

# Cheminements, graphes eulériens et hamiltoniens



© N. Brauner, 2019, M. Stehlik 2020

# Plan

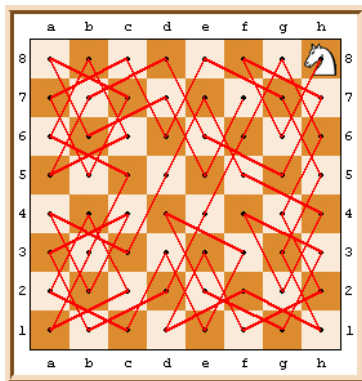
- 1 Chaînes
- 2 Connexité
- 3 Accessibilité
- 4 Parcours
- 5 Cycles
- 6 Graphes eulériens
- 7 Graphes hamiltoniens

# Plan

- 1 Chaînes
- 2 Connexité
- 3 Accessibilité
- 4 Parcours
- 5 Cycles
- 6 Graphes eulériens
- 7 Graphes hamiltoniens

# Chaînes

- se promener / parcourir / traverser un graphe
- pour savoir où on va / aller vite / aller loin / aller partout



Camion poubelle, voiture google map, plus court chemin

# Chaînes

## Définition

Une chaîne d'un graphe  $G$  est une séquence alternée de sommets et d'arêtes de la forme  $(x_0, e_1, x_1, \dots, e_k, x_k)$  où

- $x_i \in V(G)$
  - $e_i \in E(G)$
  - $e_i = x_{i-1}x_i$  pour  $i = 1, \dots, k$ .
- 
- $x_0x_k$ -chaîne (chaîne de  $x_0$  à  $x_k$ )
  - $x_0, x_k$  sont les extrémités de la chaîne
  - si pas d'ambiguïté, on note :  $(x_0, x_1, \dots, x_{k-1}, x_k)$
  - les  $x_i$  ne sont pas nécessairement distincts.

⇒ nombre infini de chaînes

# Chaînes

## Définitions

Une chaîne est dite simple si ses arêtes sont deux à deux distinctes (tous les  $e_i$  distincts).

Une chaîne est dite élémentaire si ses sommets sont deux à deux distincts (tous les  $x_i$  distincts, sauf éventuellement  $x_0 = x_k$ ).

- élémentaire  $\Rightarrow$  simple
- nombre fini de chaînes simples

L'entier  $k$  est la longueur de la chaîne (= nb d'arêtes = nb de sommets - 1).

# Chaînes

$\exists$  chaîne de  $x$  à  $y \Rightarrow \exists$  chaîne élémentaire de  $x$  à  $y$

# Chaînes

$\exists$  chaîne de  $x$  à  $y \Rightarrow \exists$  chaîne élémentaire de  $x$  à  $y$

**Données :**  $C$  une chaîne de  $x$  à  $y$

**Résultat :** Une chaîne élémentaire de  $x$  à  $y$

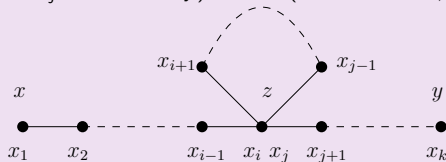
**tant que**  $C$  n'est pas élémentaire **faire**

$\exists$  un sommet  $z$  qui apparaît deux fois dans  $C$  (dont au moins une fois pas aux extrémités).

la chaîne entre deux occurrences de  $z$  devient  $z$

**retourner**  $C$

$(x_1 = x \dots x_i = z \dots x_j = z \dots x_k = y)$  devient  $(x_1 = x \dots x_{i-1}, z, x_{j+1} \dots x_k = y)$





# Chaînes

**Données :**  $C$  une chaîne de  $x$  à  $y$

**Résultat :** Une chaîne élémentaire de  $x$  à  $y$

**tant que  $C$  n'est pas élémentaire faire**

$\exists$  un sommet  $z$  qui apparaît deux fois dans  $C$  (dont au moins une fois pas aux extrémités).

        on remplace la chaîne entre deux occurrences de  $z$  par  $z$

**retourner  $C$**

❶ *l'algorithme s'exécute correctement*

# Chaînes

**Données :**  $C$  une chaîne de  $x$  à  $y$

**Résultat :** Une chaîne élémentaire de  $x$  à  $y$

**tant que  $C$  n'est pas élémentaire faire**

$\exists$  un sommet  $z$  qui apparaît deux fois dans  $C$  (dont au moins une fois pas aux extrémités).

        on remplace la chaîne entre deux occurrences de  $z$  par  $z$

**retourner  $C$**

① *l'algorithme s'exécute correctement*

② *en un nombre fini d'étapes*

# Chaînes

**Données :**  $C$  une chaîne de  $x$  à  $y$

**Résultat :** Une chaîne élémentaire de  $x$  à  $y$

**tant que  $C$  n'est pas élémentaire faire**

$\exists$  un sommet  $z$  qui apparaît deux fois dans  $C$  (dont au moins une fois pas aux extrémités).

        on remplace la chaîne entre deux occurrences de  $z$  par  $z$

**retourner  $C$**

- ① *l'algorithme s'exécute correctement*
- ② *en un nombre fini d'étapes*
  - la longueur de  $C$  décroît strictement et est positive
- ③ *en cas d'arrêt, on obtient l'objet souhaité :*

# Chaînes

**Données :**  $C$  une chaîne de  $x$  à  $y$

**Résultat :** Une chaîne élémentaire de  $x$  à  $y$

**tant que**  $C$  n'est pas élémentaire **faire**

$\exists$  un sommet  $z$  qui apparaît deux fois dans  $C$  (dont au moins une fois pas aux extrémités).

        on remplace la chaîne entre deux occurrences de  $z$  par  $z$

**retourner**  $C$

- ① *l'algorithme s'exécute correctement*
- ② *en un nombre fini d'étapes*
  - la longueur de  $C$  décroît strictement et est positive
- ③ *en cas d'arrêt, on obtient l'objet souhaité :*
  - à chaque étape,  $C$  est une chaîne de  $x$  à  $y$
  - sortie du **tant que** avec une chaîne élémentaire de  $x$  à  $y$

# Chaînes

$\exists$  chaîne de  $x$  à  $y \Rightarrow \exists$  chaîne élémentaire de  $x$  à  $y$

## Preuve alternative

Parmi toutes les chaînes reliant  $x$  à  $y$ , choisissons une chaîne  $C = (x_1 = x, x_2 \dots x_k = y)$  comportant le moins d'arêtes.

Supposons par l'absurde que  $C$  n'est pas élémentaire. Il existe alors un sommet  $z$  apparaissant au moins 2 fois dans  $C$  (dont au moins une fois pas aux extrémités). Soient  $i$  et  $j$  les 2 premiers indices tels que  $x_i = z$  et  $x_j = z$ . On "supprime" le cycle entre  $x_i = z$  et  $x_j = z$ . Alors  $C' = (x_1 = x \dots x_{i-1}, z, x_{j+1} \dots x_k = y)$  est une chaîne, reliant  $x$  à  $y$ . Sa longueur est strictement inférieure à celle de  $C$ , ce qui contredit le choix de  $C$  comme étant une plus courte chaîne.

# Chaînes

$\exists$  chaîne de  $x$  à  $y \Rightarrow \exists$  chaîne élémentaire de  $x$  à  $y$

## Preuve alternative, mais qui revient au même en fait

Parmi toutes les chaînes reliant  $x$  à  $y$ , choisissons une chaîne  $C = (x_1 = x, x_2 \dots x_k = y)$  comportant le moins d'arêtes.

Supposons par l'absurde que  $C$  n'est pas élémentaire. Il existe alors un sommet  $z$  apparaissant au moins 2 fois dans  $C$  (dont au moins une fois pas aux extrémités). Soient  $i$  et  $j$  les 2 premiers indices tels que  $x_i = z$  et  $x_j = z$ . On "supprime" le cycle entre  $x_i = z$  et  $x_j = z$ . Alors  $C' = (x_1 = x \dots x_{i-1}, z, x_{j+1} \dots x_k = y)$  est une chaîne, reliant  $x$  à  $y$ . Sa longueur est strictement inférieure à celle de  $C$ , ce qui contredit le choix de  $C$  comme étant une plus courte chaîne.

# Plan

- 1 Chaînes
- 2 Connexité**
- 3 Accessibilité
- 4 Parcours
- 5 Cycles
- 6 Graphes eulériens
- 7 Graphes hamiltoniens

# Connexité

## Définitions

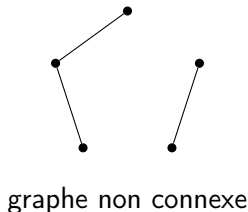
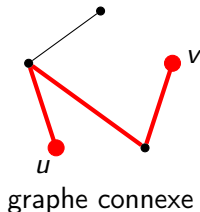
- Un graphe  $G$  est connexe si pour tout  $x, y \in V(G)$ , il existe dans  $G$  une chaîne de  $x$  à  $y$ .
- Une composante connexe d'un graphe  $G$  est un sous-graphe connexe maximal (au sens de l'inclusion).



# Connexité

## Définitions

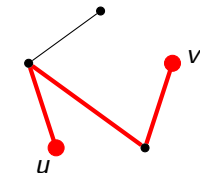
- Un graphe  $G$  est connexe si pour tout  $x, y \in V(G)$ , il existe dans  $G$  une chaîne de  $x$  à  $y$ .
- Une composante connexe d'un graphe  $G$  est un sous-graphe connexe maximal (au sens de l'inclusion).



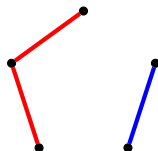
# Connexité

## Définitions

- Un graphe  $G$  est connexe si pour tout  $x, y \in V(G)$ , il existe dans  $G$  une chaîne de  $x$  à  $y$ .
- Une composante connexe d'un graphe  $G$  est un sous-graphe connexe maximal (au sens de l'inclusion).



graphe connexe



graphe non connexe

# Connexité

On peut aussi définir les composantes connexes en utilisant les relations d'équivalence. On note  $x \sim y$  s'il existe une chaîne de  $x$  à  $y$ .

- $\sim$  est une **relation d'équivalence** (graphe non orienté).  
(réflexive, symétrique et transitive).
- Les classes d'équivalence de  $\sim$  induisent les composantes connexes de  $G$ .

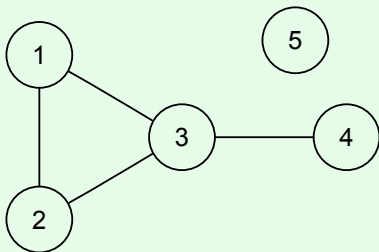
# Connexité

On peut aussi définir les composantes connexes en utilisant les relations d'équivalence. On note  $x \sim y$  s'il existe une chaîne de  $x$  à  $y$ .

- $\sim$  est une **relation d'équivalence** (graphe non orienté).  
(réflexive, symétrique et transitive).
- Les classes d'équivalence de  $\sim$  induisent les composantes connexes de  $G$ .

$G$  est connexe  $\Leftrightarrow G$  a une seule composante connexe

# Connexité



- Combien le graphe a-t-il de composantes connexes ?
- Est-il connexe ?

# Connexité

## Lien entre nombre d'arêtes et connexité

### Propriété

Tout graphe  $G$  d'ordre  $n$  connexe comporte au moins  $n - 1$  arêtes.

# Connexité

## Lien entre nombre d'arêtes et connexité

### Propriété

Tout graphe  $G$  d'ordre  $n$  connexe comporte au moins  $n - 1$  arêtes.

### Démonstration : récurrence sur l'ordre du graphe.

Soit  $P(n)$  la propriété ci-dessus.

**Init.**  $n = 1$  La propriété est clairement vraie.  $P(1)$  est vraie.

Supposons que pour  $n \geq 1$  quelconque,  $P(n)$  est vraie.

Considérons un graphe  $G = (V, E)$  connexe d'ordre  $n + 1$ .



# Connexité

## Démonstration (suite).

**Cas 1 :** Il existe dans  $G$  un sommet  $x$  de degré 1.

Une seule arête, appelons-la  $e = (x, y)$ , est alors incidente à  $x$ .

Considérons le graphe  $G' = (V \setminus \{x\}, E \setminus \{e\})$ .  $G'$  est d'ordre  $n$ .

**Le graphe  $G'$  reste connexe :** si l'on considère 2 sommets  $u$  et  $v$  de  $V(G')$  (donc  $u \neq x$  et  $v \neq x$ ), il existe dans  $G$  une chaîne  $C$ , que nous pouvons choisir élémentaire (Lemme précédent), reliant ces 2 sommets. Si une chaîne passe par  $x$ , alors elle doit passer deux fois par  $y$  ( $x$  n'est pas une extrémité). Or  $C$  est élémentaire. Donc, elle ne passe pas par  $x$ . Donc,  $C$  est également une chaîne de  $G'$  reliant  $u$  et  $v$ .

L'hypothèse de récurrence impose que  $G'$  comporte au moins  $n - 1$  arêtes. Cependant  $G$  possède exactement un sommet et une arête de plus que  $G'$ . Donc  $G$  possède au moins  $n$  arêtes :  $P(n + 1)$  est vraie. □



# Connexité

## Démonstration (fin).

**Cas 2 :** Il n'existe pas dans  $G$  de sommet de degré 1.

Puisque  $G$  est connexe d'ordre au moins 2, il ne peut exister de sommet isolé. Tout sommet de  $G$  est donc de degré au moins 2. On a alors :

$$\sum_{v \in V} d(v) \geq 2|V| = 2(n+1)$$

et donc

$$2|E| = \sum_{v \in V} d(v) \geq 2(n+1)$$

ce qui permet de conclure que  $G$  a au moins  $n$  arêtes.

Donc  $P(n)$  est vraie pour tout  $n \geq 1$ .



# Plan

- 1 Chaînes
- 2 Connexité
- 3 Accessibilité**
- 4 Parcours
- 5 Cycles
- 6 Graphes eulériens
- 7 Graphes hamiltoniens

# Accessibilité

Soit  $S \subsetneq V$  et  $S \neq \emptyset$

## Définition

On appelle co-cycle de  $S$  l'ensemble des arêtes de  $E(G)$  ayant exactement une extrémité dans  $S$

$$\{ij \in E \mid i \in S \text{ et } j \notin S\}$$

## Question

Quel est le lien entre co-cycle de  $S$  et co-cycle de  $V \setminus S$ ?

# Accessibilité

## Définition

$t$  est **accessible** depuis  $s$  s'il existe une chaîne de  $s$  à  $t$

*Question* oui/non avec certificat :

Est-ce que  $t$  est **accessible** depuis  $s$  ?

# Accessibilité

## Définition

$t$  est **accessible** depuis  $s$  s'il existe une chaîne de  $s$  à  $t$

*Question* oui/non avec certificat :

Est-ce que  $t$  est **accessible** depuis  $s$  ?

*oui* :  $st$ -chaîne

# Accessibilité

## Définition

$t$  est **accessible** depuis  $s$  s'il existe une chaîne de  $s$  à  $t$

*Question* oui/non avec certificat :

Est-ce que  $t$  est **accessible** depuis  $s$  ?

*oui* :  $st$ -chaîne

*non* :  $S \subset V$  avec  $s \in S$  et  $t \notin S$  et  $\text{co-cycle}(S) = \emptyset$

# Accessibilité

## Théorème

*Si pour  $S \subset V$  on a  $s \in S$  et  $t \notin S$  et  $\text{co-cycle}(S) = \emptyset$  alors, il n'y a pas de chaîne de  $s$  à  $t$ .*

## Démonstration : par l'absurde.

Supposons qu'il existe une chaîne  $C = x_0, x_1 \dots x_k$  de  $s = x_0$  à  $t = x_k$ .

Soit  $i$  le plus grand indice d'un sommet de  $C$  qui appartient à  $S$ .

Comme  $s \in S$ , l'indice  $i$  existe et comme  $t \notin S$ , on a  $i < k$ .

Par définition de  $i$ , l'arête  $x_i x_{i+1}$  a une extrémité dans  $S$  et l'autre hors de  $S$  ce qui contredit  $\text{co-cycle}(S) = \emptyset$ . □

Un tel  $S$  est alors bien un certificat du non !

# Accessibilité

## Accessibles

- algorithme (méthode) de graphe
- prend un sommet  $s$  en paramètre
- retourne la liste de tous les sommets accessibles depuis  $s$



# Accessibilité

## Accessibles

- algorithme (méthode) de graphe
- prend un sommet  $s$  en paramètre
- retourne la liste de tous les sommets accessibles depuis  $s$

---

### Accessibles( $s$ )

---

**Données** : un sommet  $s$  (et un graphe  $G$ )

**Résultat** : l'ensemble des sommets accessibles depuis  $s$  (dans  $G$ )

$S \leftarrow \{s\}$

**tant que** *il existe*  $ij \in E$  avec  $i \in S$  et  $j \notin S$  **faire**

$S \leftarrow S \cup \{j\}$

**retourner**  $S$

---

# Accessibilité

## Accessibilité

- algorithme (méthode) de graphe
- prend deux sommets  $s$  et  $t$  en paramètre
- retourne vrai ssi il existe une chaîne d'extrémités  $s$  et  $t$

# Accessibilité

## Accessibilité

- algorithme (méthode) de graphe
- prend deux sommets  $s$  et  $t$  en paramètre
- retourne vrai ssi il existe une chaîne d'extrémités  $s$  et  $t$

---

### Accessibilité( $s, t$ )

---

**Données** : deux sommets  $s$  et  $t$

**Résultat** : Vrai ssi il existe une  $st$ -chaîne

**retourner**  $t \in Accessibles(s)$

---

# Accessibilité

## Vérification de Accessibles( $s$ )

- 1 *tout  $s$ 'exécute correctement*

# Accessibilité

## Vérification de Accessibles( $s$ )

- 1 *tout  $s$  s'exécute correctement*
- 2 *en un nombre fini d'étapes*

# Accessibilité

## Vérification de Accessibles( $s$ )

- ① *tout  $s$  s'exécute correctement*
- ② *en un nombre fini d'étapes*
  - $S \subset V$  et à chaque étape  $|S|$  augmente de 1
- ③ *en cas d'arrêt, on obtient l'objet souhaité*

# Accessibilité

## Vérification de Accessibles( $s$ )

- ① *tout  $s$  s'exécute correctement*
- ② *en un nombre fini d'étapes*
  - $S \subset V$  et à chaque étape  $|S|$  augmente de 1
- ③ *en cas d'arrêt, on obtient l'objet souhaité*
  - tous les sommets accessibles depuis  $s$  sont dans  $S$   
(preuve juste après)
  - tous les sommets de  $S$  sont accessibles :  
on va écrire `Reconstruire_chaine( $s, t$ )` qui pour tout  
sommets  $t \in S$  renvoie une  $st$ -chaîne

Donc **Accessible( $s$ )** retourne l'ensemble des sommets de la  
composante connexe de  $s$

# Accessibilité

## Certificats si non

Tous les sommets accessibles depuis  $s$  sont dans  $S$

À la fin de l'exécution de l'algorithme,  $S$  viole la condition d'entrée dans le **tant que**. Donc, une arête  $a$  soit ses deux extrémités dans  $S$ , soit aucune de ses extrémités dans  $S$  (donc  $\text{co-cycle}(S) = \emptyset$ ). Donc d'après le théorème, si  $t \notin S$ , alors il n'y a pas de chaîne de  $s$  à  $t$ .

Si **Accessibilité**( $s, t$ ) retourne non alors **Accessible**( $s$ ) retourne  $S$  avec  $s \in S$  et  $t \notin S$  et  $\text{co-cycle}(S) = \emptyset$

On a démontré que si  $t \notin S$  alors il n'existe pas de  $st$ -chaîne. Donc  $S$  est un certificat du non pour l'existence d'une chaîne de  $s$  à  $t$ .



# Cheminement

Adaptez **Accessibles( $s$ )** pour conserver la mémoire de la chaîne de  $s$  aux sommets de  $S$

# Cheminement

Adaptez **Accessibles( $s$ )** pour conserver la mémoire de la chaîne de  $s$  aux sommets de  $S$

---

## Accessibles( $s$ )

---

**Données** : un sommet  $s$

**Résultat** : l'ensemble des sommets accessibles depuis  $s$  et la mémoire de la chaîne de  $s$  aux sommets de  $S$

$S \leftarrow \{s\}$

$from(s) \leftarrow s$

**tant que** *il existe*  $ij \in E$  avec  $i \in S$  et  $j \notin S$  **faire**

$S \leftarrow S \cup \{j\}$   
     $from(j) \leftarrow i$

**retourner**  $S$  et  $from$

---

$from(j)$  a reçu une valeur pour tous les éléments de  $S$

# Accessibilité

## Certificats si oui

En TP, il vous faudra écrire un algorithme

**Reconstruire\_Chaine**( $s, t, from$ ) qui utilise *from* pour construire une *st*-chaîne. Cet algorithme sera appelé seulement si  $t \in Accessibles(s)$ .

Cela vous permettra d'obtenir un certificat du oui.

# Accessibilité

$t$  est **accessible** depuis  $s$  s'il existe une chaîne de  $s$  à  $t$

*Question* oui/non avec certificat :

Est-ce que  $t$  est **accessible** depuis  $s$  ?

*oui* :  $st$ -chaîne

*non* :  $S \subset V$  avec  $s \in S$  et  $t \notin S$  et  $\text{co-cycle}(S) = \emptyset$

# Plan

- 1 Chaînes
- 2 Connexité
- 3 Accessibilité
- 4 Parcours**
- 5 Cycles
- 6 Graphes eulériens
- 7 Graphes hamiltoniens

# Parcours dans les graphes

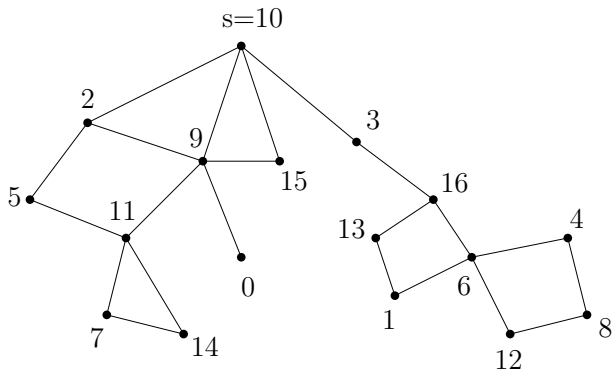
**Accessibles** est un parcours de graphe sans ordre particulier et pas très efficace.

Plus efficace et ordre maîtrisé : **parcours en largeur** et **en profondeur** comme sur les arbres.

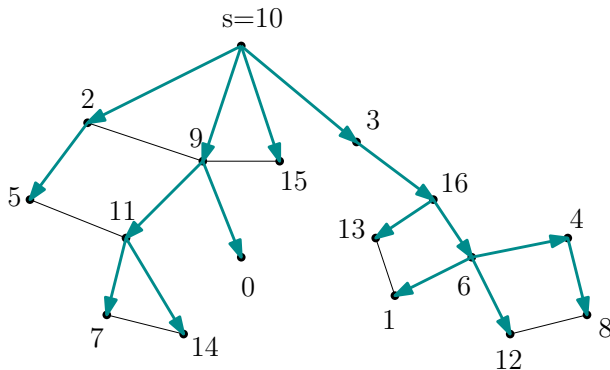
Dans le cadre des graphes, il faut être vigilant car il peut y avoir des cycles. On risque alors de boucler à l'infini : soit parce qu'on ajoute dans la file/pile un sommet déjà visité, soit parce qu'on fait un appel récursif sur un sommet déjà visité.

Dans les deux cas, on règle le problème avec par exemple un **tableau de booléens qui dit quels sommets ont déjà été visités**. Dans le cas d'une méthode récursive, c'est en fait une méthode auxiliaire qui sera récursive, et qui prendra en argument supplémentaire la liste des sommets déjà visités.

# Parcours dans les graphes



# Parcours dans les graphes

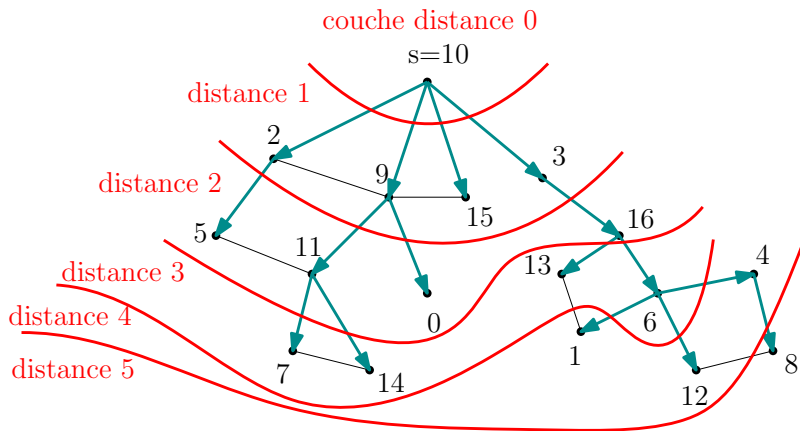


Parcours en largeur au départ de  $s = 10$ :

10 2 3 9 15 5 16 0 11 6 13 7 14 1 4 12 8



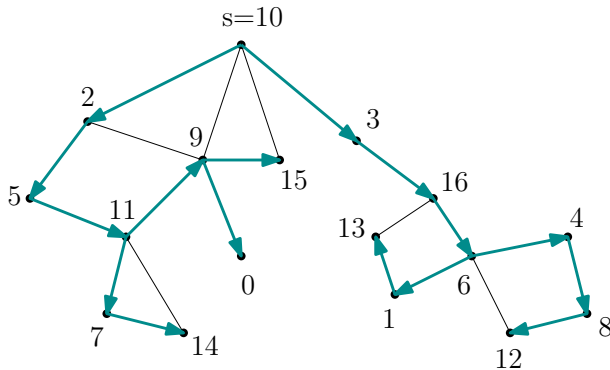
# Parcours dans les graphes



Parcours en largeur au départ de  $s = 10$ :

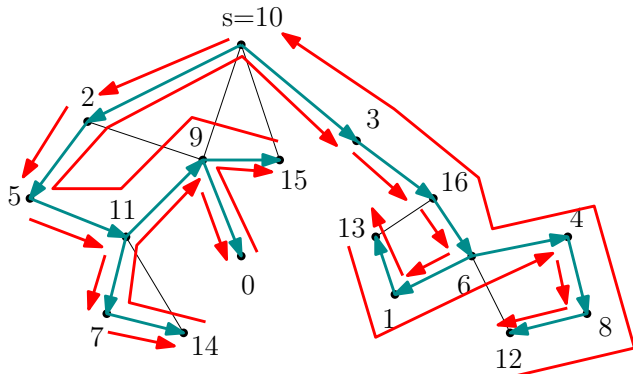
10 2 3 9 15 5 16 0 11 6 13 7 14 1 4 12 8

# Parcours dans les graphes



Parcours en profondeur au départ de  $s = 10$ :  
**10 2 5 11 7 14 9 0 15 3 16 6 1 13 4 8 12**

# Parcours dans les graphes



Parcours en profondeur au départ de  $s = 10$ :  
 10 2 5 11 7 14 9 0 15 3 16 6 1 13 4 8 12

# Parcours en Largeur dans un graphe

---

## ParcoursLargeur( $G, s$ )

---

**Données** : un graphe  $G$  à  $n$  sommets, un sommet  $s$

**Résultat** : affiche les sommets de  $G$  dans l'ordre d'un parcours en largeur depuis  $s$

$F \leftarrow \text{File}()$

$\text{bool } \text{dejaVu}[n] \leftarrow \{\text{Faux}, \dots, \text{Faux}\}$

$F.\text{enfile}(s)$

$\text{dejaVu}[s] \leftarrow \text{Vrai}$

**tant que**  $F$  n'est pas vide **faire**

$v \leftarrow F.\text{pop}()$

    Afficher  $v$

**pour** tout voisin  $u$  de  $v$  **faire**

**si** non  $\text{dejaVu}[u]$  **alors**

$F.\text{enfile}(u)$

$\text{dejaVu}[u] \leftarrow \text{Vrai}$

# Parcours en profondeur récursif dans un graphe

---

**ParcoursProfondeur( $G, s$ )**

---

**Données :** un graphe  $G$  à  $n$  sommets, un sommet  $s$

**Résultat :** affiche les sommets de  $G$  dans l'ordre d'un parcours en profondeur depuis  $s$

bool  $dejaVu[n] \leftarrow \{Faux, \dots, Faux\}$

ParcoursProfondeurAux( $G, s, dejaVu$ )

---



---

**ParcoursProfondeurAux( $G, v, dejaVu$ )**

---

Afficher  $v$

$dejaVu[v] \leftarrow Vrai$

**pour** tout voisin  $u$  de  $v$  faire

|  |   |
|--|---|
| si non $dejaVu[u]$ alors <table> <tr> <td style="border-left: 1px solid black; padding-left: 10px;">                     ParcoursProfondeurAux(<math>G, u, dejaVu</math>)                 </td> </tr> </table> | ParcoursProfondeurAux( $G, u, dejaVu$ ) |
| ParcoursProfondeurAux( $G, u, dejaVu$ )  |   |

---

# Plan

- 1 Chaînes
- 2 Connexité
- 3 Accessibilité
- 4 Parcours
- 5 Cycles**
- 6 Graphes eulériens
- 7 Graphes hamiltoniens

# Cycles

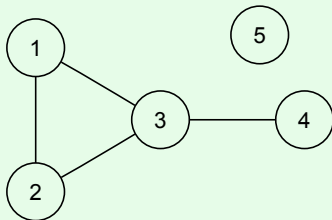
## Définitions

Un cycle est une chaîne simple dont les deux extrémités sont le même sommet ( $x_0 = x_k$ ) (un cycle ne peut être réduit à un seul sommet).

Dans un cycle élémentaire, chaque sommet apparaît une seule fois (sauf les extrémités).

Quelle est la longueur minimale d'un cycle ?

Y-a-t-il dans le graphe suivant un cycle élémentaire ? Et un cycle non élémentaire ?



# Cycles

## Existence d'un cycle

Si dans un graphe  $G$  tout sommet est de degré supérieur ou égal à 2, alors  $G$  possède au moins un cycle.



# Cycles

## Existence d'un cycle

Si dans un graphe  $G$  tout sommet est de degré supérieur ou égal à 2, alors  $G$  possède au moins un cycle.

Adaptez l'algorithme **Accessibilité(s)** pour montrer la propriété précédente.

# Cycles

## Cycle( $x_0$ )

**Données** : un sommet  $x_0$

**Résultat** : Un cycle (élémentaire)

$k \leftarrow 0$     $i \leftarrow x_0$     $C \leftarrow (x_0)$

**tant que** *il existe  $j \notin C$  tel que  $ij \in E$*  **faire**

$k \leftarrow k + 1$

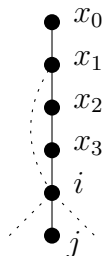
$x_k \leftarrow j$

    ajouter  $x_k$  en queue de  $C$

$i \leftarrow j$

Soit  $x_h$  avec  $h < k - 1$  un voisin de  $x_k$  dans  $C$

**retourner**  $(x_k, x_h, x_{h+1}, \dots, x_{k-1}, x_k)$



# Cycles

## Cycle( $x_0$ )

**Données** : un sommet  $x_0$

**Résultat** : Un cycle (élémentaire)

$k \leftarrow 0$     $i \leftarrow x_0$     $C \leftarrow (x_0)$

**tant que** *il existe  $j \notin C$  tel que  $ij \in E$*  **faire**

$k \leftarrow k + 1$

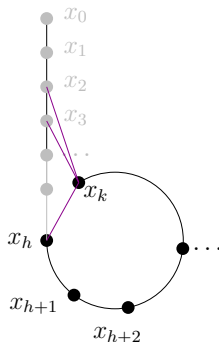
$x_k \leftarrow j$

    ajouter  $x_k$  en queue de  $C$

$i \leftarrow j$

Soit  $x_h$  avec  $h < k - 1$  un voisin de  $x_k$  dans  $C$

**retourner**  $(x_k, x_h, x_{h+1}, \dots, x_{k-1}, x_k)$



# Cycles

- 1 *tout s'exécute correctement*

# Cycles

- ❶ *tout s'exécute correctement*
  - Tous les voisins de  $x_k$  sont dans  $C$   
(sortie du **tant que** donc toutes les arêtes  $x_k x$  vérifient  $x \in C$ )
  - $d(x_k) \geq 2$  donc  $x_k$  a dans  $C$  un voisin  $\neq x_{k-1}$  (noté  $x_h$ )

# Cycles

- ❶ *tout s'exécute correctement*
  - Tous les voisins de  $x_k$  sont dans  $C$   
(sortie du **tant que** donc toutes les arêtes  $x_k x$  vérifient  $x \in C$ )
  - $d(x_k) \geq 2$  donc  $x_k$  a dans  $C$  un voisin  $\neq x_{k-1}$  (noté  $x_h$ )
- ❷ *en un nombre fini d'étapes*
- ❸ *en cas d'arrêt, on obtient l'objet souhaité*

# Cycles

- ❶ *tout  $s$  s'exécute correctement*
  - Tous les voisins de  $x_k$  sont dans  $C$   
(sortie du **tant que** donc toutes les arêtes  $x_k x$  vérifient  $x \in C$ )
  - $d(x_k) \geq 2$  donc  $x_k$  a dans  $C$  un voisin  $\neq x_{k-1}$  (noté  $x_h$ )
- ❷ *en un nombre fini d'étapes*
- ❸ *en cas d'arrêt, on obtient l'objet souhaité*
  - $C$  est une chaîne élémentaire : chaque sommet au plus une fois et  $x_i x_{i+1} \in E$
  - $x_k x_h \in E$  et  $x_k x_h$  n'est pas une arête de  $C$  ( $x_h \neq x_{k-1}$ ) et les deux extrémités du résultat sont identiques donc c'est un cycle

Remarque : le cycle obtenu est élémentaire.

# Cycles

Cette propriété simple implique qu'un graphe sans cycle possède au moins un sommet de degré 0 ou 1 (en passant par la contraposée).

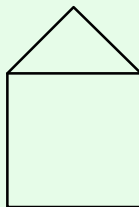


# Plan

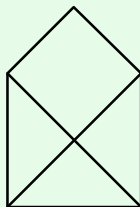
- 1 Chaînes
- 2 Connexité
- 3 Accessibilité
- 4 Parcours
- 5 Cycles
- 6 Graphes eulériens**
- 7 Graphes hamiltoniens

# Chaîne eulérienne

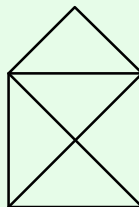
Quels dessins peuvent être dessinés sans lever le crayon et sans passer deux fois sur le même trait ?



(a)



(b)



(c)

# Chaîne eulérienne

## Définition

Une **chaîne eulérienne** de  $G = (V, E)$  est une chaîne qui emprunte chaque arête de  $G$  une et une seule fois.

*Remarque* : la chaîne peut ne pas être élémentaire, c'est-à-dire qu'elle peut passer plusieurs fois par le même sommet.

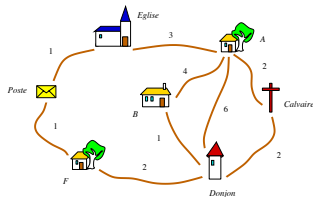
# Chaîne eulérienne

**Des exemples ?**

# Chaîne eulérienne

## Des exemples ?

- Sablage des routes
- Ramassage des ordures
- Tournée du facteur
- Faire toutes les pistes d'une station de ski



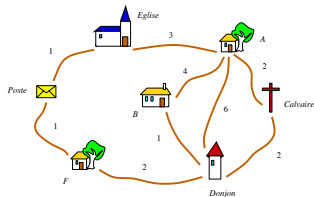
# Chaîne eulérienne

## Des exemples ?

- Sablage des routes
- Ramassage des ordures
- Tournée du facteur
- Faire toutes les pistes d'une station de ski

Mais aussi

- Sculpture ballon
- Reconstitution de séquences d'ADN



# Modélisons...



# Modélisons...



Maths en ballons (feat. Leonhard le chien) - Micmaths



# Euler, 1707-1783

## Les ponts de Königsberg

Problème fondateur de la théorie des graphes, Euler, 1736

La ville de Königsberg (aujourd'hui Kaliningrad) est construite autour de deux îles reliées entre elles par un pont et six ponts relient le continent à l'une ou l'autre des deux îles.

Existe-t-il une promenade dans les rues de Königsberg permettant, à partir d'un point de départ au choix, de passer une et une seule fois par chaque pont étant entendu qu'on ne peut traverser le Pregel qu'en passant sur les ponts ?



# Euler, 1707-1783

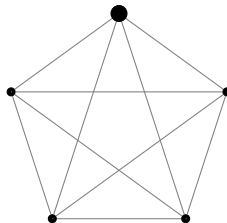
Il a même sa tête sur les billets de banque Suisses.



# Cycles eulériens

## Définition

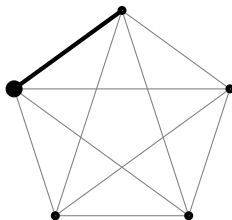
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

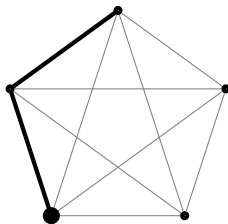
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

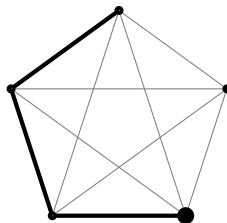
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

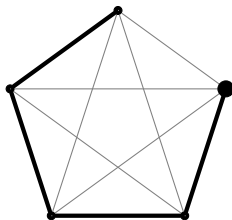
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

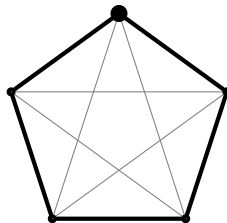
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .

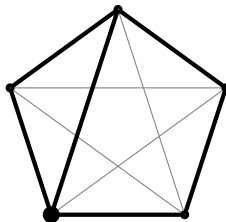




# Cycles eulériens

## Définition

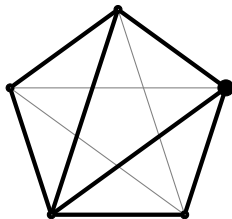
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

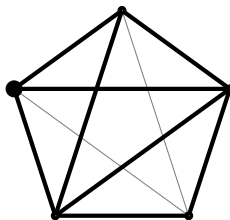
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

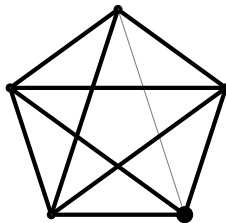
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

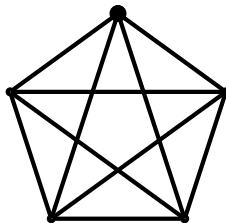
- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



# Cycles eulériens

## Définition

- Un **cycle eulérien** de  $G = (V, E)$  est un cycle qui emprunte chaque arête de  $G$  une et une seule fois.
- Un graphe  $G$  est **eulérien** s'il existe un cycle eulérien dans  $G$ .



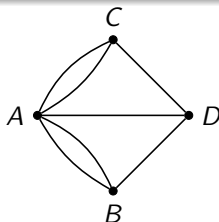
# Caractérisation des graphes eulériens

## Théorème (Euler 1736)

Un graphe connexe  $G$  est eulérien si et seulement tout sommet de  $G$  est de degré pair.

## Remarques

- Euler a prouvé que si un graphe connexe est eulérien alors tous les degrés sont pairs, il a ainsi donné une réponse négative au problème des sept ponts.
- si un graphe est eulérien, alors il est connexe, sauf éventuellement quelques sommets isolés.



# Cycle eulérien

“ $\Rightarrow$ ” : soit  $G$  un graphe connexe eulérien.

Soit  $v \in V$  un sommet et soit  $k$  le nombre de fois où il apparaît dans le cycle eulérien  $C$ .

$C$  contient  $k$  arêtes de la forme  $(x, v)$  (le cycle rentre  $k$  fois dans  $v$ ) et  $k$  arêtes de la forme  $(v, x)$  (le cycle sort  $k$  fois de  $v$ ).

Comme  $C$  est simple (pas d'arêtes en double),  $d(v) = 2k$  est pair.

# Cycle eulérien

“ $\Leftarrow$ ” : par l'absurde, supposons qu'il existe des graphes connexes dont tous les degrés sont pairs qui n'admettent pas de cycle eulérien.

- Soit  $G$  un tel graphe
- Soit  $C$  une plus grande chaîne simple (pas de répétition d'arête) de  $G$  (en nombre d'arêtes).



# Cycle eulérien

**Cas 1**  $C$  n'est pas un cycle.

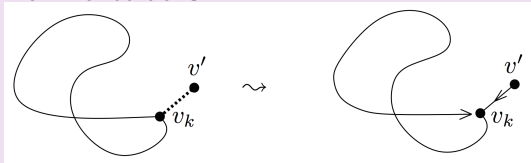
$C$  est une chaîne. Les extrémités d'une chaîne ont un degré impair dans la chaîne. Comme le degré du sommet d'arrivée est pair, on a une arête qui en sort et qui n'est pas dans la chaîne. On peut donc rajouter cette arête et ainsi rallonger la chaîne. contradiction avec la maximalité de  $C$  ❌.

# Cycle eulérien

**Cas 2**  $C = (v_0, v_1 \dots v_p, v_0)$  est un cycle. Il n'est pas eulérien par hypothèse.

- **Cas 2.1** Il existe un sommet qui n'est pas dans  $C$

Comme le graphe est connexe, il existe une arête  $v_k v'$  avec  $v' \notin C$ . La chaîne  $(v', v_k, v_{k+1} \dots v_p, v_0 \dots v_{k-1}, v_k)$  contredit la maximalité de  $C$ .

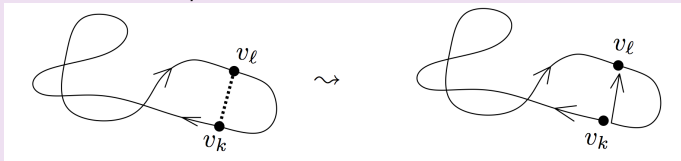


Dessins de *Invitation to mathematics* de Matoušek et Nešetřil

# Cycle eulérien

- Cas 2.2** Tous les sommets sont dans  $C$

Il existe une arête  $v_l v_k$  qui n'est pas dans  $C$ . La chaîne  $(v_k, v_{k-1} \dots v_0, v_p \dots v_{k+1}, v_k, v_l)$  contredit la maximalité de  $C$ .



Dessins de *Invitation to mathematics* de Matoušek et Nešetřil

# Graphes eulériens

**Et si le graphe n'est pas eulérien ?**

- Dans quel cas a-t-on une chaîne eulérienne ?

# Graphes eulériens

## Et si le graphe n'est pas eulérien ?

- Dans quel cas a-t-on une chaîne eulérienne ?

### Théorème

Un graphe connexe  $G$  contient une chaîne eulérienne si et seulement si dans  $G$ , le nombre de sommets de degré impair est 0 ou 2.

- si 0 sommets de degré impair, on a un graphe eulérien
- si 2 sommets de degré impair, ces 2 sommets seront les extrémités de toute chaîne eulérienne dans  $G$ .

# Graphes eulériens

## Et si le graphe n'est pas eulérien ?

- Dans quel cas a-t-on une chaîne eulérienne ?

### Théorème

Un graphe connexe  $G$  contient une chaîne eulérienne si et seulement si dans  $G$ , le nombre de sommets de degré impair est 0 ou 2.

- si 0 sommets de degré impair, on a un graphe eulérien
- si 2 sommets de degré impair, ces 2 sommets seront les extrémités de toute chaîne eulérienne dans  $G$ .
- Minimiser le nombre de chaînes pour parcourir toutes les arêtes (facile ?)

# Graphes eulériens

## Et si le graphe n'est pas eulérien ?

- Dans quel cas a-t-on une chaîne eulérienne ?

### Théorème

Un graphe connexe  $G$  contient une chaîne eulérienne si et seulement si dans  $G$ , le nombre de sommets de degré impair est 0 ou 2.

- si 0 sommets de degré impair, on a un graphe eulérien
- si 2 sommets de degré impair, ces 2 sommets seront les extrémités de toute chaîne eulérienne dans  $G$ .
- Minimiser le nombre de chaînes pour parcourir toutes les arêtes (facile ?)
- Si le graphe est connexe autoriser à passer plusieurs fois par certaines arêtes ; Minimiser la distance inutile parcourue (postier chinois)

# Multigraphe eulérien

La caractérisation des graphes eulériens s'étend en fait aux multigraphes sans boucle. Le degré  $d(v)$  d'un sommet  $v$  est alors le nombre d'arêtes incidentes à  $v$ .

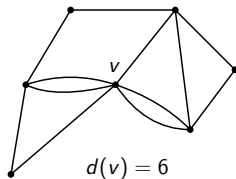
## Théorème

Un multigraphe  $G$  sans boucle est eulérien si et seulement si  $G$  est connexe (sauf éventuellement des sommets isolés), et tout sommet de  $G$  est de degré pair.

*Remarque :* Les boucles éventuelles sur les sommets non-isolés peuvent être insérées à n'importe quel moment sur un cycle eulérien<sup>a</sup>.

---

a. Ou, de manière équivalente, une boucle sur un sommet augmente son degré de 2.





# Multigraphe eulérien

La caractérisation des graphes eulériens s'étend en fait aux multigraphes sans boucle. Le degré  $d(v)$  d'un sommet  $v$  est alors le nombre d'arêtes incidentes à  $v$ .

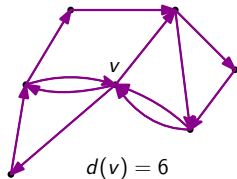
## Théorème

Un multigraphe  $G$  sans boucle est eulérien si et seulement si  $G$  est connexe (sauf éventuellement des sommets isolés), et tout sommet de  $G$  est de degré pair.

*Remarque :* Les boucles éventuelles sur les sommets non-isolés peuvent être insérées à n'importe quel moment sur un cycle eulérien<sup>a</sup>.

---

a. Ou, de manière équivalente, une boucle sur un sommet augmente son degré de 2.



# Multigraphe eulérien

La caractérisation des graphes eulériens s'étend en fait aux multigraphes sans boucle. Le degré  $d(v)$  d'un sommet  $v$  est alors le nombre d'arêtes incidentes à  $v$ .

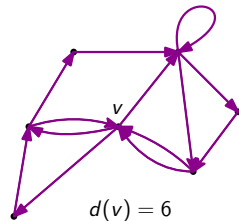
## Théorème

Un multigraphe  $G$  sans boucle est eulérien si et seulement si  $G$  est connexe (sauf éventuellement des sommets isolés), et tout sommet de  $G$  est de degré pair.

*Remarque :* Les boucles éventuelles sur les sommets non-isolés peuvent être insérées à n'importe quel moment sur un cycle eulérien <sup>a</sup>.

---

a. Ou, de manière équivalente, une boucle sur un sommet augmente son degré de 2.



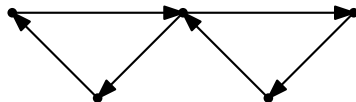
# Graphe orienté eulérien

La caractérisation des graphes eulériens s'étend aussi aux graphes orientés<sup>1</sup>.

- degré entrant  $d^-(v)$  : nombre d'arcs  $a = \overrightarrow{uv}$
- degré sortant  $d^+(v)$  : nombre d'arcs  $a = \overrightarrow{vw}$

## Théorème

Un graphe orienté  $\vec{G}$  admet un circuit eulérien si et seulement si le graphe non-orienté sous-jacent  $G$  est connexe (sauf éventuellement des sommets isolés), et dans  $\vec{G}$  tout sommet  $v$  vérifie  $d^-(v) = d^+(v)$ .



1. En français, un graphe orienté peut admettre un digon

# Graphe orienté eulérien

La caractérisation des graphes eulériens s'étend aussi aux graphes orientés<sup>1</sup>.


- degré entrant  $d^-(v)$  : nombre d'arcs  $a = \overrightarrow{uv}$
- degré sortant  $d^+(v)$  : nombre d'arcs  $a = \overrightarrow{vw}$

## Théorème

Un graphe orienté  $\vec{G}$  admet un circuit eulérien si et seulement si le graphe non-orienté sous-jacent  $G$  est connexe (sauf éventuellement des sommets isolés), et dans  $\vec{G}$  tout sommet  $v$  vérifie  $d^-(v) = d^+(v)$ .

Faire la preuve en vous inspirant du cas non-orienté.

---

1. En français, un graphe orienté peut admettre un digon .

# Plan

- 1 Chaînes
- 2 Connexité
- 3 Accessibilité
- 4 Parcours
- 5 Cycles
- 6 Graphes eulériens
- 7 Graphes hamiltoniens**

# Cycle Hamiltonien

## Définitions

Un **cycle hamiltonien** de  $G = (V, E)$  est un cycle qui emprunte chaque sommet de  $G$  une et une seule fois.

Un graphe est **hamiltonien** si et seulement si il contient un cycle hamiltonien.

*Intérêt* : Tâches à ordonnancer avec set-up, voyageur de commerce...

# Cycle Hamiltonien

## Définitions

Un **cycle hamiltonien** de  $G = (V, E)$  est un cycle qui emprunte chaque sommet de  $G$  une et une seule fois.

Un graphe est **hamiltonien** si et seulement si il contient un cycle hamiltonien.

*Intérêt* : Tâches à ordonnancer avec set-up, voyageur de commerce...

**Attention !** Définition qui ressemble beaucoup aux graphes eulériens.... mais problème très difficile ! Pas de bonne caractérisation.

Proposez un graphe simple d'ordre 5 pour chaque cas suivant :

- ① G1 est hamiltonien et eulérien ;
- ② G2 est hamiltonien et non eulérien ;
- ③ G3 est non hamiltonien et eulérien ;
- ④ G4 est non hamiltonien et non eulérien ;



Combien de fois devez vous lever le stylo au minimum pour reproduire ces formes inspirées de Kandinsky sans passer deux fois sur le même trait ?



formes pour "Ensemble Multicolore" de Kandinsky