ D_3 = D_2 \\ D_4 = b\left(a \cup b^2\right)^* b \cup \epsilon \\ D_5 = D_2 \\ D_6 = \emptyset \end{cases} \qquad \begin{cases} D_1 = \left(a \cup b^2\right)^* b \\ D_2 = \left(a \cup b^2\right)^* b \\ D_3 = \left(a \cup b^2\right)^* b \\ D_5 = \left(a \cup b^2\right)^* b \\ D_4 = b\left(a \cup b^2\right)^* b \cup \epsilon \\ D_6 = \emptyset \end{cases}$$

Automates

Exemple de synthèse : langage de l'automate (méthode du départ)

$$\begin{cases} D_2 = \left(a \cup b^2\right)^* b \\ D_1 = D_2 \\ D_3 = D_2 \\ D_4 = b\left(a \cup b^2\right)^* b \cup \epsilon \\ D_5 = D_2 \\ D_6 = \emptyset \end{cases} \qquad \begin{cases} D_1 = \left(a \cup b^2\right)^* b \\ D_2 = \left(a \cup b^2\right)^* b \\ D_3 = \left(a \cup b^2\right)^* b \\ D_5 = \left(a \cup b^2\right)^* b \\ D_4 = b\left(a \cup b^2\right)^* b \cup \epsilon \\ D_6 = \emptyset \end{cases}$$

$$\boxed{\mathcal{L} = D_1 = \left(a \cup b^2\right)^* b}$$

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

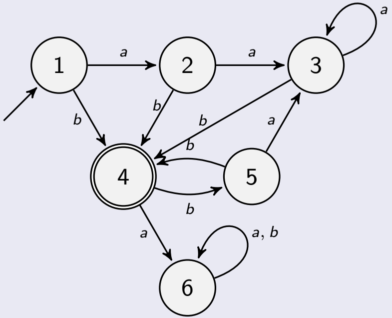
T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6



Automates

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

Méthode de l'arrivée

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\boxed{\text{Méthode de l'arrivée}}$

$$\begin{cases} A_1 &= \epsilon \\ A_2 &= A_1 a \\ A_3 &= A_2 a \cup A_3 a \cup A_5 a \\ A_4 &= A_1 b \cup A_2 b \cup A_3 b \cup A_5 b \\ A_5 &= A_4 b \\ A_6 &= A_4 a \cup A_6 \{a, b\} \end{cases}$$

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\boxed{\text{Méthode de l'arrivée}}$

$$\begin{cases} A_1 = \epsilon \\ A_2 = A_1 a \\ A_3 = A_2 a \cup A_3 a \cup A_5 a \\ A_4 = A_1 b \cup A_2 b \cup A_3 b \cup A_5 b \\ A_5 = A_4 b \\ A_6 = A_4 a \cup A_6\{a, b\} \end{cases} \qquad \begin{cases} A_1 = \epsilon \\ A_2 = a \\ A_3 = a^2 \cup A_3 a \cup A_4 ba \\ A_4 = b \cup ab \cup A_3 b \cup A_4 b^2 \\ A_5 = A_4 b \\ A_6 = A_6\{a, b\} \cup A_4 a \end{cases}$$

Automates

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$\begin{cases} A_3 = A_3 a \cup \left(A_4 ba \cup a^2\right) \\ A_4 = b \cup ab \cup A_3 b \cup A_4 b^2 \\ A_5 = A_4 b \\ A_6 = A_6\{a, b\} \cup A_4 a \\ A_1 = \epsilon \\ A_2 = a \end{cases}$$

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$\begin{cases} A_3 = A_3 a \cup \left(A_4 ba \cup a^2\right) \\ A_4 = b \cup ab \cup A_3 b \cup A_4 b^2 \\ A_5 = A_4 b \\ A_6 = A_6\{a, b\} \cup A_4 a \\ A_1 = \epsilon \\ A_2 = a \end{cases} \qquad \begin{cases} A_3 = \left(A_4 ba \cup a^2\right) a^* \\ A_4 = b \cup ab \cup \left(A_4 ba \cup a^2\right) a^* b \cup A_4 b^2 \\ A_5 = A_4 b \\ A_6 = A_6\{a, b\} \cup A_4 a \\ A_1 = \epsilon \\ A_2 = a \end{cases}$$

Automates

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$\left\{\begin{array}{l} A_4 = A_4\left(b^2 \cup baa^* b\right) \cup \left(b \cup ab \cup a^2 a^* b\right) \\ A_3 = \left(A_4 ba \cup a^2\right) a^* \\ A_5 = A_4 b \\ A_6 = A_6\{a, b\} \cup A_4 a \\ A_1 = \epsilon \\ A_2 = a \end{array}\right.$$

Automates

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$\begin{cases} A_4 = \left(b \cup ab \cup a^2a^*b\right)\left(b^2 \cup baa^*b\right)^* \\ A_3 = \left(A_4ba \cup a^2\right)a^* \\ A_5 = A_4b \\ A_6 = A_6\{a,b\} \cup A_4a \\ A_1 = \epsilon \\ A_2 = a \end{cases}$$

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$\begin{cases} A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^* \\ A_3 = \left(A_4 ba \cup a^2\right) a^* \\ A_5 = A_4 b \\ A_6 = A_6\{a, b\} \cup A_4 a \\ A_1 = \epsilon \\ A_2 = a \end{cases}$$

$$\boxed{\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^*}$$

Automates

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right)\left(b^2 \cup baa^* b\right)^*$$

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^*$

Simplification :

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2a^*b\right)\left(b^2 \cup baa^*b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2a^*b\right) = \left(\epsilon \cup a \cup a^2a^*\right)b$

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \cup a^2 a^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \left(\epsilon \cup aa^*\right)\right) b$

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2a^*b\right)\left(b^2 \cup baa^*b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2a^*b\right) = \left(\epsilon \cup a \cup a^2a^*\right) b$

$\left(b \cup ab \cup a^2a^*b\right) = \left(\epsilon \cup a\left(\epsilon \cup aa^*\right)\right) b$

$\left(b \cup ab \cup a^2a^*b\right) = \left(\epsilon \cup aa^*\right) b$

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2a^*b\right)\left(b^2 \cup baa^*b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2a^*b\right) = \left(\epsilon \cup a \cup a^2a^*\right)b$

$\left(b \cup ab \cup a^2a^*b\right) = \left(\epsilon \cup a\left(\epsilon \cup aa^*\right)\right)b$

$\left(b \cup ab \cup a^2a^*b\right) = \left(\epsilon \cup aa^*\right)b$

$\left(b \cup ab \cup a^2a^*b\right) = a^*b$

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \cup a^2 a^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \left(\epsilon \cup aa^*\right)\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup aa^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = a^* b$

$\left(b^2 \cup baa^* b\right)^* = b \left(\epsilon \cup aa^*\right) b$

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \cup a^2 a^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \left(\epsilon \cup aa^*\right)\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup aa^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = a^* b$

$\left(b^2 \cup baa^* b\right)^* = b \left(\epsilon \cup aa^*\right) b$

$\left(b^2 \cup baa^* b\right)^* = ba^* b$

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \cup a^2 a^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \left(\epsilon \cup aa^*\right)\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup aa^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = a^* b$

$\left(b^2 \cup baa^* b\right)^* = b \left(\epsilon \cup aa^*\right) b$

$\left(b^2 \cup baa^* b\right)^* = ba^* b$

$\mathcal{L} = a^* b \left(ba^* b\right)^*$

Automates

Exemple de synthèse : langage de l'automate (méthode de l'arrivée)

$\mathcal{L} = A_4 = \left(b \cup ab \cup a^2 a^* b\right) \left(b^2 \cup baa^* b\right)^*$

Simplification :

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \cup a^2 a^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup a \left(\epsilon \cup aa^*\right)\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = \left(\epsilon \cup aa^*\right) b$

$\left(b \cup ab \cup a^2 a^* b\right) = a^* b$

$\left(b^2 \cup baa^* b\right)^* = b \left(\epsilon \cup aa^*\right) b$

$\left(b^2 \cup baa^* b\right)^* = ba^* b$

$\mathcal{L} = a^* b \left(ba^* b\right)^*$

Méthode du départ : $\mathcal{L} = \left(a \cup b^2\right)^* b$.

Automates

Exemple de synthèse : langage de l'automate (méthode de Nerode)

Exemple de synthèse : langage de l'automate (méthode de Nerode)

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

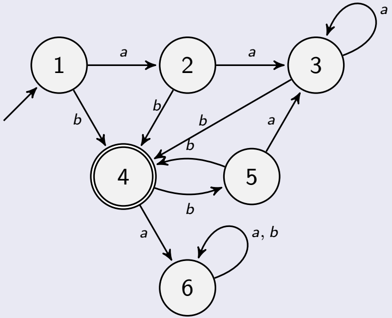
T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6

Automates

Exemple de synthèse : langage de l'automate (méthode de Nerode)

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6



Automates

Exemple de synthèse : automate minimal (méthode de Nerode)

Automates

Exemple de synthèse : automate minimal (méthode de Nerode)

	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
1		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
2		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
3		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
4	ϵ	$\{4\}$		$\{4\}$	bb	$\{4\}$
5		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
6		$\{1,2,3,5,6\}$		$\{6\}$		$\{6\}$

Exemple de synthèse : automate minimal (méthode de Nerode)

	0	$\equiv_0$	1	$\equiv_1$	2	$\equiv_2$
1		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
2		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
3		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
4	ϵ	$\{4\}$		$\{4\}$	bb	$\{4\}$
5		$\{1,2,3,5,6\}$	b	$\{1,2,3,5\}$	ab	$\{1,2,3,5\}$
6		$\{1,2,3,5,6\}$		$\{6\}$		$\{6\}$

Classes : $\{1,2,3,5\}$, $\{4\}$, $\{6\}$

Automates

Exemple de synthèse : langage de l'automate (méthode de Nerode)

Exemple de synthèse : langage de l'automate (méthode de Nerode)

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

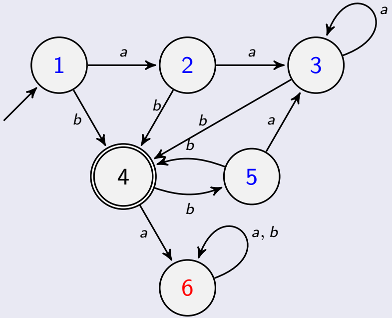
T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6

Automates

Exemple de synthèse : langage de l'automate (méthode de Nerode)

$$i_0 = 1$$
$$\mathcal{F} = \{4\}$$

T	1	2	3	4	5	6
a	2	3	3	6	3	6
b	4	4	4	5	4	6



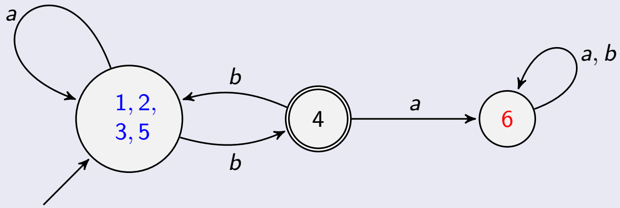
Automates

Automates

Exemple de synthèse : automate minimal

Automates

Exemple de synthèse : automate minimal



Annexes : automates

Annexes : automates

Démonstration du lemme d'Arden

Annexes : automates

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2 X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1}X \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1}X \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On en déduit $\forall n \geq 0$ $\mathcal{L}^n\mathcal{L}' \subset X$ et $\boxed{\mathcal{L}^*\mathcal{L}' \subset X}$.

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2 X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1} X \cup \mathcal{L}^n \mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On en déduit $\forall n \geq 0$ $\mathcal{L}^n \mathcal{L}' \subset X$ et $\boxed{\mathcal{L}^* \mathcal{L}' \subset X}$.
Il existe donc un langage $\mathcal{L}''$ tel que $\boxed{X = \mathcal{L}^* \mathcal{L}' \cup \mathcal{L}''}$.

Annexes : automates

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2 X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1} X \cup \mathcal{L}^n \mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On en déduit $\forall n \geq 0$ $\mathcal{L}^n\mathcal{L}' \subset X$ et $\boxed{\mathcal{L}^*\mathcal{L}' \subset X}$.

Il existe donc un langage $\mathcal{L}''$ tel que $\boxed{X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''}$.
On a alors $X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\left(\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''\right) \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}'' \cup \mathcal{L}'$

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2 X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1}X \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On en déduit $\forall n \geq 0$ $\mathcal{L}^n\mathcal{L}' \subset X$ et $\boxed{\mathcal{L}^*\mathcal{L}' \subset X}$.

Il existe donc un langage $\mathcal{L}''$ tel que $\boxed{X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''}$.
On a alors $X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\,(\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}'' \cup \mathcal{L}'$
puis $\forall n \geq 0$,
$X = \mathcal{L}^{n+1}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^{n+1}\mathcal{L}'' \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.

Annexes : automates

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2 X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1}X \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On en déduit $\forall n \geq 0$ $\mathcal{L}^n\mathcal{L}' \subset X$ et $\boxed{\mathcal{L}^*\mathcal{L}' \subset X}$.

Il existe donc un langage $\mathcal{L}''$ tel que $\boxed{X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''}$.
On a alors $X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\left(\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''\right) \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}'' \cup \mathcal{L}'$
puis $\forall n \geq 0$,
$X = \mathcal{L}^{n+1}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^{n+1}\mathcal{L}'' \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On a $\forall n \geq 0$, $\mathcal{L}^n\mathcal{L}'' \subset X$, donc $\mathcal{L}^*\mathcal{L}'' \subset X$.

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1}X \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On en déduit $\forall n \geq 0$ $\mathcal{L}^n\mathcal{L}' \subset X$ et $\boxed{\mathcal{L}^*\mathcal{L}' \subset X}$.

Il existe donc un langage $\mathcal{L}''$ tel que $\boxed{X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''}$.
On a alors $X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\left(\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''\right) \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}'' \cup \mathcal{L}'$
puis $\forall n \geq 0$,
$X = \mathcal{L}^{n+1}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^{n+1}\mathcal{L}'' \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On a $\forall n \geq 0$, $\mathcal{L}^n\mathcal{L}'' \subset X$, donc $\mathcal{L}^*\mathcal{L}'' \subset X$.
On obtient $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}'' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}''$.

Démonstration du lemme d'Arden

Si $X = \mathcal{L}X \cup \mathcal{L}'$ alors $X = \mathcal{L}(\mathcal{L}X \cup \mathcal{L}') \cup \mathcal{L}' = \mathcal{L}^2 X \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$
et $\forall n \geq 0$, $X = \mathcal{L}^{n+1}X \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On en déduit $\forall n \geq 0$ $\mathcal{L}^n\mathcal{L}' \subset X$ et $\boxed{\mathcal{L}^*\mathcal{L}' \subset X}$.

Il existe donc un langage $\mathcal{L}''$ tel que $\boxed{X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''}$.
On a alors $X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\left(\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}''\right) \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}'' \cup \mathcal{L}'$
puis $\forall n \geq 0$,
$X = \mathcal{L}^{n+1}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^{n+1}\mathcal{L}'' \cup \mathcal{L}^n\mathcal{L}' \cup \mathcal{L}^{n-1}\mathcal{L}' \cup \ldots \cup \mathcal{L}^2\mathcal{L}' \cup \mathcal{L}\mathcal{L}' \cup \mathcal{L}'$.
On a $\forall n \geq 0$, $\mathcal{L}^n\mathcal{L}'' \subset X$, donc $\mathcal{L}^*\mathcal{L}'' \subset X$.
On obtient $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}'' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}''$.

Conclusion : $\boxed{X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')}$.

Annexes : automates

Démonstration du lemme d'Arden

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.

En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,

Annexes : automates

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.

En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,

$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.
En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,
$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.
En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,
$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

1. Si $\epsilon \in \mathcal{L}$ alors $\mathcal{L}\mathcal{L}^* = \mathcal{L}^*$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ avec $\mathcal{L}''$ quelconque.

Annexes : automates

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.
En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,
$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

1. Si $\epsilon \in \mathcal{L}$ alors $\mathcal{L}\mathcal{L}^* = \mathcal{L}^*$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ avec $\mathcal{L}''$ quelconque.
2. Sinon, on suppose $\mathcal{L}^*\mathcal{L}'' \not\subset \mathcal{L}^*\mathcal{L}'$.

Annexes : automates

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.
En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,
$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

1. Si $\epsilon \in \mathcal{L}$ alors $\mathcal{L}\mathcal{L}^* = \mathcal{L}^*$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ avec $\mathcal{L}''$ quelconque.
2. Sinon, on suppose $\mathcal{L}^*\mathcal{L}'' \not\subset \mathcal{L}^*\mathcal{L}'$.
   Il existe alors un mot $m \in \mathcal{L}^*\mathcal{L}''$, $m \notin \mathcal{L}^*\mathcal{L}'$ avec $l(m)$ la plus petite possible.

Annexes : automates

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.
En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,
$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

1. Si $\epsilon \in \mathcal{L}$ alors $\mathcal{L}\mathcal{L}^* = \mathcal{L}^*$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ avec $\mathcal{L}''$ quelconque.
2. Sinon, on suppose $\mathcal{L}^*\mathcal{L}'' \not\subset \mathcal{L}^*\mathcal{L}'$.
   Il existe alors un mot $m \in \mathcal{L}^*\mathcal{L}''$, $m \notin \mathcal{L}^*\mathcal{L}'$ avec $l(m)$ la plus petite possible.
   Comme $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$, $m \in \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

Annexes : automates

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.
En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,
$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

1. Si $\epsilon \in \mathcal{L}$ alors $\mathcal{L}\mathcal{L}^* = \mathcal{L}^*$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ avec $\mathcal{L}''$ quelconque.
2. Sinon, on suppose $\mathcal{L}^*\mathcal{L}'' \not\subset \mathcal{L}^*\mathcal{L}'$.
   Il existe alors un mot $m \in \mathcal{L}^*\mathcal{L}''$, $m \notin \mathcal{L}^*\mathcal{L}'$ avec $l(m)$ la plus petite possible.
   Comme $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$, $m \in \mathcal{L}\mathcal{L}^*\mathcal{L}''$.
   Donc $m = jk$ avec $j \in \mathcal{L}$, $j \neq \epsilon$ et $k \in \mathcal{L}^*\mathcal{L}''$.

Annexes : automates

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.

En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,

$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$

$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

1. Si $\epsilon \in \mathcal{L}$ alors $\mathcal{L}\mathcal{L}^* = \mathcal{L}^*$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ avec $\mathcal{L}''$ quelconque.
2. Sinon, on suppose $\mathcal{L}^*\mathcal{L}'' \not\subset \mathcal{L}^*\mathcal{L}'$.
   Il existe alors un mot $m \in \mathcal{L}^*\mathcal{L}''$, $m \notin \mathcal{L}^*\mathcal{L}'$ avec $l(m)$ la plus petite possible.
   Comme $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$, $m \in \mathcal{L}\mathcal{L}^*\mathcal{L}''$.
   Donc $m = jk$ avec $j \in \mathcal{L}$, $j \neq \epsilon$ et $k \in \mathcal{L}^*\mathcal{L}''$.
   On a donc $k \in \mathcal{L}^*\mathcal{L}''$ avec $l(k) < l(m)$.

Annexes : automates

Démonstration du lemme d'Arden

Précisons $\mathcal{L}''$.
En reprenant $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$,
$X = \mathcal{L}X \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\,\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$.

1. Si $\epsilon \in \mathcal{L}$ alors $\mathcal{L}\mathcal{L}^* = \mathcal{L}^*$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$ avec $\mathcal{L}''$ quelconque.
2. Sinon, on suppose $\mathcal{L}^*\mathcal{L}'' \not\subset \mathcal{L}^*\mathcal{L}'$.
   Il existe alors un mot $m \in \mathcal{L}^*\mathcal{L}''$, $m \notin \mathcal{L}^*\mathcal{L}'$ avec $l(m)$ la plus petite possible.
   Comme $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}\mathcal{L}^*\mathcal{L}''$, $m \in \mathcal{L}\mathcal{L}^*\mathcal{L}''$.
   Donc $m = jk$ avec $j \in \mathcal{L}$, $j \neq \epsilon$ et $k \in \mathcal{L}^*\mathcal{L}''$.
   On a donc $k \in \mathcal{L}^*\mathcal{L}''$ avec $l(k) < l(m)$.
   On en déduit $\mathcal{L}^*\mathcal{L}'' \subset \mathcal{L}^*\mathcal{L}'$ et $X = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}'$.

Annexes : automates

Annexes : automates

Démonstration du lemme d'Arden

Annexes : automates

Démonstration du lemme d'Arden

Réciproquement

Annexes : automates

Démonstration du lemme d'Arden

Réciproquement

1. Si $\epsilon \in \mathcal{L}$ alors
$\mathcal{L}\mathcal{L}^*\left(\mathcal{L}' \cup \mathcal{L}''\right) \cup \mathcal{L}' = \mathcal{L}^*\left(\mathcal{L}' \cup \mathcal{L}''\right) \cup \mathcal{L}' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$

Démonstration du lemme d'Arden

Réciproquement

1. Si $\epsilon \in \mathcal{L}$ alors
$\mathcal{L}\mathcal{L}^* (\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}^* (\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = (\mathcal{L}^* \cup \epsilon) \mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}''$

Annexes : automates

Démonstration du lemme d'Arden

Réciproquement

1. Si $\epsilon \in \mathcal{L}$ alors
$\mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = (\mathcal{L}^* \cup \epsilon)\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}''$
$= \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$.

Démonstration du lemme d'Arden

Réciproquement

1. Si $\epsilon \in \mathcal{L}$ alors
$\mathcal{L}\mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'') \cup \mathcal{L}' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' \cup \mathcal{L}'$
$= \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = (\mathcal{L}^* \cup \epsilon)\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}'' = \mathcal{L}^*\mathcal{L}' \cup \mathcal{L}^*\mathcal{L}''$
$= \mathcal{L}^*(\mathcal{L}' \cup \mathcal{L}'')$.
2. Sinon, $\mathcal{L}\mathcal{L}^*\mathcal{L}' \cup \mathcal{L}' = (\mathcal{L}\mathcal{L}^* \cup \epsilon)\mathcal{L}' = \mathcal{L}^*\mathcal{L}'$

Annexes : historique

Graphes

Annexes : historique

Graphes

La théorie des graphes est apparue avec le problème des ponts de Königsberg au 18ème siècle.
Les développements ont débuté dans les années 1960 avec les travaux du mathématicien français Claude Berge (1926-2002).

Annexes : historique

M2201
Graphes et langages

Département Informatique
IUT de Saint-Dié

Graphes
Généralités
Arbre
Cycle eulérien, hamiltonien
Coloration
Le problème du plus court chemin
Arbre recouvrant de poids minimal

Langages et automates
Langages
Automates

Annexes

Automates

Annexes : historique

Automates

1. Claude Shannon (1949) : Théorie de l'information et de la communication

Annexes : historique

Automates

1. Claude Shannon (1949) : Théorie de l'information et de la communication
2. Stephen Kleene (1956) : Automates finis, expressions rationnelles

Annexes : historique

Automates

1. Claude Shannon (1949) : Théorie de l'information et de la communication
2. Stephen Kleene (1956) : Automates finis, expressions rationnelles
3. Noam Chomsky (1956) : Grammaires formelles

Annexes

Bibliographie et webographie

Annexes

Bibliographie et webographie

1. Théorie des graphes de Olivier Cogis et Claudine Robert
   Vuibert
2. Les graphes de F. Droesbeke, M. Hallin et C. Lefevre
   Ellipses
3. Graphes et algorithmes de Michel Gondran et Michel Minoux
   Eyrolles
4. Méthodes mathématiques pour l'informatique de Jacques Vélu
   Dunod

Annexes

Bibliographie et webographie

Annexes

Bibliographie et webographie

1. La recherche numéro 441 (mai 2010)

Annexes

Bibliographie et webographie

1. La recherche numéro 441 (mai 2010)
2. Les maths cent théorèmes de Roger Beslon et Daniel Lignon
   Le Polygraphe, éditeur
3. Introduction à la théorie des graphes de Jean-Manuel Mny
   CRDP Lyon