Cheminements

Nadia Brauner et Yann Kieffer

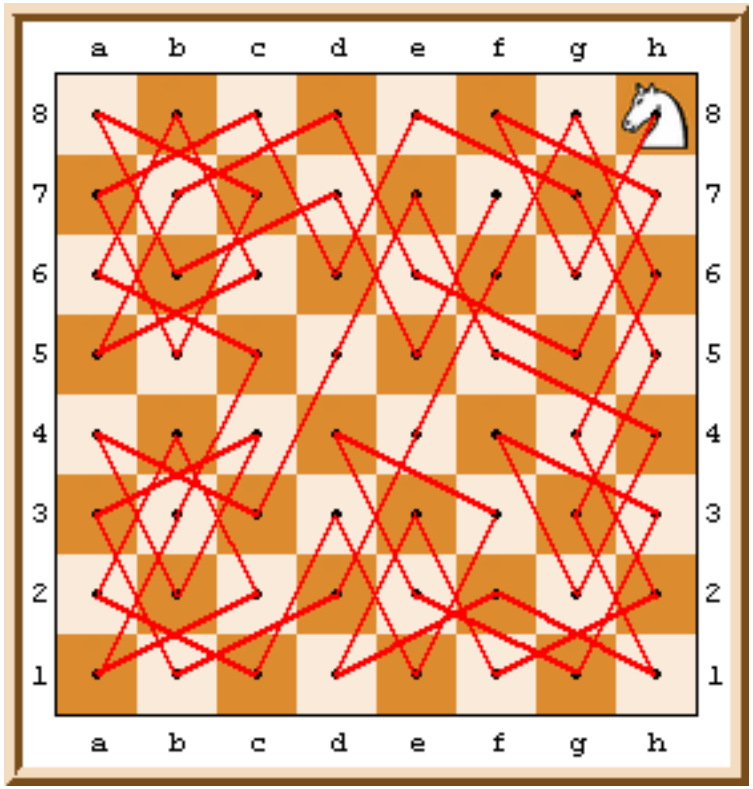
Nadia.Brauner@imag.fr

Plan

Plan

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

Chaîne : séquence alternée de sommets et d'arêtes

$(x_0, e_1, x_1, e_2 \dots x_{k-1}, e_k, x_k)$ où $x_i \in V$ $e_i = x_{i-1}x_i \in E$

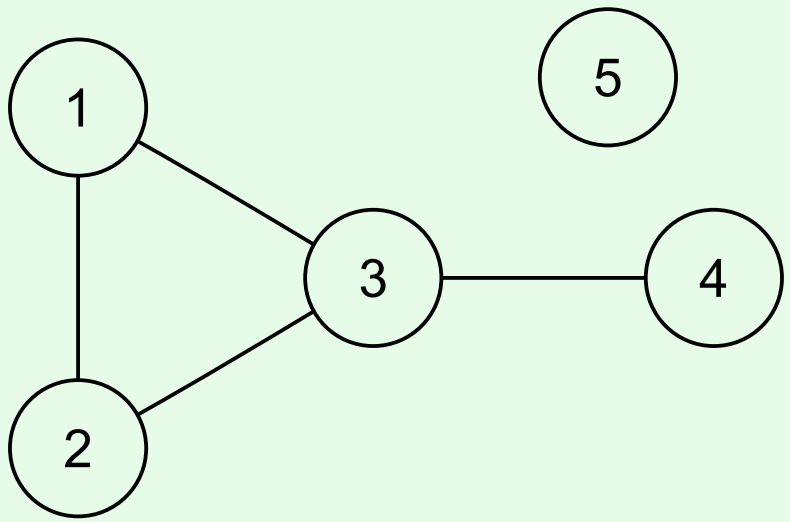
- Chaîne de x_0 à x_k
- x_0x_k-chaîne
- x_0, x_k sont les **extrémités** de la chaîne
- si pas d'ambiguïté : $(x_0, x_1, \dots x_{k-1}, x_k)$

Les x_i ne sont pas nécessairement distincts

Chaînes

Chaînes

Décrivez 10 chaînes dans le graphe suivant



- nombre infini de chaînes

Chaînes

Chaîne simple : pas de répétition d'arêtes
(tous les e_i distincts)

Chaîne élémentaire : chaque sommet apparaît une seule fois
(tous les x_i distincts, sauf éventuellement $x_0 = x_k$)

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

Longueur d'une chaîne = nb d'arêtes = k = nb de sommets -1

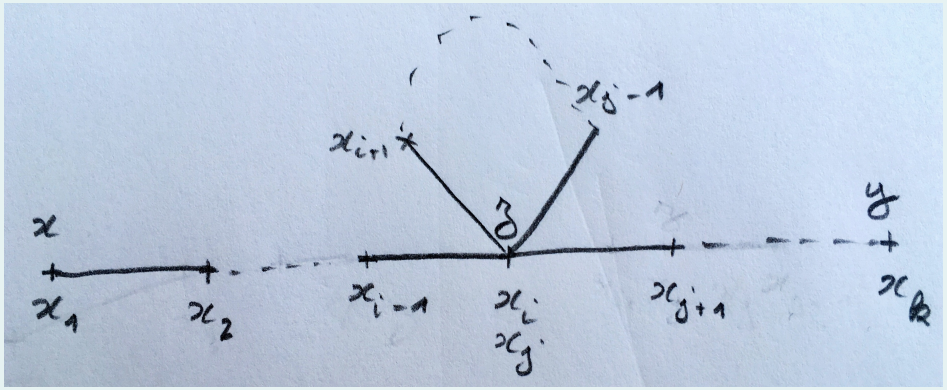
Chaînes

$\exists$ 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.
    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.
    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

vérifier les conditions aux bord si $x = z$ par exemple

Chaînes

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

Preuve alternative, mais identique en fait

Parmi toutes les chaînes reliant x à y, choisissons une chaîne $C = (x_1 = x, x_2 ... x_k = y)$ 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. 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 $C' = (x_1 = x ... x_{i-1}, z, x_{j+1} ... x_k = y)$ 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

Connexité

$G = (V, E)$ **connexe** ssi pour tout $x, y \in V, x \neq y$, il existe dans G une chaîne de x à y

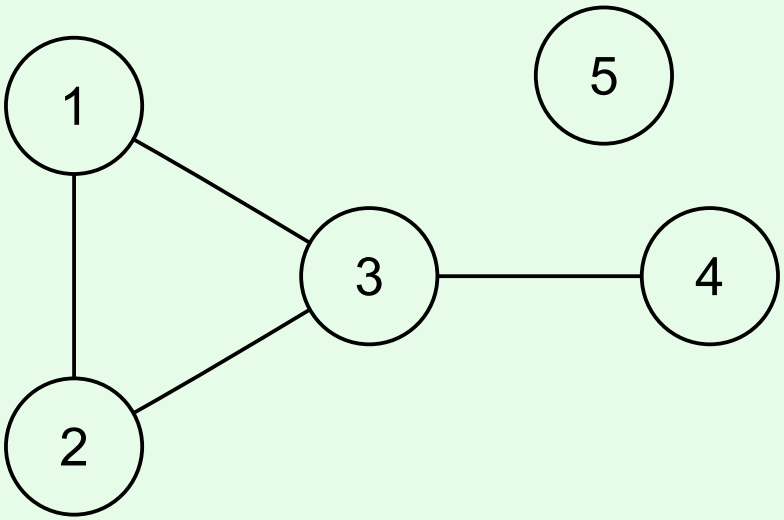
$x \sim y$ s'il existe une chaîne de x à y

$\sim$ est une **relation d'équivalence**
(réflexive, symétrique et transitive).

Composantes connexes = classes d'équivalences pour $\sim$

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

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

Induction sur l'ordre du graphe
n = 1 La propriété est clairement vraie.

Supposons la propriété prouvée sur les graphes d'ordre inférieur à n. Considérons un graphe $G = (V, E)$ connexe d'ordre $n+1$.

Connexité

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 $G' = (V \backslash \{x\}, E \backslash \{e\})$.
Le graphe G' reste connexe : si l'on considère 2 sommets u et v, 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. 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 d'induction 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 n arêtes.

Connexité

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. $2|E|$ doit alors être supérieur à $2|V|$, ce qui permet de conclure.

Plan

Accessibilité

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

Co-cycle de S :

ensemble des arêtes ayant exactement une extrémité dans S

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

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

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 co-cycle$(S) = \emptyset$

Accessibilité

Theorem

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

Supposons qu'il existe une chaîne $C = x_0, x_1 ... x_k$ 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 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

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(s,t)

Données : deux sommets s et t

Résultat : Vrai ssi il existe une st-chaîne

retourner $t \in \mathit{Accessibles}(s)$

Accessibilité

Vérification de Accessibles(s)

1. *tout 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_chaîne(`s`,`t`)` qui pour tout sommet $t \in S$ renvoie une st-chaîne

Donc **Accessible(s)** retourne 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 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 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

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

Écrivez un algorithme **Reconstruire_Chaîne(*s*, *t*)** qui utilise *from* pour construire une *st*-chaîne

---

Reconstruire_Chaîne(*s*, *t*, *from*)

Données : deux sommets s et t, et *from*, le résultat de **Accessibles(*s*)**

Résultat : Un *st*-chaîne

$C \leftarrow (t) \quad j \leftarrow t$

tant que $j \neq s$ **faire**

  ajouter $from(j)$ en tête de C

  $j \leftarrow from(j)$

retourner C

Appelé uniquement si $t \in Accessibles(s)$

Accessibilité

Vérification de `Reconstruire_Chaîne(s,t)`

1. *tout s'exécute correctement*
   - $j \in S \Rightarrow \mathit{from}(j) \in S$
   - donc les valeurs successives de j sont dans S
   - donc $\mathit{from}(j)$ est bien défini
2. *en un nombre fini d'étapes*
   - Soit f la fonction qui à $j \in S$ associe l'étape de **Accessibles(**s**)** à laquelle j a été rajouté dans S.
   - f est décroissante le long de from : $f(j) < f(\mathit{from}(j))$.
   - Dans le **tant que**, $f(j)$ est strict. décroissante et positive.
3. *en cas d'arrêt, on obtient l'objet souhaité*
   - À chaque étape, C commence par j et on lui rajoute $\mathit{from}(j)$.
   - Or $(j, \mathit{from}(j)) \in E$ donc C est une chaîne
   - on initialise C avec t donc t est une extrémité de C
   - En sortie du **tant que**, C a aussi comme extrémité s
   - donc C est une st-chaîne

Accessibilité

Certificats si oui

Si $t \in$ **Accessibles(**s**)**, alors **Reconstruire_Chaîne(**s**,**t**)** renvoie une st-chaîne

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 co-cycle$(S) = \emptyset$

Plan

Parcours

Pour programmer un parcours, il faut définir l'ordre dans lequel on considère les sommets.

Deux parcours principaux :

- en **largeur**
  - BFS pour *Breadth-First Search*
  - S est géré comme une file (FIFO)
  - priorité aux sommets entrés en premier dans S
- en **profondeur**
  - DFS pour *Depth-First Search*
  - S est géré comme une pile (LIFO)
  - priorité aux sommets entrés en dernier dans S

Parcours

Les plus courtes chaînes (en nombre d'arêtes)

Soient s et t deux sommets

$S \subset V$ est une st-**coupe** si $s \in S$ et $t \notin S$

- Si S est une st-coupe alors chaque chaîne de s à t contient au moins une arête de co-cycle(S)
- Si on a une famille de k st-coupes de co-cycles deux à deux disjoints, alors une plus courte chaîne de s à t a une longueur supérieure à k

Est-ce que la réciproque est vraie ?
Si on a une plus courte chaîne de s à t de longueur supérieure à k alors il existe une famille de k st-coupes de co-cycles deux a deux disjoints

Parcours

BFS pour les plus courtes chaînes (en nombre d'arêtes)

Étendre BFS pour trouver une plus courte chaîne de s à t et une famille de coupes comme certificat

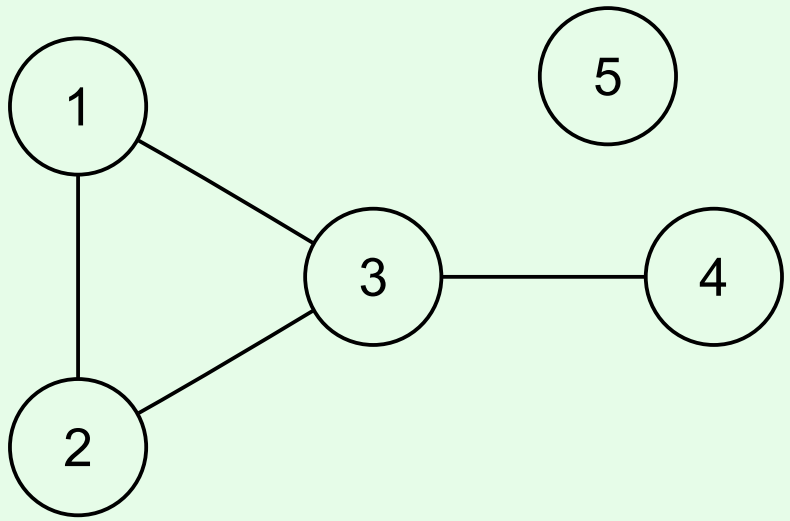
Adapter BFS pour trouver le plus court chemin entre deux ensembles S et T de sommets : on cherche un chemin ayant une extrémité dans S, une extrémité dans T et de longueur minimum (peut-être zéro si $S \cap T \neq \emptyset$).

Cycles

Cycle : chaîne simple dont les deux extrémités sont le même sommet ($x_0 = x_k$)

Cycle élémentaire : chaque sommet apparaît une seule fois (sauf extrémités)

Quelle est la longueur minimale d'un cycle ?
Y-a-t-il dans le graphe suivant un cycle élémentaire ? Et un cycle non élémentaires ?



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

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

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 $(x_k, x_h, x_{h+1}, \ldots, x_{k-1}, x_k)$

Cycles

1. *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)
2. *en un nombre fini d'étapes*
3. *en cas d'arrêt, on obtient l'objet souhaité*
   - C est une chaîne élémentaire : chaque sommet au plus une fois ($x_h \neq x_{k-1}$) 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 et les deux extrémités du résultat sont identiques donc c'est un cycle

Cycles

Cette propriété simple implique qu'un graphe sans cycle possède au moins un sommet de degré 0 ou 1.