

Algorithms and Probability

Week 10

G08 - mkilic

8.V.2025

Overview

1. Miller Rabin & Fermat - Beispiel
2. Finden von Duplikate - Beispiel
3. Lange Pfade

Roadmap

1. Graphentheorie

- Zusammenhang
- Kreise
- Matchings
- Färbungen

2. W'keitstheorie

- Bedingte W'keit
- Unabhängigkeit
- (mehrere) Zufallsvariablen
- Diskrete Verteilungen
- Abschätzen von W'keiten
- Randomisierte Algorithmen

3. Algorithmen

- Lange-Bunte Pfade
- MaxFlow
- MinCut
- Kleinster umschliessender Kreis
- Konvexe Hülle

Lange Pfade

Wir versuchen lange Pfade zu finden. Es gibt einige Varianten:

- suche längsten Pfad zwischen zwei Knoten
- suche längsten Pfad in dem Graphen
- suche Pfad von einer bestimmten Länge B

Wir interessieren uns für die Entscheidungsvariante von dem Letzteren.

Lange Pfade: Problemstellung

LONG-PATH

Gegeben (G, B) , G ein Graph und $B \in \mathbb{N}_0$, stelle fest ob es einen Pfad der Länge B in G gibt.

Lange Pfade: Problemstellung

LONG-PATH

Gegeben (G, B) , G ein Graph und $B \in \mathbb{N}_0$, stelle fest ob es einen Pfad der Länge B in G gibt.

Länge eines Pfaden: Ein Pfad der Länge ℓ ist die Folge $\langle v_0, v_1, \dots, v_\ell \rangle$ mit $\{v_{i-1}, v_i\} \in E$ für alle $i \in [\ell]$ und *besteht aus $\ell + 1$ Knoten*.

LONG-PATH-Problem ist schwer

LONG-PATH ist $\mathcal{NP}$ – *Vollständig*. D.h. es existiert keinen (bisher gefundenen) Polynomzeitalgorithmus für LONG-PATH. Folgender Satz sagt warum:

Satz 3.1.

Falls wir LONG-PATH für Graphen mit n Knoten in $t(n)$ Zeit entscheiden können, dann können wir in $t(2n - 2) + O(n^2)$ Zeit entscheiden, ob ein Graph mit n Knoten einen Hamiltonkreis hat.

Reduktion auf Hamiltonkreis

- wähle einen beliebigen Knoten v aus
- ersetze alle Kanten inzident zu v : $\{v, w_1\}, \dots, \{v, w_{deg(v)}\}$ durch neue Kanten $\{w'_1, w_1\}, \dots, \{w'_{deg(v)}, w_{deg(v)}\}$
- die w'_i sind neue Knoten und haben $deg(w'_i) = 1$ für alle i .
- Jetzt hat der neue Graph G' genau $n - 1 + deg(v)$ Knoten also maximal $n - 1 + n - 1 = 2n - 2$ wie behauptet ¹

¹weil max Grad eines Knoten ist $n - 1$

Reduktion auf Hamiltonkreis

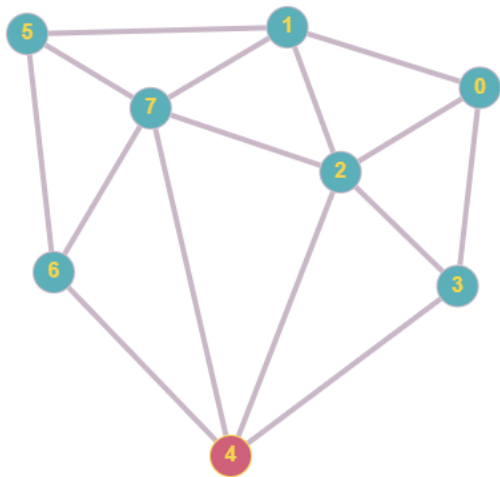
- wähle einen beliebigen Knoten v aus
- ersetze alle Kanten inzident zu v : $\{v, w_1\}, \dots, \{v, w_{deg(v)}\}$ durch neue Kanten $\{w'_1, w_1\}, \dots, \{w'_{deg(v)}, w_{deg(v)}\}$
- die w'_i sind neue Knoten und haben $deg(w'_i) = 1$ für alle i .
- Jetzt hat der neue Graph G' genau $n - 1 + deg(v)$ Knoten also maximal $n - 1 + n - 1 = 2n - 2$ wie behauptet ¹

Wir versuchen zu zeigen:

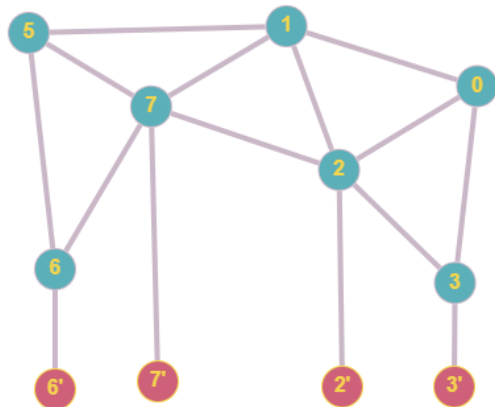
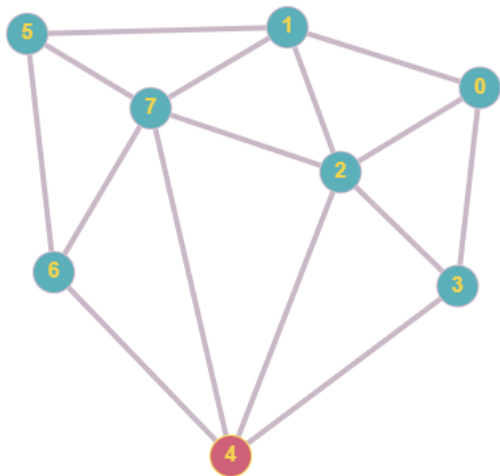
G hat einen Hamiltonkreis $\iff G'$ hat einen Pfad der Länge n

¹weil max Grad eines Knoten ist $n - 1$

Reduktion auf Hamiltonkreis



Reduktion auf Hamiltonkreis



Reduktion auf Hamiltonkreis

G hat einen Hamiltonkreis $\iff G'$ hat einen Pfad der Länge n

- $\Rightarrow$ Sei $\langle v_1, v_2, \dots, v_n, v_1 \rangle$ ein Hamiltonkreis in G . Dann für $v = v_1$ ist $\langle v'_2, v_2, \dots, v_n, v'_n \rangle$ ein Pfad der Länge n in G' .
- $\Leftarrow$ Sei $\langle u_0, u_1, \dots, u_n \rangle$ ein Pfad der Länge n in G' . Alle Knoten ausser u_0 und u_n haben mindestens Grad 2 also die sind die $n - 1$ Knoten aus G . Dann müssen u_0 und u_n zwei verschiedene neue Knoten in G' sein. Wir wissen deshalb dass u_1 und u_{n-1} zu v in G benachbart sind. Also ist

$$\langle v, u_1, \dots, u_{n-1}, v \rangle$$

ein Hamiltonkreis in G .

¹Die konstruktion von G' aus G geht in $O(n^2)$. Deshalb gilt die Laufzeit aus dem Satz 3.1.

Bunte Pfade & Color Coding

Lange-Pfade-Problem ist schwer. Wie sieht es aus mit B klein im Vergleich zu n wie z.B. $B = \log n$?

Der Algorithmus dafür benutzt eine randomisierte Methode, diese heisst: *Color-coding*
Dazu betrachten wir k -gefärbten Graphen. Diese Färbung muss keine *zulässige* Färbung sein. Wir haben eine Abbildung $\gamma : V \rightarrow [k]$

¹Engl. *bunter Pfad: colored path*

Bunte Pfade & Color Coding

Lange-Pfade-Problem ist schwer. Wie sieht es aus mit B klein im Vergleich zu n wie z.B. $B = \log n$?

Der Algorithmus dafür benutzt eine randomisierte Methode, diese heisst: *Color-coding*
Dazu betrachten wir k -gefärbten Graphen. Diese Färbung muss keine *zulässige* Färbung sein. Wir haben eine Abbildung $\gamma : V \rightarrow [k]$

Bunter Pfad

Ein Pfad in G heisst bunt, wenn alle Knoten auf dem Pfad verschiedene Farben haben.

¹Engl. *bunter Pfad: colored path*

Bunte Pfade Finden

Bunte Pfade Finden

Jetzt versuchen wir für einen gegebenen Graphen $G = (V, E)$ entscheiden ob es einen bunten Pfad der Länge $k - 1$ gibt.

Bunte Pfade Finden

Bunte Pfade Finden

Jetzt versuchen wir für einen gegebenen Graphen $G = (V, E)$ entscheiden ob es einen bunten Pfad der Länge $k - 1$ gibt.

- Ein bunter Pfad der Länge $k - 1$ benutzt k Farben.

Bunte Pfade Finden

Wir definieren

$$P_i(v) := \left\{ S \in \binom{[k]}{i+1} \mid \exists \text{ in } v \text{ endender genau mit } S \text{ gefärbter bunter Pfad} \right\}$$

- $P_i(v)$ ist eine Menge der Mengen von Farben
- $P_i(v)$ enthält $i + 1$ Farben
- $P_i(v) \neq \emptyset$ g.d.w. es einen bunten Pfad der Länge i gibt, der in v endet.
- Jede Menge $S \in P_i(v)$ muss $\gamma(v)$, die Farbe von v , enthalten.

Bunte Pfade Finden

Wir suchen einen bunten Pfad der Länge $k - 1$:

$$\exists \text{ bunter Pfad der Länge } k - 1 \iff \bigcup_{v \in V} P_{k-1}(v) \neq \emptyset$$

Für einen beliebigen Knoten v :

- $P_0(v) = \{\{\gamma(v)\}\}$
- $P_1(v) = \{\{\gamma(x), \gamma(v)\} \mid x \in N(v) \text{ und } \gamma(x) \neq \gamma(v)\}$
- Allgemein:

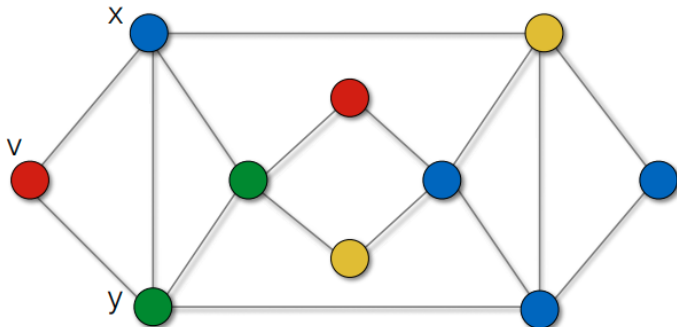
$$P_i(v) = \bigcup_{x \in N(v)} \{R \cup \{\gamma(v)\} \mid R \in P_{i-1}(x) \text{ und } \gamma(v) \notin R\}$$

Bunte Pfade Finden

Algorithm BUNT(G, i)

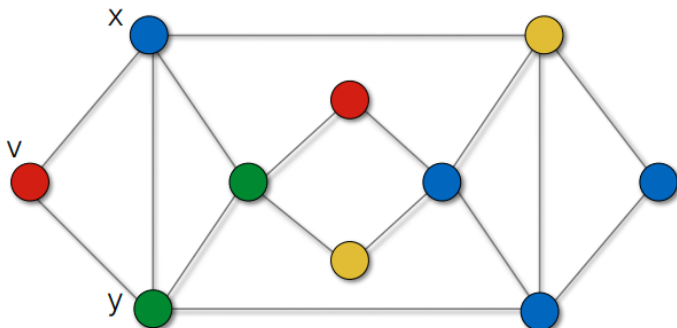
```
1: for all  $v \in V$  do  
2:    $P_i(v) \leftarrow \emptyset$   
3:   for all  $x \in N(v)$  do  
4:     for all  $R \in P_{i-1}(x)$  mit  $\gamma(v) \notin R$  do  
5:        $P_i(v) \leftarrow P_i(v) \cup \{R \cup \{\gamma(v)\}\}$ 
```

k=4, k = [red, yellow, blue, green]



Beispiel

$k=4$, $k = [\text{red}, \text{yellow}, \text{blue}, \text{green}]$



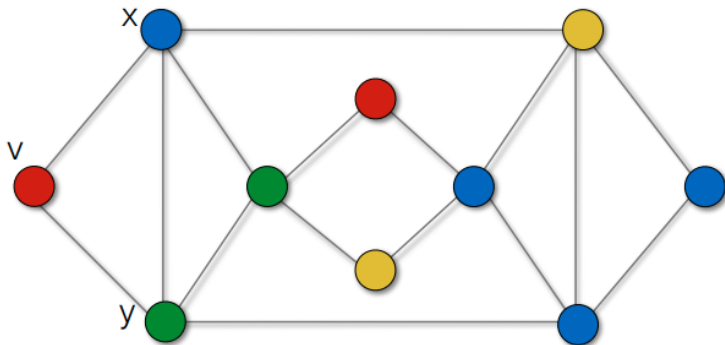
$$P_1(v) = \{ \{\text{red}, \text{blue}\}, \{\text{red}, \text{green}\} \}$$

$$P_1(x) = \{ \{\text{blue}, \text{red}\}, \{\text{blue}, \text{green}\}, \{\text{blue}, \text{yellow}\} \}$$

$$P_1(y) = \{ \{\text{green}, \text{red}\}, \{\text{green}, \text{blue}\} \}$$

Beispiel cont'd

$k=4$, $k = [\text{red}, \text{yellow}, \text{blue}, \text{green}]$



$$\begin{aligned}
 P_2(v) &= \bigcup_{w \in N(v)} \bigcup_{S \in P_1(w), \text{red} \notin S} \{S \cup \{\text{red}\}\} \\
 &= \underbrace{\{ \{\text{blue}, \text{green}\} \cup \{\text{red}\}, \{\text{blue}, \text{yellow}\} \cup \{\text{red}\} \}}_{P_1(x)} \cup \underbrace{\{ \{\text{green}, \text{blue}\} \cup \{\text{red}\} \}}_{P_1(y)}
 \end{aligned}$$

Laufzeitanalyse von Bunt(G, i)

- Jede Menge in $P_{i-1}(v)$ hat genau i Elemente.
- Da $P_{i-1}(v) \subseteq \binom{[k]}{i}$ gilt:

$$|P_{i-1}(v)| \leq \binom{k}{i}$$

- In der i -ten Runde wird für jeden Nachbarn $x \in N(v)$ und jede Menge $R \in P_{i-1}(x)$ geprüft, ob $\gamma(v) \notin R$.
- Der Aufwand pro Knoten v ist also:

$$\deg(v) \cdot \binom{k}{i} \cdot i$$

- Gesamtkosten pro Runde:

$$\mathcal{O} \left(\sum_{v \in V} \deg(v) \cdot \binom{k}{i} \cdot i \right) = \mathcal{O} \left(m \cdot \binom{k}{i} \cdot i \right)$$

wobei m die Anzahl der Kanten ist.

Gesamtkosten über alle Runden

- Es gibt $k - 1$ Runden (für $i = 1$ bis $k - 1$).
- Insgesamt ergibt sich der Aufwand:

$$\mathcal{O} \left(\sum_{i=1}^{k-1} \binom{k}{i} \cdot i \cdot m \right)$$

- Abschätzung:

$$\sum_{i=1}^{k-1} \binom{k}{i} \cdot i \leq \sum_{i=0}^k \binom{k}{i} \cdot i \leq k \cdot 2^k$$

- Damit ergibt sich:

$$\mathcal{O}(2^k \cdot k \cdot m)$$

Für $k = \mathcal{O}(\log n)$ haben wir einen polynomiellen Algorithmus für bunte Pfade.

Zufallsfärbungen

Wir wollen jetzt wieder entscheiden für einen Graph $G = (V, E)$ ob es in G einen Pfad der Länge B gibt.

Dazu färben wir die Knoten zufällig mit den $[k]$ Farben, wobei $k = B + 1$ und prüfen ob es einen bunten Pfad der Länge $k - 1$ gibt.

- Wenn es keinen Pfad der Länge $k - 1$ in G gibt, dann werden wir sicher keinen bunten Pfad finden.
- Wenn es einen Pfad der Länge $k - 1$ existiert, können wir Glück haben und in unserer zufälliger Färbung kann dieser Pfad bunt gefärbt sein. Wir können aber auch Pech haben und kein Pfad der Länge $k - 1$ wird bunt gefärbt.

Wir haben **einseitiger Fehler!**

Laufzeitanalyse - Zufallsfärbungen

Wenn G einen Pfad P der Länge $k - 1$ enthält, dann gibt es k^k mögliche Färbungen von P . Davon sind $k!$ bunt. Also ist die Erfolgswahrscheinlichkeit:

$$\begin{aligned} p_{\text{Erfolg}} &:= \Pr[\exists \text{ bunter Pfad der Länge } k - 1] \\ &\geq \Pr[P \text{ ist bunt}] = \frac{k!}{k^k} \geq e^{-k} \end{aligned}$$

Satz 3.2.

Sei G ein Graph mit einem Pfad der Länge $k - 1$.

1. Eine zufällige Färbung mit k Farben erzeugt einen bunten Pfad der Länge $k - 1$ mit Wahrscheinlichkeit

$$P_{\text{Erfolg}} \geq e^{-k}.$$

2. Bei wiederholten Färbungen mit k Farben ist der Erwartungswert der Anzahl Versuche, bis man einen bunten Pfad der Länge $k - 1$ erhält

$$\frac{1}{P_{\text{Erfolg}}} \leq e^k.$$

Algorithmus - lange Pfade

Wir bauen jetzt einen Monte Carlo Algorithmus, der unseren Test höchstens $\lceil \lambda e^k \rceil$ mal wiederholt. Wenn der Algorithmus in so vielen Runden die Bestätigung für einen Pfad der Länge $k - 1$ findet, antwortet er JA, sonst NEIN

Satz 3.3.

1. Der Algorithmus hat eine Laufzeit von

$$O(\lambda(2e)^k km).$$

2. Antwortet der Algorithmus mit JA, dann hat der Graph einen Pfad der Länge $k - 1$.
3. Hat der Graph einen Pfad der Länge $k - 1$, dann ist die Wahrscheinlichkeit, dass der Algorithmus mit NEIN antwortet, höchstens $e^{-\lambda}$.

¹Die Angaben aus diesem Satz erhält man durch Satz 2.74 mit $\varepsilon = e^{-k}$ und $\delta = e^{-\lambda}$

The End

