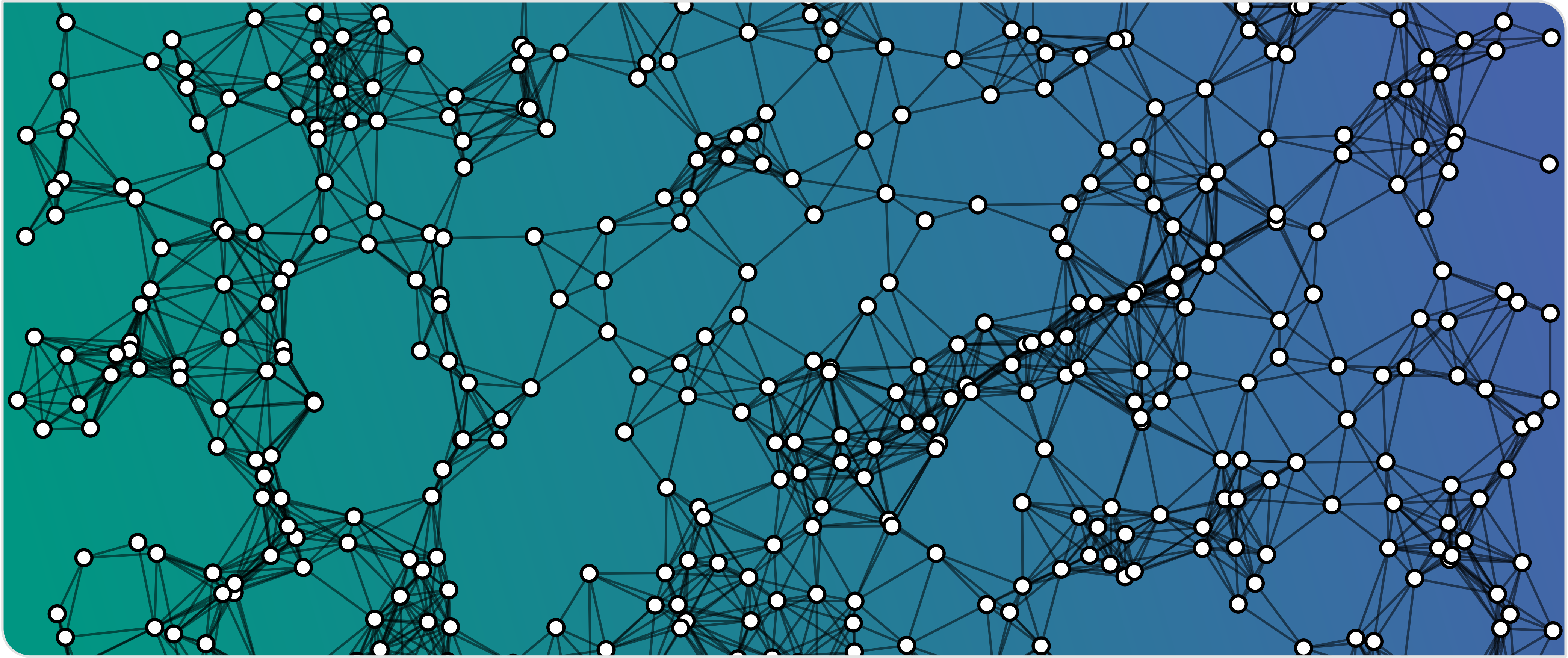
Karlsruher Institut für Technologie

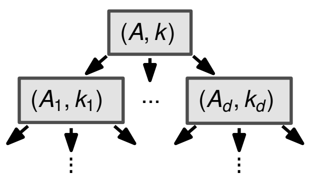
Parametrisierte Algorithmen

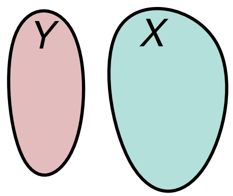
Color Coding

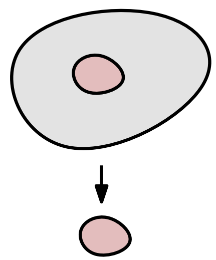


Inhalt

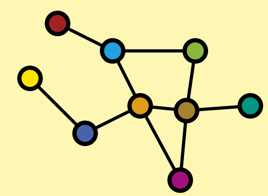
Basic Toolbox
- beschränkte Suchbäume
- iterative Kompression
- Kernbildung

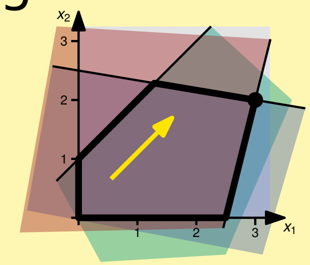


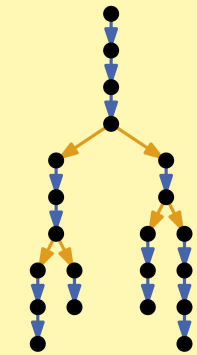




Erweiterte Toolbox
- lineare Programme
- Branch-and-Reduce
- Color Coding

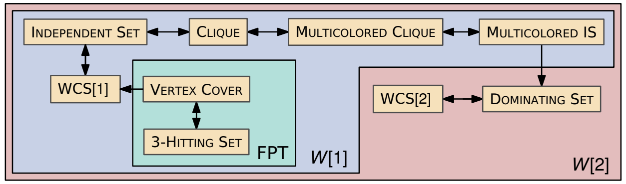


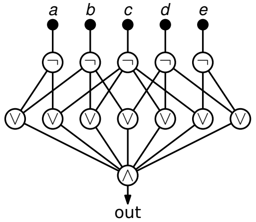




Baumweite
- dynamische Programme
- chordale & planare Graphen
- Courcelles Theorem

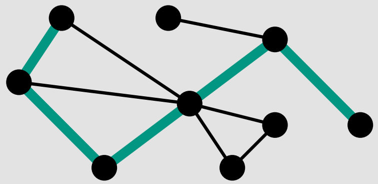
Untere Schranken
- parametrisierte Reduktionen
- boolesche Schaltkreise und die W-Hierarchie
- ETH und SETH





Umwege

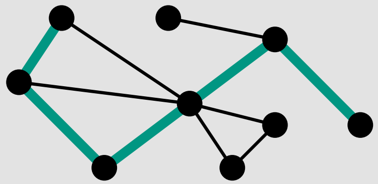
Problem: LONGEST PATH
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

Umwege

Problem: Longest Path
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



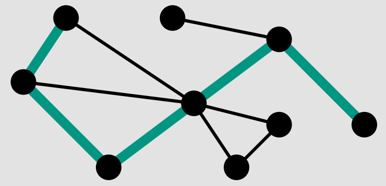
(mit Länge meine ich hier Anzahl Knoten)

Komplexität

- NP-schwer: Reduktion von Hamiltonian Path
- heute: FPT

Umwege

Problem: LONGEST PATH
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

Komplexität

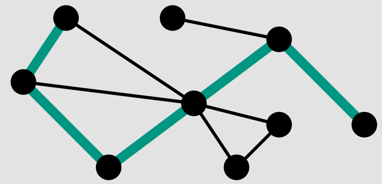
- NP-schwer: Reduktion von HAMILTONIAN PATH
- heute: FPT

Für gerichtete azyklische Graphen (DAGs)

Geht das in poly Zeit?

Umwege

Problem: Longest Path
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

Komplexität

- NP-schwer: Reduktion von Hamiltonian Path
- heute: FPT

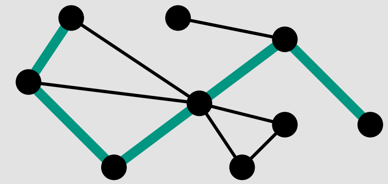
Für gerichtete azyklische Graphen (DAGs)

Geht das in poly Zeit?

- dynamisches Programm

Umwege

Problem: Longest Path
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

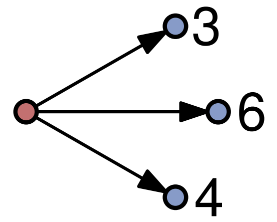
Komplexität

- NP-schwer: Reduktion von Hamiltonian Path
- heute: FPT

Für gerichtete azyklische Graphen (DAGs)

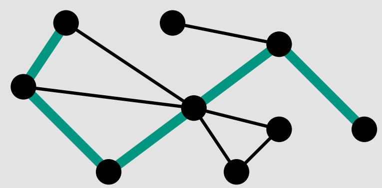
Geht das in poly Zeit?

- dynamisches Programm
  - für jeden Knoten v: längster Pfad der bei v startet
  - iteriere entsprechend topologischer Sortierung



Umwege

Problem: Longest Path
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



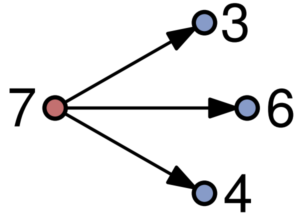
(mit Länge meine ich hier Anzahl Knoten)

Komplexität
- NP-schwer: Reduktion von Hamiltonian Path
- heute: FPT

Für gerichtete azyklische Graphen (DAGs)

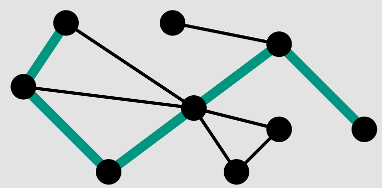
Geht das in poly Zeit?

- dynamisches Programm
  - für jeden Knoten v: längster Pfad der bei v startet
  - iteriere entsprechend topologischer Sortierung



Umwege

Problem: LONGEST PATH
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

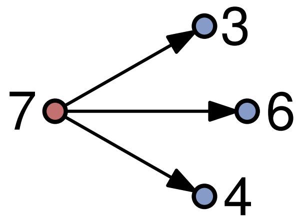
Komplexität

- NP-schwer: Reduktion von HAMILTONIAN PATH
- heute: FPT

Für gerichtete azyklische Graphen (DAGs)

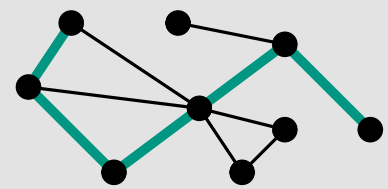
Geht das in poly Zeit?

- dynamisches Programm
  - für jeden Knoten v: längster Pfad der bei v startet
  - iteriere entsprechend topologischer Sortierung
- alternativ: Matrixmultiplikation



Umwege

Problem: LONGEST PATH
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

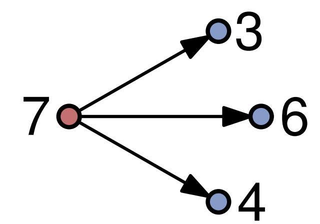
Komplexität

- NP-schwer: Reduktion von HAMILTONIAN PATH
- heute: FPT

Für gerichtete azyklische Graphen (DAGs)

Geht das in poly Zeit?

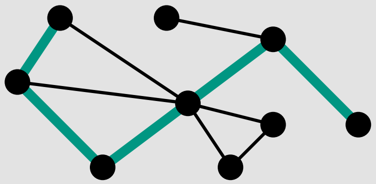
- dynamisches Programm
  - für jeden Knoten v: längster Pfad der bei v startet
  - iteriere entsprechend topologischer Sortierung



- alternativ: Matrixmultiplikation
  - sei A die Adjazenzmatrix und betrachte A^k
  - Eintrag (u, v) entspricht Anzahl Pfaden den Länge k von u nach v

Umwege

Problem: LONGEST PATH
Gegeben sind ein Graph G und ein Parameter k. Gibt es einen Pfad der Länge mindestens k in G?



(mit Länge meine ich hier Anzahl Knoten)

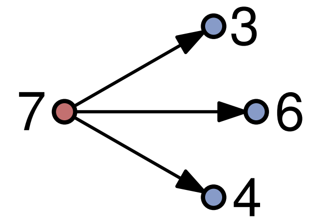
Komplexität

- NP-schwer: Reduktion von HAMILTONIAN PATH
- heute: FPT

Für gerichtete azyklische Graphen (DAGs)

Geht das in poly Zeit?

- dynamisches Programm
  - für jeden Knoten v: längster Pfad der bei v startet
  - iteriere entsprechend topologischer Sortierung



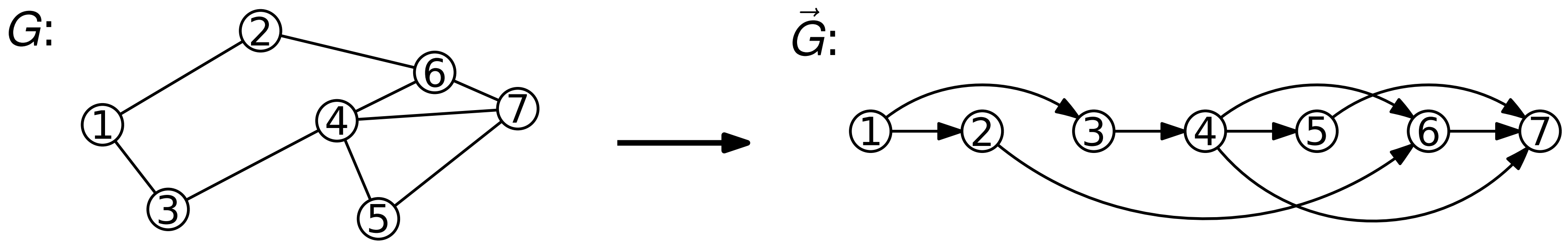
- alternativ: Matrixmultiplikation
  - sei A die Adjazenzmatrix und betrachte A^k
  - Eintrag (u, v) entspricht Anzahl Pfaden den Länge k von u nach v

Warum funktioniert das nicht auch für ungerichtete Graphen?

Kreise verbieten

Idee für ungerichtete Graphen

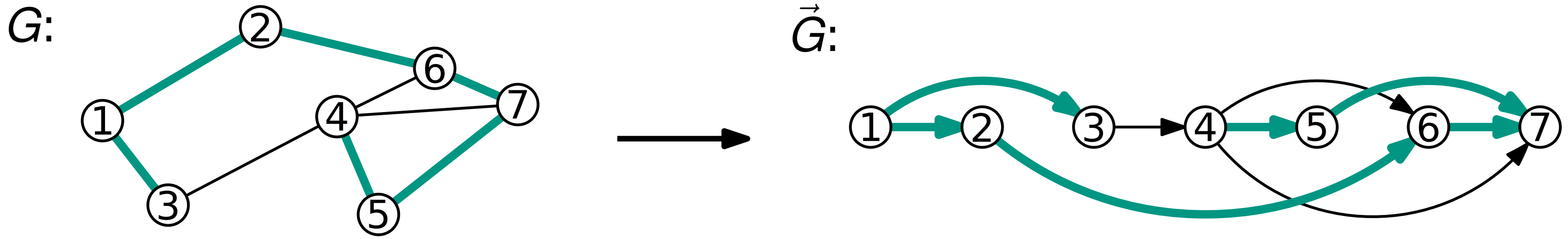
- verwandle ungerichteten Graph G in einen DAG $\vec{G}$
- suche dann einen längsten Pfad in $\vec{G}$



Kreise verbieten

Idee für ungerichtete Graphen

- verwandle ungerichteten Graph G in einen DAG $\vec{G}$
- suche dann einen längsten Pfad in $\vec{G}$

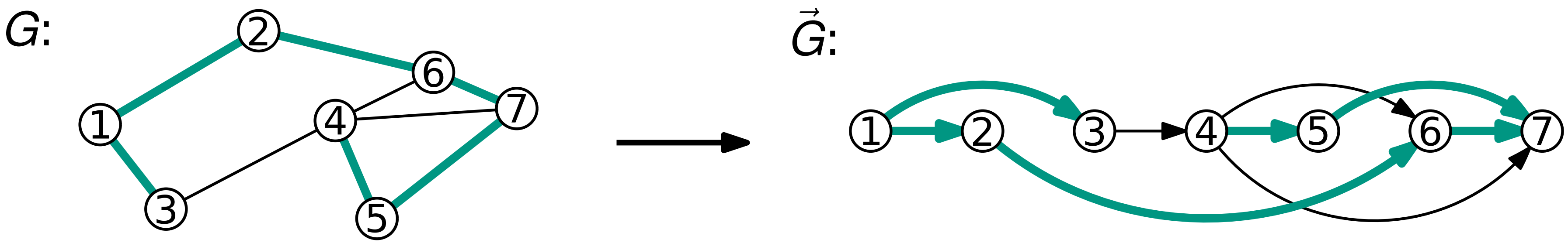


Problem: nicht alle Pfade in G sind korrekt gerichtete Pfade in $\vec{G}$

Kreise verbieten

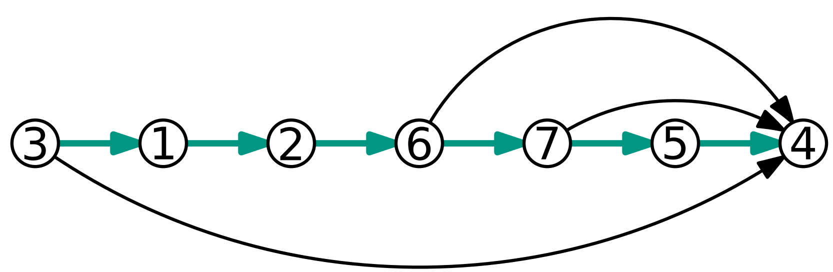
Idee für ungerichtete Graphen

- verwandle ungerichteten Graph G in einen DAG $\vec{G}$
- suche dann einen längsten Pfad in $\vec{G}$



Problem: nicht alle Pfade in G sind korrekt gerichtete Pfade in $\vec{G}$

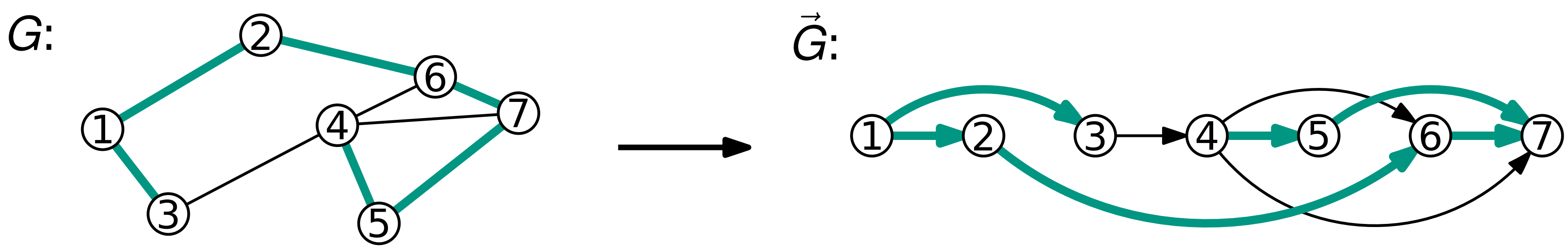
Aber: es gibt immer eine gute Übersetzung in einen DAG



Kreise verbieten

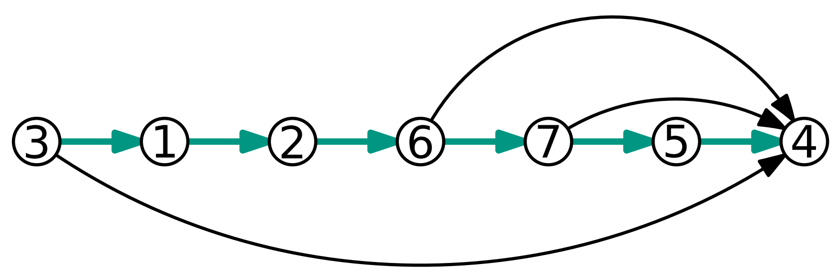
Idee für ungerichtete Graphen

- verwandle ungerichteten Graph G in einen DAG $\vec{G}$
- suche dann einen längsten Pfad in $\vec{G}$



Problem: nicht alle Pfade in G sind korrekt gerichtete Pfade in $\vec{G}$

Aber: es gibt immer eine gute Übersetzung in einen DAG



Wie finden wir eine gute Übersetzung in einen DAG?

Randomisierung

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

Randomisierung

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein gerichteter Pfad in $\vec{G}$ mit Wahrscheinlichkeit $\frac{2}{k!}$.

Randomisierung

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein gerichteter Pfad in $\vec{G}$ mit Wahrscheinlichkeit $\frac{2}{k!}$.

Beweis

- sei $P = v_1, \dots, v_k$

Randomisierung

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein gerichteter Pfad in $\vec{G}$ mit Wahrscheinlichkeit $\frac{2}{k!}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Ordnung dieser k Knoten ist gleich wahrscheinlich

Randomisierung

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein gerichteter Pfad in $\vec{G}$ mit Wahrscheinlichkeit $\frac{2}{k!}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Ordnung dieser k Knoten ist gleich wahrscheinlich
- also: $k!$ Ereignisse, die jeweils mit Wkt $\frac{1}{k!}$ auftreten

Randomisierung

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein gerichteter Pfad in $\vec{G}$ mit Wahrscheinlichkeit $\frac{2}{k!}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Ordnung dieser k Knoten ist gleich wahrscheinlich
- also: $k!$ Ereignisse, die jeweils mit Wkt $\frac{1}{k!}$ auftreten
- zwei dieser Ereignisse sind gut: $v_1, \dots, v_k$ und $v_k, \dots, v_1$

Randomisierung

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

> **Theorem**
> Sei P ein Pfad der Länge k in G. Dann ist P ein gerichteter Pfad in $\vec{G}$ mit Wahrscheinlichkeit $\frac{2}{k!}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Ordnung dieser k Knoten ist gleich wahrscheinlich
- also: $k!$ Ereignisse, die jeweils mit Wkt $\frac{1}{k!}$ auftreten
- zwei dieser Ereignisse sind gut: $v_1, \dots, v_k$ und $v_k, \dots, v_1$

Beachte

- Erfolgswahrscheinlichkeit hängt nur von k ab

Randomisierung

Karlsruher Institut für Technologie

Konstruktion des DAGs $\vec{G}$

- wähle zufällige Reihenfolge der Knoten in G
- richte alle Kanten in G von vorne nach hinten

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein gerichteter Pfad in $\vec{G}$ mit Wahrscheinlichkeit $\frac{2}{k!}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Ordnung dieser k Knoten ist gleich wahrscheinlich
- also: $k!$ Ereignisse, die jeweils mit Wkt $\frac{1}{k!}$ auftreten
- zwei dieser Ereignisse sind gut: $v_1, \dots, v_k$ und $v_k, \dots, v_1$

Beachte

- Erfolgswahrscheinlichkeit hängt nur von k ab
- Wahrscheinlichkeit kann durch Wiederholung geboosted werden
- Anzahl Wiederholungen nur von k abhängig (um konstante Wkt zu erreichen)

Randomisierter FPT Algorithmus

Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit Longest Path für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Randomisierter FPT Algorithmus

Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit Longest Path für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert

Randomisierter FPT Algorithmus

Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit LONGEST PATH für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert
- ja-Instanzen werden falsch erkannt mit Wahrscheinlichkeit:

Randomisierter FPT Algorithmus

Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit Longest Path für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert
- ja-Instanzen werden falsch erkannt mit Wahrscheinlichkeit:
  - $1 - \frac{2}{k!}$ für einen Rateversuch
  - $\left(1 - \frac{2}{k!}\right)^{k!}$ für $k!$ unabhängige Rateversuche

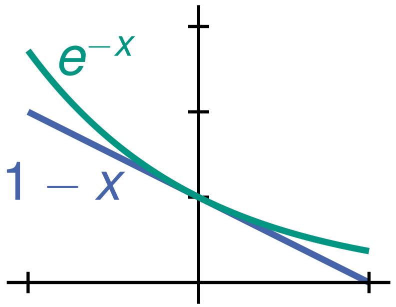
Randomisierter FPT Algorithmus

Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit LONGEST PATH für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert
- ja-Instanzen werden falsch erkannt mit Wahrscheinlichkeit:
  - $1 - \frac{2}{k!}$ für einen Rateversuch
  - $\left(1 - \frac{2}{k!}\right)^{k!}$ für $k!$ unabhängige Rateversuche



Randomisierter FPT Algorithmus

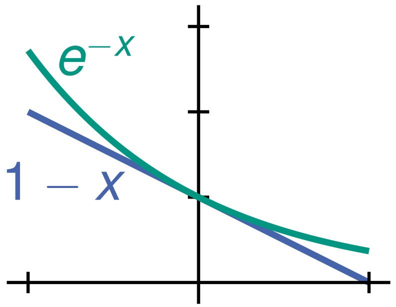
Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit LONGEST PATH für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert
- ja-Instanzen werden falsch erkannt mit Wahrscheinlichkeit:
  - $1 - \frac{2}{k!}$ für einen Rateversuch
  - $\left(1 - \frac{2}{k!}\right)^{k!}$ für $k!$ unabhängige Rateversuche

$$\left(1 - \frac{2}{k!}\right)^{k!} \leq \left(e^{-\frac{2}{k!}}\right)^{k!} = e^{-2} < \frac{1}{2}$$



Randomisierter FPT Algorithmus

Karlsruher Institut für Technologie

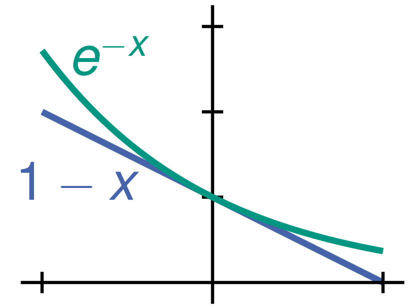
Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit LONGEST PATH für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert
- ja-Instanzen werden falsch erkannt mit Wahrscheinlichkeit:
  - $1 - \frac{2}{k!}$ für einen Rateversuch
  - $\left(1 - \frac{2}{k!}\right)^{k!}$ für $k!$ unabhängige Rateversuche

$$\left(1 - \frac{2}{k!}\right)^{k!} \leq \left(e^{-\frac{2}{k!}}\right)^{k!} = e^{-2} < \frac{1}{2}$$



Zusammenfassung

- Monte Carlo Algorithmus mit einseitigem Fehler

Randomisierter FPT Algorithmus

Karlsruher Institut für Technologie

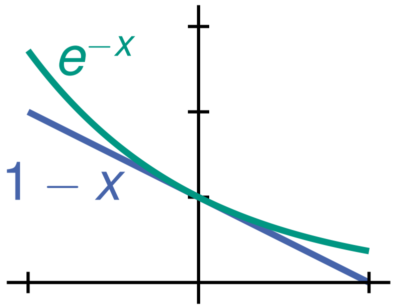
Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit LONGEST PATH für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert
- ja-Instanzen werden falsch erkannt mit Wahrscheinlichkeit:
  - $1 - \frac{2}{k!}$ für einen Rateversuch
  - $\left(1 - \frac{2}{k!}\right)^{k!}$ für $k!$ unabhängige Rateversuche

$$\left(1 - \frac{2}{k!}\right)^{k!} \leq \left(e^{-\frac{2}{k!}}\right)^{k!} = e^{-2} < \frac{1}{2}$$



Zusammenfassung

- Monte Carlo Algorithmus mit einseitigem Fehler
- n Wiederholungen um mit hoher Wkt korrektes Ergebnis zu erhalten

Randomisierter FPT Algorithmus

Karlsruher Institut für Technologie

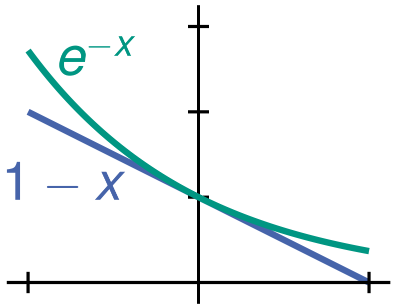
Erfolgswahrscheinlichkeit: $\frac{2}{k!}$

Theorem
Nach $k!$ unabhängigen Rateversuchen für $\vec{G}$ ist die Wahrscheinlichkeit LONGEST PATH für (G, k) korrekt zu entscheiden größer als $\frac{1}{2}$.

Beweis

- nein-Instanzen werden immer korrekt als solche identifiziert
- ja-Instanzen werden falsch erkannt mit Wahrscheinlichkeit:
  - $1 - \frac{2}{k!}$ für einen Rateversuch
  - $\left(1 - \frac{2}{k!}\right)^{k!}$ für $k!$ unabhängige Rateversuche

$$\left(1 - \frac{2}{k!}\right)^{k!} \leq \left(e^{-\frac{2}{k!}}\right)^{k!} = e^{-2} < \frac{1}{2}$$



Zusammenfassung

- Monte Carlo Algorithmus mit einseitigem Fehler
- n Wiederholungen um mit hoher Wkt korrektes Ergebnis zu erhalten
- Gesamtlaufzeit: $O(k!mn)$

Ein Schritt zurück

Was haben wir gerade gemacht?

- wünschen uns zusätzliche Struktur (Knotenordnung)
- die Struktur lässt uns das Problem in polynomieller Zeit lösen
- für jede Lösung: nur $f(k)$ viele Möglichkeiten für die Wahl der Struktur

Ein Schritt zurück

Was haben wir gerade gemacht?

- wünschen uns zusätzliche Struktur (Knotenordnung)
- die Struktur lässt uns das Problem in polynomieller Zeit lösen
- für jede Lösung: nur $f(k)$ viele Möglichkeiten für die Wahl der Struktur
- Wkt für ungünstig geratene Struktur hängt nur von k ab
- boosten der Erfolgswkt: nur von k abhängig viele Wiederholungen

Ein Schritt zurück

Karlsruher Institut für Technologie

Was haben wir gerade gemacht?

- wünschen uns zusätzliche Struktur (Knotenordnung)
- die Struktur lässt uns das Problem in polynomieller Zeit lösen
- für jede Lösung: nur $f(k)$ viele Möglichkeiten für die Wahl der Struktur
- Wkt für ungünstig geratene Struktur hängt nur von k ab
- boosten der Erfolgswkt: nur von k abhängig viele Wiederholungen

Dasselbe in Grün: Color Coding

- betrachte gefärbte Knoten
- suche nur bunte Lösungen

Ein Schritt zurück

Karlsruher Institut für Technologie

Was haben wir gerade gemacht?

- wünschen uns zusätzliche Struktur (Knotenordnung)
- die Struktur lässt uns das Problem in polynomieller Zeit lösen
- für jede Lösung: nur $f(k)$ viele Möglichkeiten für die Wahl der Struktur
- Wkt für ungünstig geratene Struktur hängt nur von k ab
- boosten der Erfolgswkt: nur von k abhängig viele Wiederholungen

Dasselbe in Grün: Color Coding

- betrachte gefärbte Knoten
- suche nur bunte Lösungen

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Ein Schritt zurück

Was haben wir gerade gemacht?

- wünschen uns zusätzliche Struktur (Knotenordnung)
- die Struktur lässt uns das Problem in polynomieller Zeit lösen
- für jede Lösung: nur $f(k)$ viele Möglichkeiten für die Wahl der Struktur
- Wkt für ungünstig geratene Struktur hängt nur von k ab
- boosten der Erfolgswkt: nur von k abhängig viele Wiederholungen

Dasselbe in Grün: Color Coding

- betrachte gefärbte Knoten
- suche nur bunte Lösungen

> **Problem: Colorful LONGEST PATH**
> Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
> (ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

- Struktur hilft: FPT-Algo für gefärbtes Problem
- zufällige Färbung $\rightarrow$ Fehlerwkt nur von k abhängig

Ein Schritt zurück

Was haben wir gerade gemacht?

- wünschen uns zusätzliche Struktur (Knotenordnung)
- die Struktur lässt uns das Problem in polynomieller Zeit lösen
- für jede Lösung: nur $f(k)$ viele Möglichkeiten für die Wahl der Struktur
- Wkt für ungünstig geratene Struktur hängt nur von k ab
- boosten der Erfolgswkt: nur von k abhängig viele Wiederholungen

Dasselbe in Grün: Color Coding

- betrachte gefärbte Knoten
- suche nur bunte Lösungen

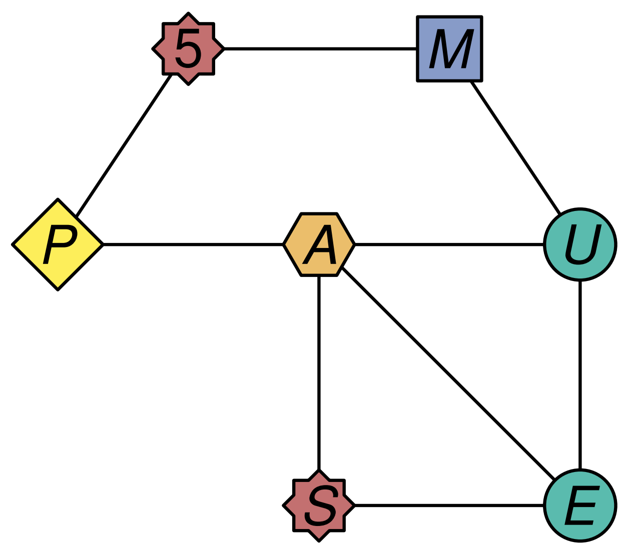
Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

- Struktur hilft: FPT-Algo für gefärbtes Problem
- zufällige Färbung $\rightarrow$ Fehlerwkt nur von k abhängig
- danach: Derandomisierung

Bunte Pfade

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

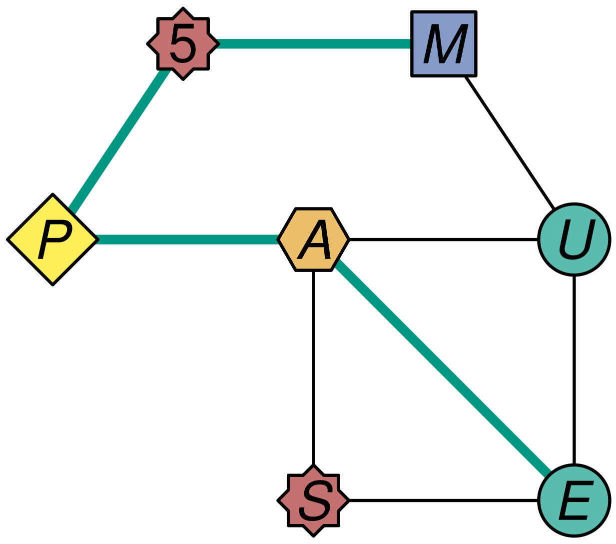
Wie lang ist der längste bunte Pfad?



Bunte Pfade

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \ldots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Wie lang ist der längste bunte Pfad?



Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten $\rightarrow n^\ell$ Möglichkeiten

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten $\rightarrow n^\ell$ Möglichkeiten
- besser: merke bisher besuchte Farben

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \ldots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten $\rightarrow n^\ell$ Möglichkeiten
- besser: merke bisher besuchte Farben $\rightarrow 2^k$ Möglichkeiten

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten $\rightarrow n^\ell$ Möglichkeiten
- besser: merke bisher besuchte Farben $\rightarrow 2^k$ Möglichkeiten

Dynamisches Programm

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten $\rightarrow n^\ell$ Möglichkeiten
- besser: merke bisher besuchte Farben $\rightarrow 2^k$ Möglichkeiten

Dynamisches Programm

- für jeden Knoten v: berechne für welche Farbmengen $S \subseteq [k]$ ein Pfad dieser Farben existiert, der in v startet

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten $\rightarrow n^\ell$ Möglichkeiten
- besser: merke bisher besuchte Farben $\rightarrow 2^k$ Möglichkeiten

Dynamisches Programm

- für jeden Knoten v: berechne für welche Farbmengen $S \subseteq [k]$ ein Pfad dieser Farben existiert, der in v startet
- betrachte dazu die Farbmengen in aufsteigender Größe

Wie helfen die Farben?

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)

Lösung für DAGs

- Pfad der Länge ℓ von v aus und $uv \in E \Rightarrow$ Pfad der Länge $\ell + 1$ von u aus

Ungerichtete Graphen

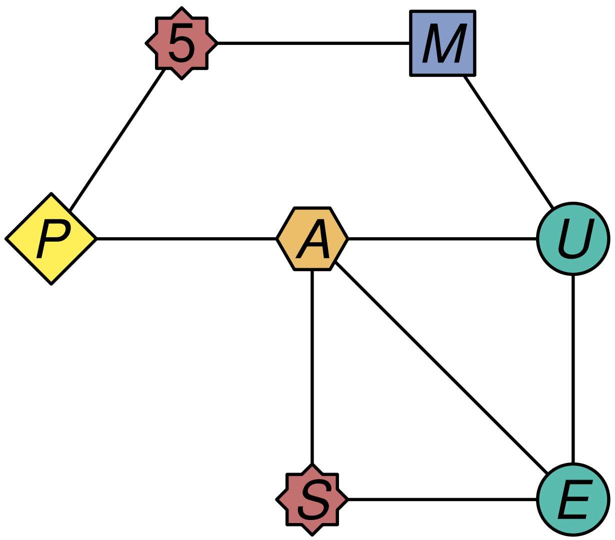
- Problem: Pfad der Länge ℓ von v aus könnte u schon enthalten
- Idee: merke zusätzlich bisher besuchte Knoten $\rightarrow n^\ell$ Möglichkeiten
- besser: merke bisher besuchte Farben $\rightarrow 2^k$ Möglichkeiten

Dynamisches Programm

- für jeden Knoten v: berechne für welche Farbmengen $S \subseteq [k]$ ein Pfad dieser Farben existiert, der in v startet
- betrachte dazu die Farbmengen in aufsteigender Größe
- Farbmengen der Größe $\ell + 1$ ergibt sich aus Farbmengen der Größe ℓ der Nachbarn

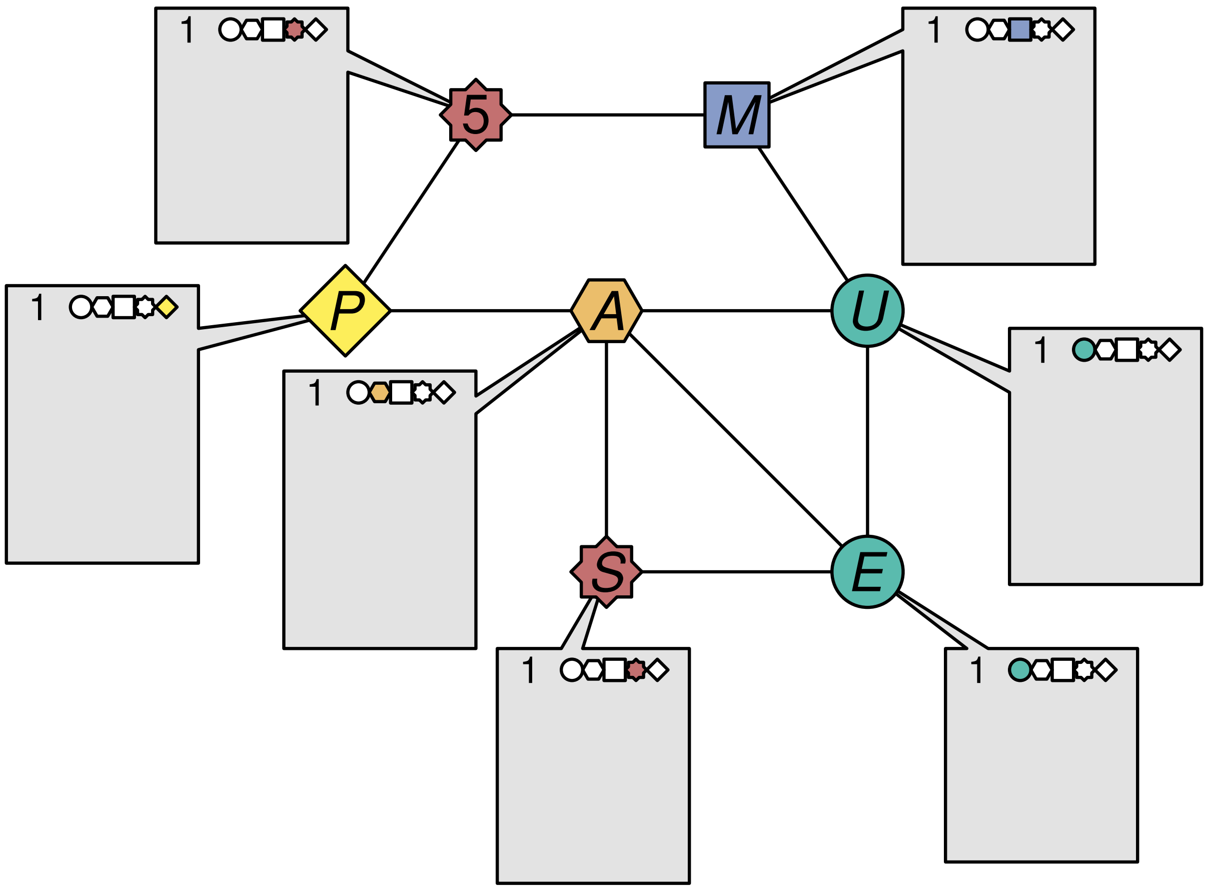
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



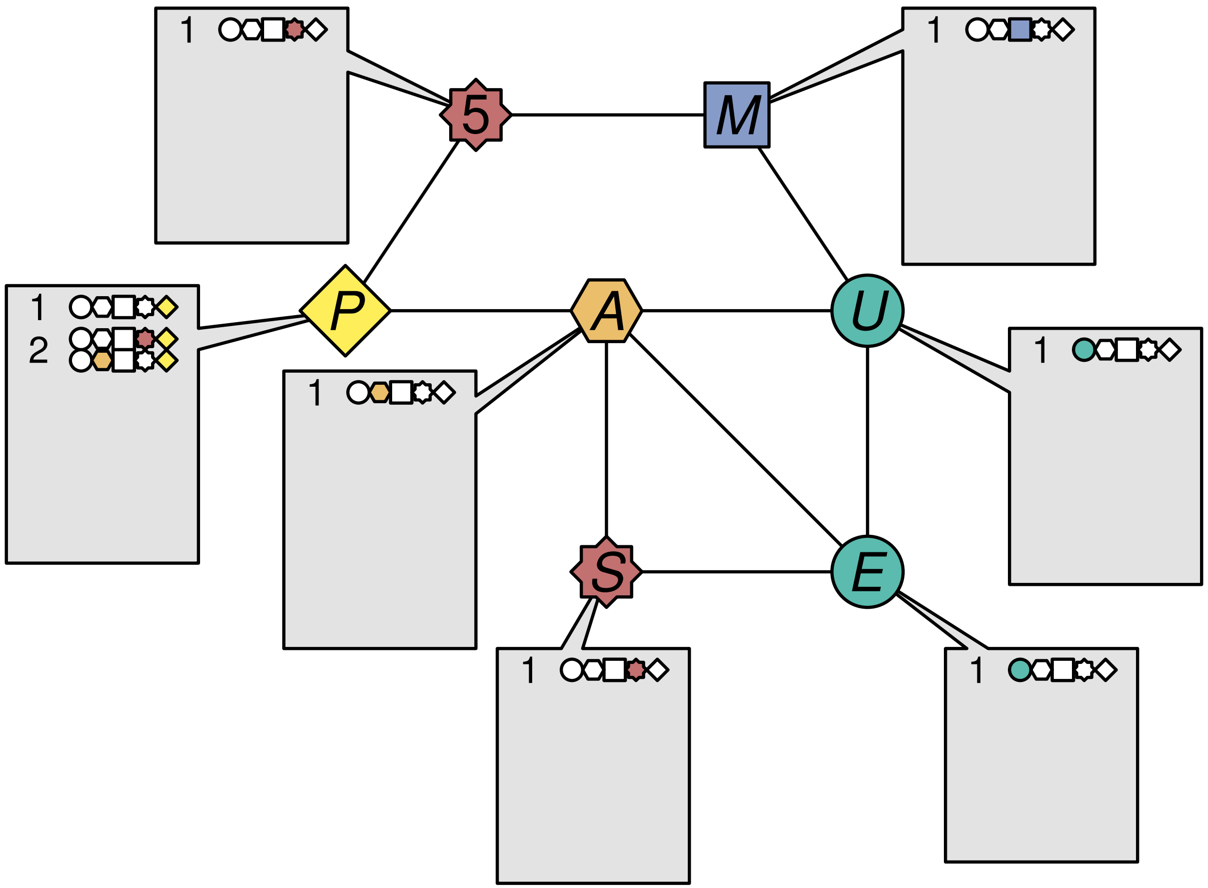
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



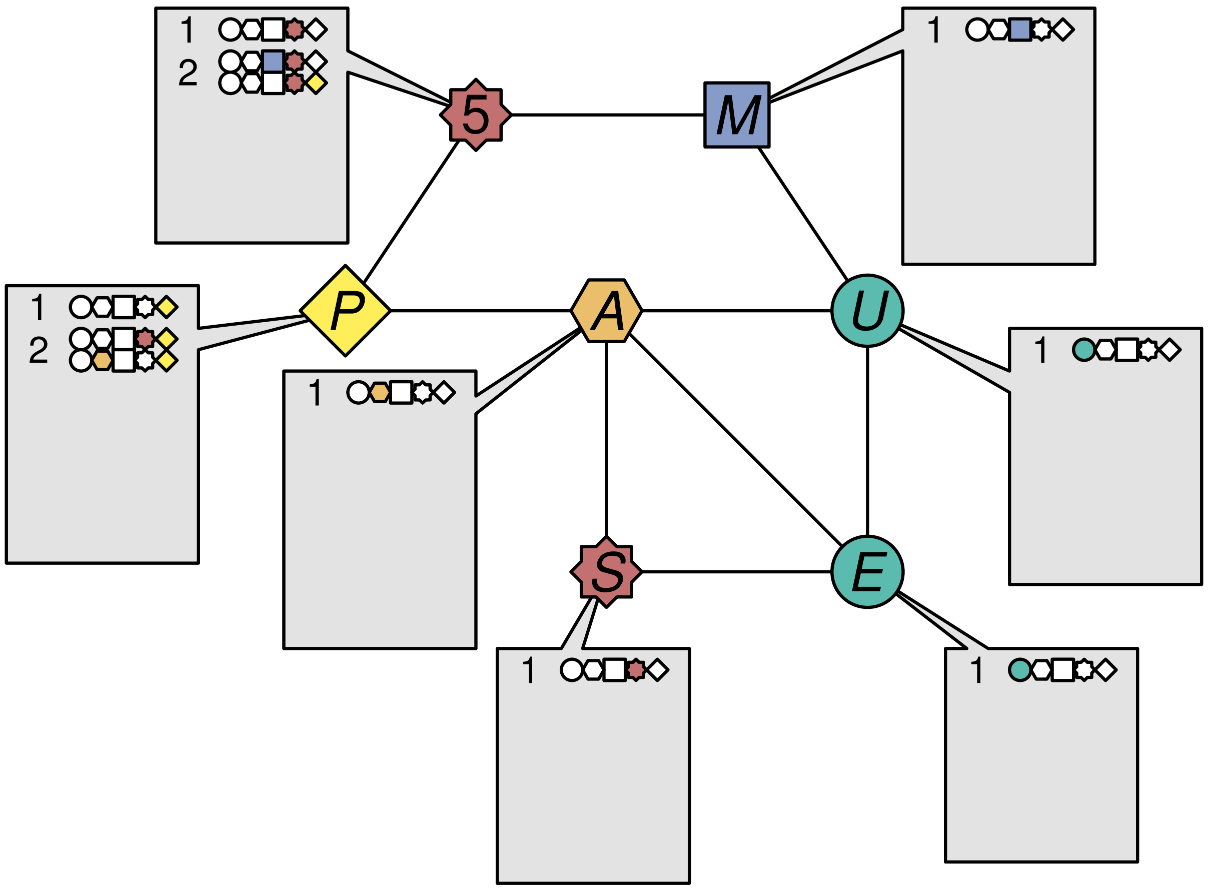
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



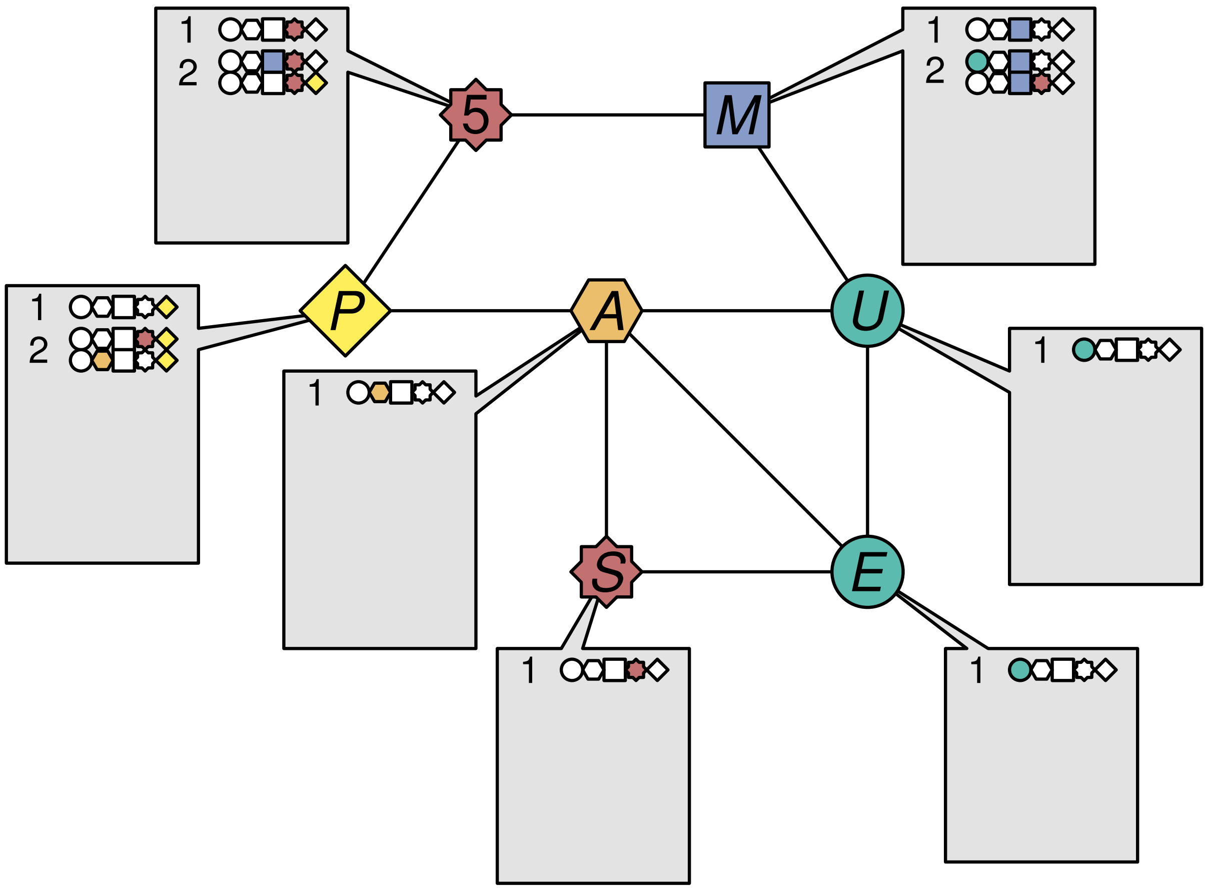
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \ldots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



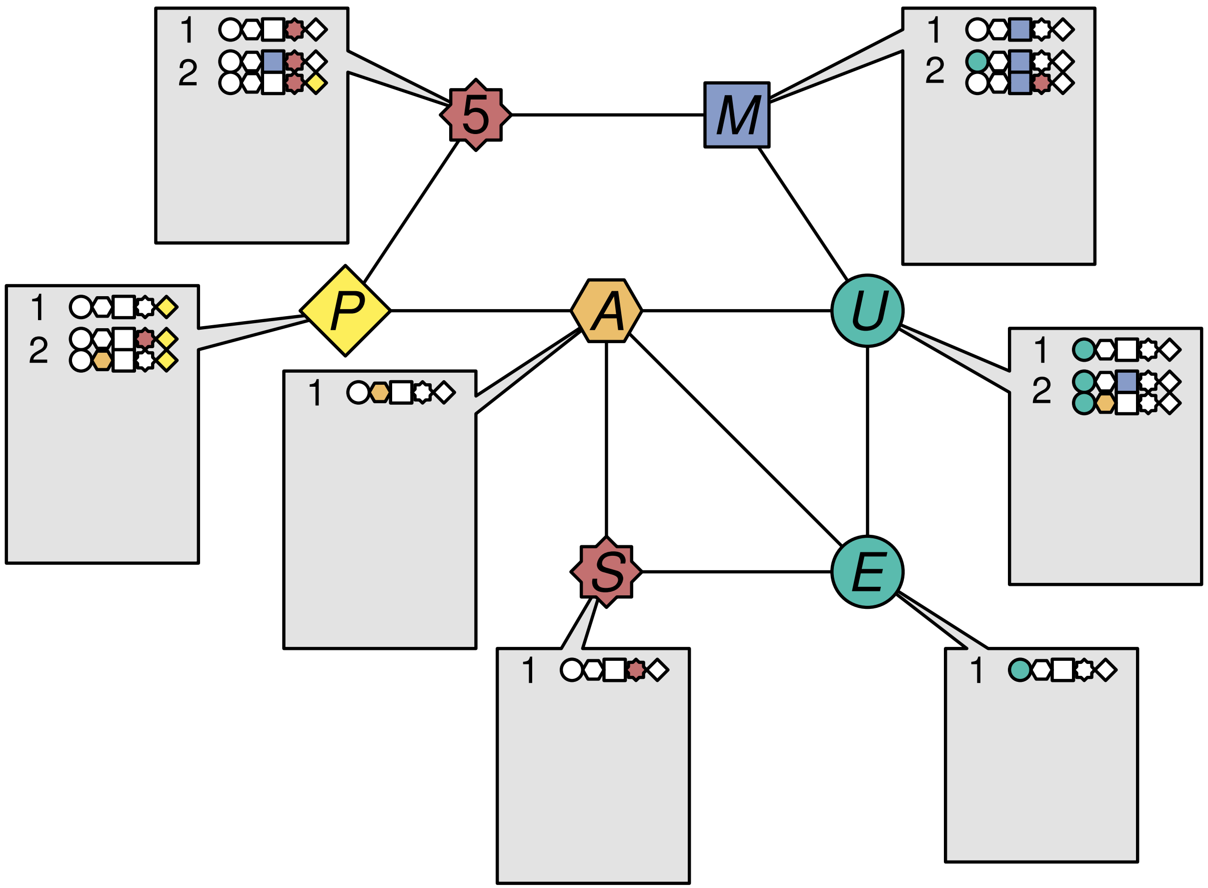
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



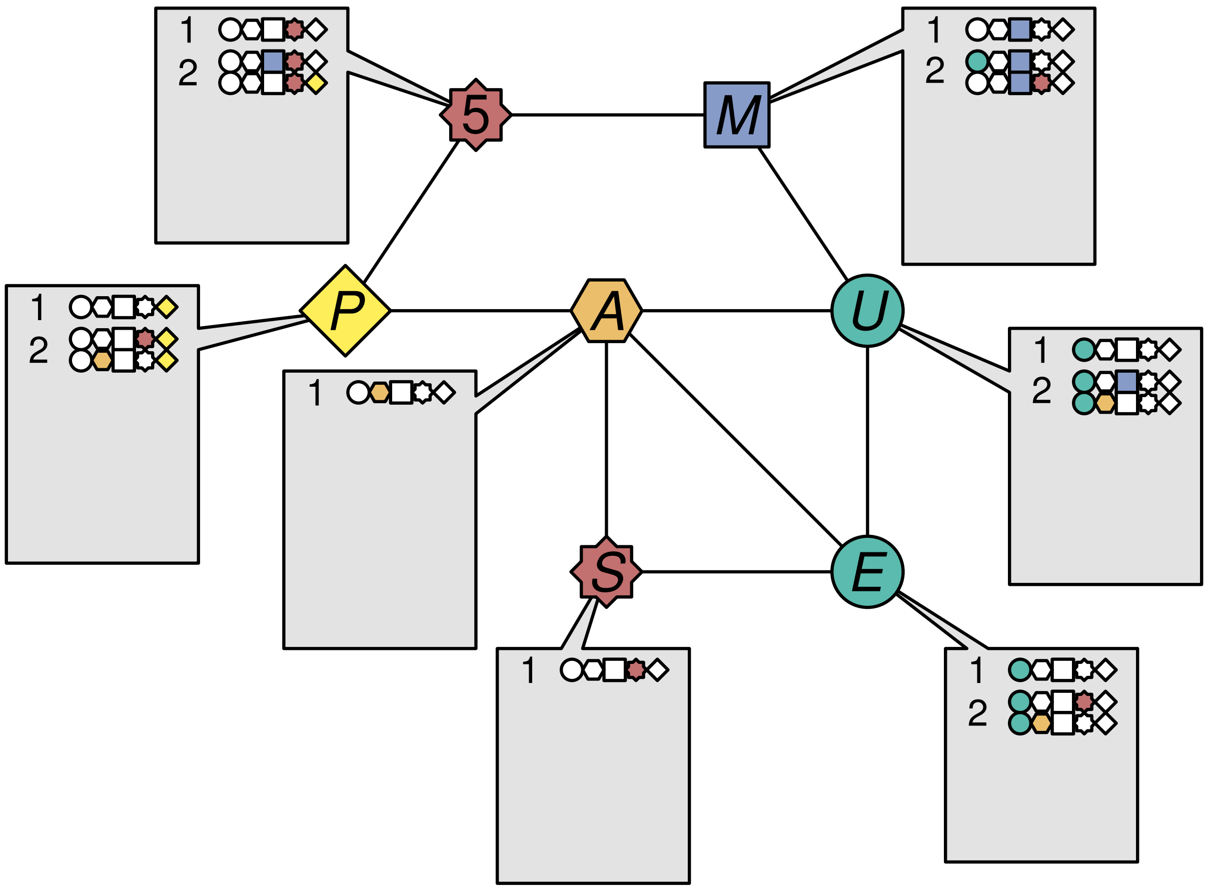
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



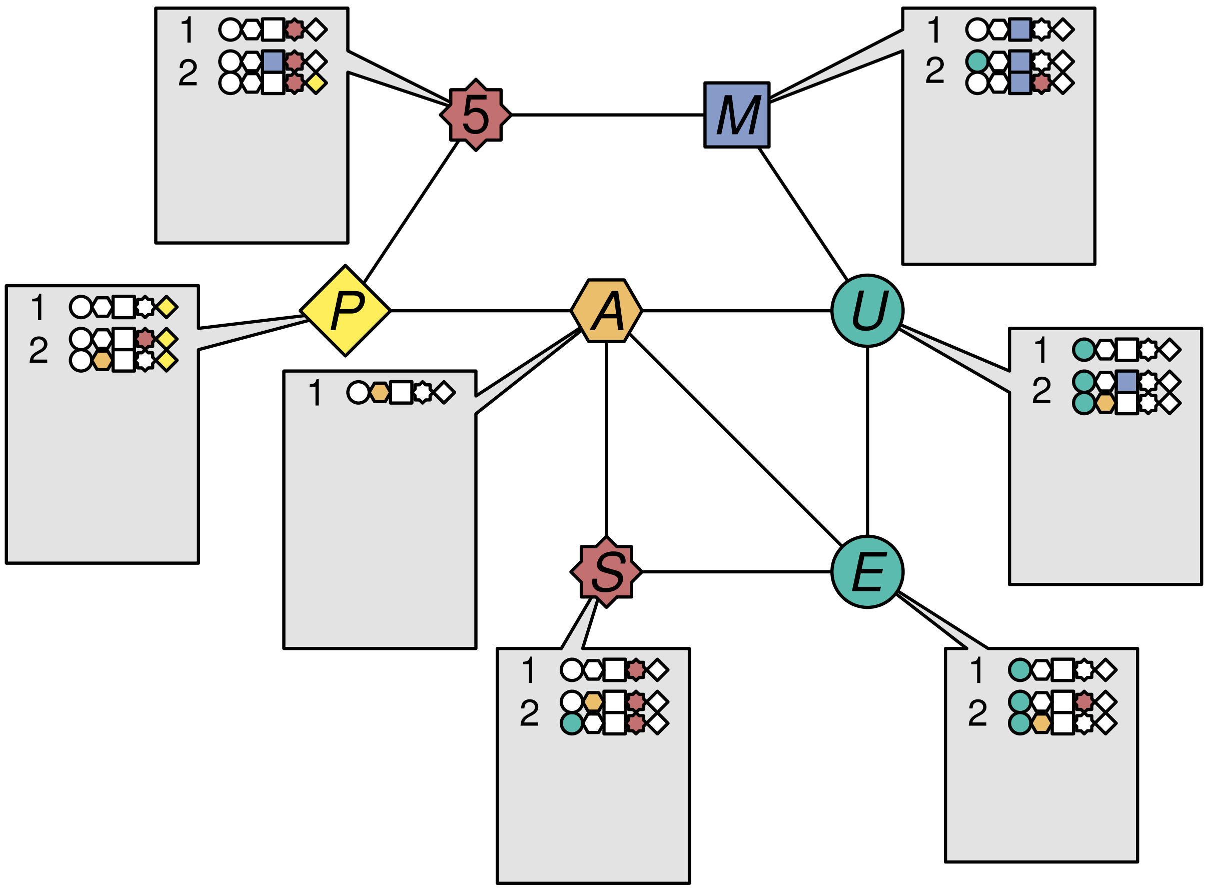
Dynamisches Programm

Problem: Colorful Longest Path
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



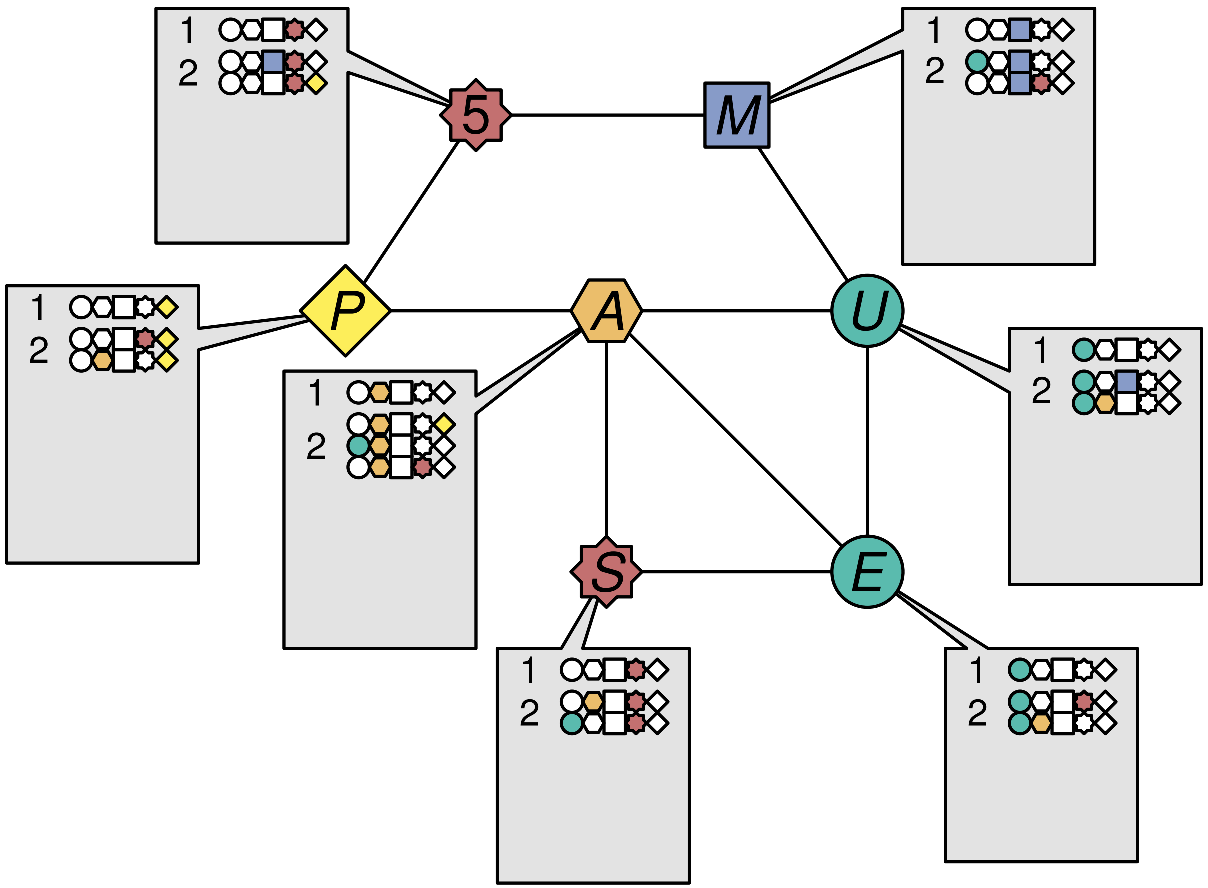
Dynamisches Programm

Problem: Colorful **LONGEST PATH**
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



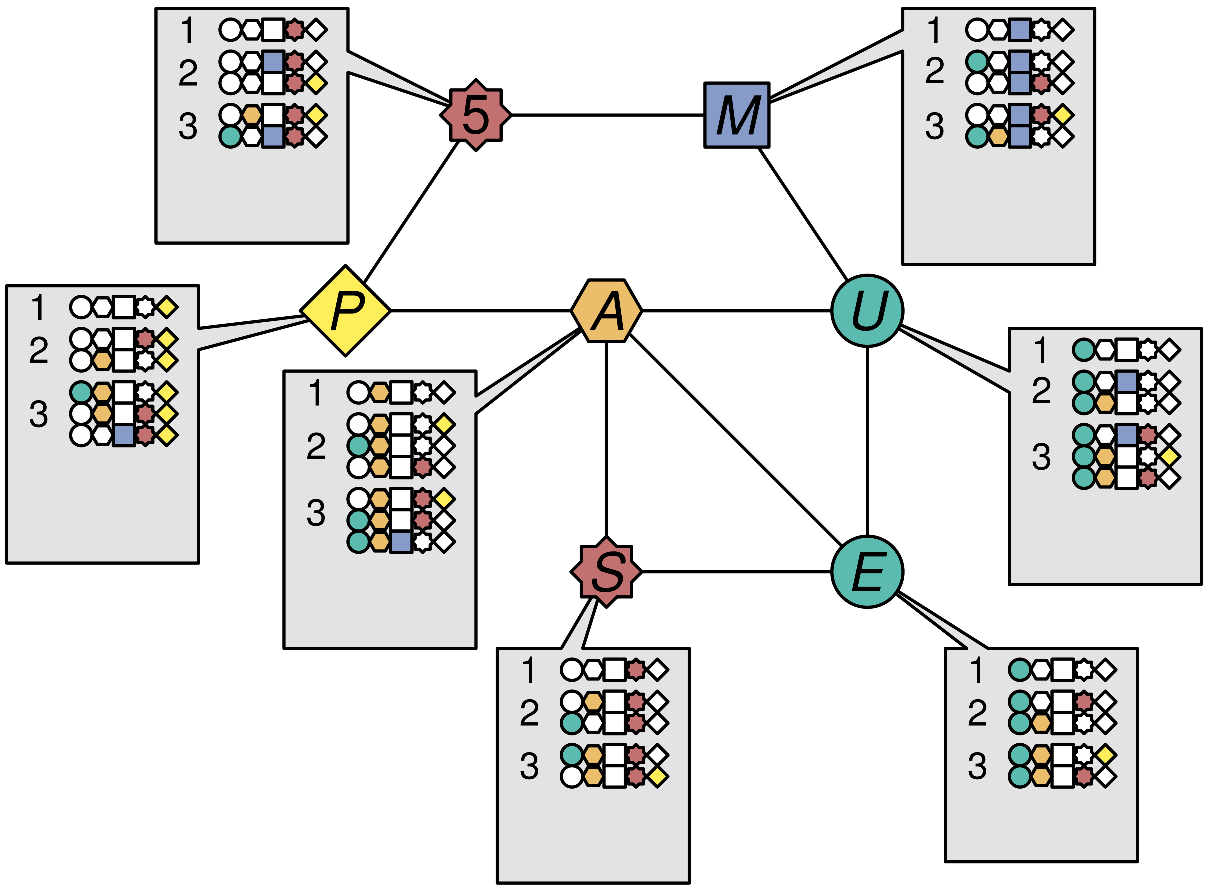
Dynamisches Programm

Problem: Colorful Longest Path
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



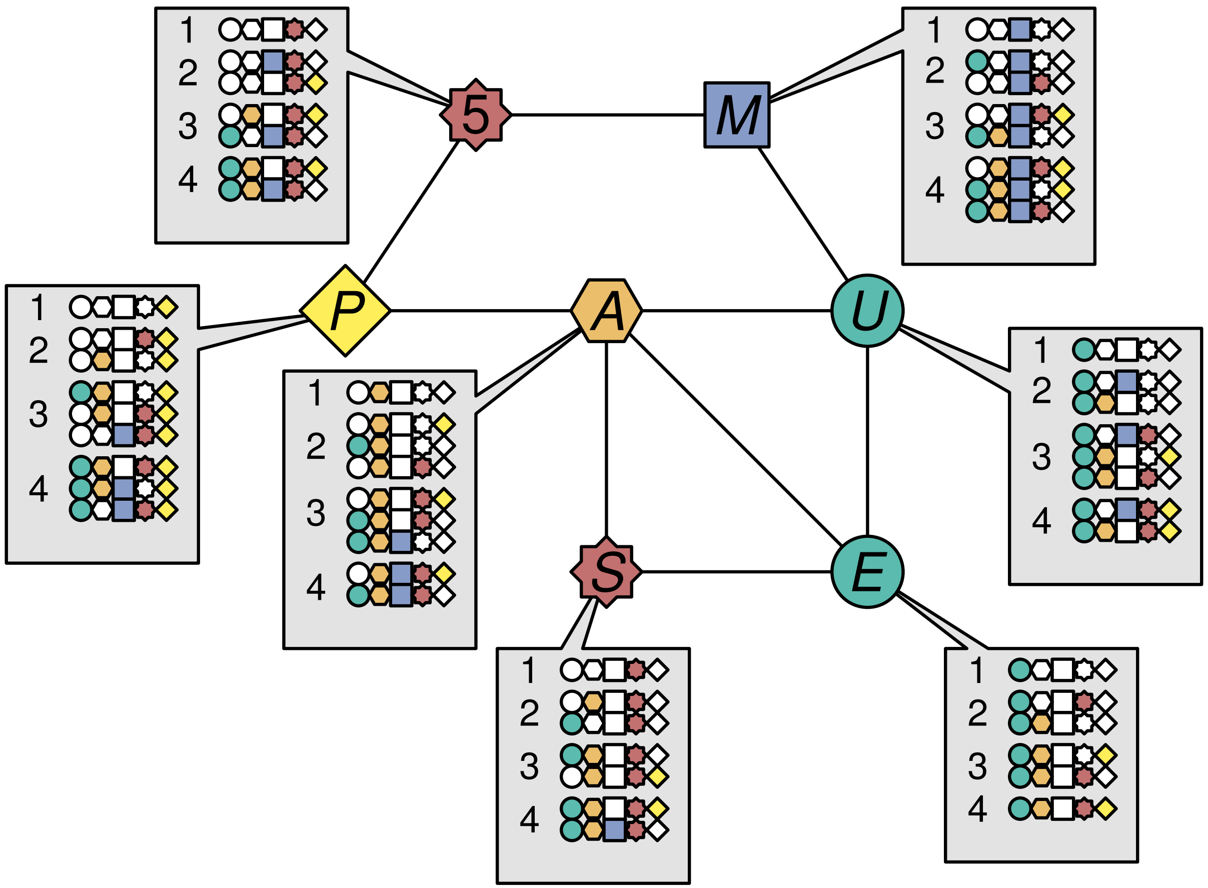
Dynamisches Programm

Problem: **Colorful** **LONGEST PATH**
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \ldots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



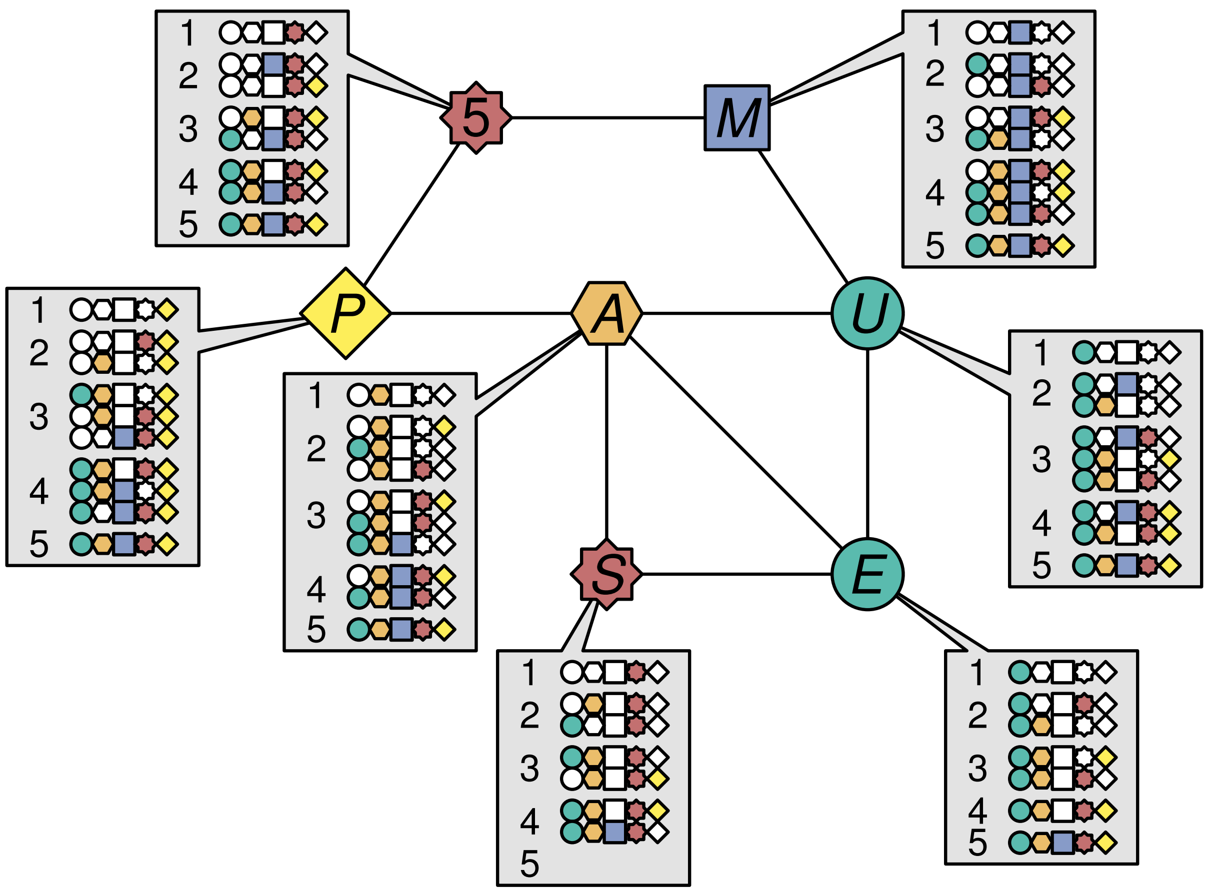
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



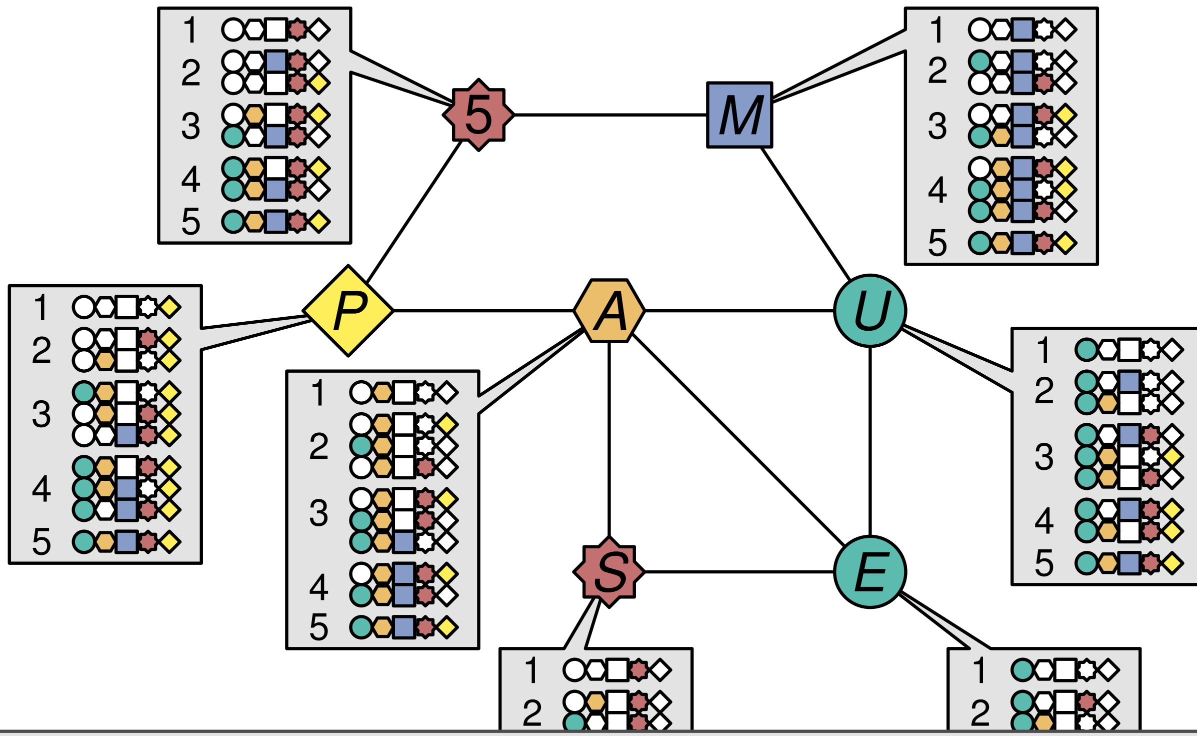
Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



Dynamisches Programm

Problem: Colorful LONGEST PATH
Gegeben ein Graph $G = (V, E)$, einen Parameter k, sowie eine Partition (Färbung) $V_1, \dots, V_k$ von V. Gibt es einen bunten Pfad der Länge k in G?
(ein Pfad ist bunt, wenn er aus jedem V_i nur einen Knoten enthält)



Theorem
COLORFUL LONGEST PATH kann in $O(2^k km)$ Zeit gelöst werden.

Randomisierte Färbung

Plan

- färbe Knoten und löse Colorful Longest Path statt Longest Path
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Randomisierte Färbung

Plan

- färbe Knoten und löse Colorful Longest Path statt Longest Path
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein bunter Pfad in der zufällig gefärbten Instanz mit Wahrscheinlichkeit mindestens $\frac{1}{e^k}$.

Randomisierte Färbung

Karlsruher Institut für Technologie

Plan

- färbe Knoten und löse Colorful Longest Path statt Longest Path
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein bunter Pfad in der zufällig gefärbten Instanz mit Wahrscheinlichkeit mindestens $\frac{1}{e^k}$.

Beweis

- sei $P = v_1, \dots, v_k$

Randomisierte Färbung

Plan

- färbe Knoten und löse COLORFUL LONGEST PATH statt LONGEST PATH
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein bunter Pfad in der zufällig gefärbten Instanz mit Wahrscheinlichkeit mindestens $\frac{1}{e^k}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Farbkombination dieser k Knoten ist gleich wahrscheinlich
- also: k^k Ereignisse, die jeweils mit Wkt $\frac{1}{k^k}$ auftreten

Randomisierte Färbung

Karlsruher Institut für Technologie

Plan

- färbe Knoten und löse Colorful Longest Path statt Longest Path
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Theorem
Sei P ein Pfad der Länge k in G. Dann ist P ein bunter Pfad in der zufällig gefärbten Instanz mit Wahrscheinlichkeit mindestens $\frac{1}{e^k}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Farbkombination dieser k Knoten ist gleich wahrscheinlich
- also: k^k Ereignisse, die jeweils mit Wkt $\frac{1}{k^k}$ auftreten
- $k!$ dieser Ereignisse sind gut

Randomisierte Färbung

Plan

- färbe Knoten und löse Colorful Longest Path statt Longest Path
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Theorem

Sei P ein Pfad der Länge k in G. Dann ist P ein bunter Pfad in der zufällig gefärbten Instanz mit Wahrscheinlichkeit mindestens $\frac{1}{e^k}$.

Beweis

- sei $P = v_1, \ldots, v_k$
- jede Farbkombination dieser k Knoten ist gleich wahrscheinlich
- also: k^k Ereignisse, die jeweils mit Wkt $\frac{1}{k^k}$ auftreten
- $k!$ dieser Ereignisse sind gut
- also: $\frac{k!}{k^k}$

Randomisierte Färbung

Plan

- färbe Knoten und löse Colorful Longest Path statt Longest Path
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Theorem

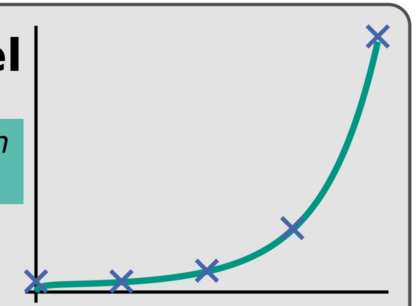
Sei P ein Pfad der Länge k in G. Dann ist P ein bunter Pfad in der zufällig gefärbten Instanz mit Wahrscheinlichkeit mindestens $\frac{1}{e^k}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Farbkombination dieser k Knoten ist gleich wahrscheinlich
- also: k^k Ereignisse, die jeweils mit Wkt $\frac{1}{k^k}$ auftreten
- $k!$ dieser Ereignisse sind gut
- also: $\frac{k!}{k^k}$

Stirlingformel

$$n! > \sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^n$$



Randomisierte Färbung

Plan

- färbe Knoten und löse Colorful Longest Path statt Longest Path
- Problem: Pfad der Länge k ist in gefärbter Instanz ggf. nicht bunt
- Lösung: wähle für jeden Knoten zufällig eine von k Farben

(unabhängig, gleichverteilt)

Theorem

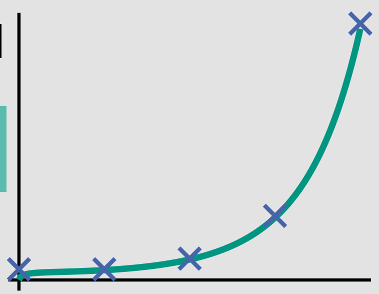
Sei P ein Pfad der Länge k in G. Dann ist P ein bunter Pfad in der zufällig gefärbten Instanz mit Wahrscheinlichkeit mindestens $\frac{1}{e^k}$.

Beweis

- sei $P = v_1, \dots, v_k$
- jede Farbkombination dieser k Knoten ist gleich wahrscheinlich
- also: k^k Ereignisse, die jeweils mit Wkt $\frac{1}{k^k}$ auftreten
- $k!$ dieser Ereignisse sind gut
- also: $\frac{k!}{k^k} > \frac{1}{e^k}$

Stirlingformel

$n! > \sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^n$



Wahrscheinlichkeit boosten

- $\Theta(e^k)$ Wiederholungen $\rightarrow$ konstante Wkt

Derandomisierung

Idee

- ausreichend viele zufällige Färbungen $\rightarrow$ eine ist gut (mit hoher Wkt)
- stattdessen: wähle clever mehrere Färbungen, sodass eine gut ist

Derandomisierung

Idee

- ausreichend viele zufällige Färbungen $\rightarrow$ eine ist gut (mit hoher Wkt)
- stattdessen: wähle clever mehrere Färbungen, sodass eine gut ist

Wünsch dir was

- Menge von Färbungen der n Elemente (Knoten) mit k Farben
- jede k-elementige Teilmenge ist in mindestens einer Färbung bunt

Derandomisierung

Idee

- ausreichend viele zufällige Färbungen $\rightarrow$ eine ist gut (mit hoher Wkt)
- stattdessen: wähle clever mehrere Färbungen, sodass eine gut ist

Wünsch dir was

- Menge von Färbungen der n Elemente (Knoten) mit k Farben
- jede k-elementige Teilmenge ist in mindestens einer Färbung bunt
- nennt man auch: (n, k) perfekte Familie von Hash-Funktionen

Derandomisierung

Idee

- ausreichend viele zufällige Färbungen $\rightarrow$ eine ist gut (mit hoher Wkt)
- stattdessen: wähle clever mehrere Färbungen, sodass eine gut ist

Wünsch dir was

- Menge von Färbungen der n Elemente (Knoten) mit k Farben
- jede k-elementige Teilmenge ist in mindestens einer Färbung bunt
- nennt man auch: (n, k) perfekte Familie von Hash-Funktionen

Theorem (ohne Beweis)
Eine (n, k) perfekte Familie von Hash-Funktionen der Größe $O(6{,}4^k \log^2 n)$ kann in $O(6{,}4^k n \log^2 n)$ Zeit konturiert werden.

Derandomisierung

Idee

- ausreichend viele zufällige Färbungen $\rightarrow$ eine ist gut (mit hoher Wkt)
- stattdessen: wähle clever mehrere Färbungen, sodass eine gut ist

Wünsch dir was

- Menge von Färbungen der n Elemente (Knoten) mit k Farben
- jede k-elementige Teilmenge ist in mindestens einer Färbung bunt
- nennt man auch: (n, k) perfekte Familie von Hash-Funktionen

Theorem (ohne Beweis)
Eine (n, k) perfekte Familie von Hash-Funktionen der Größe $O(6{,}4^k \log^2 n)$ kann in $O(6{,}4^k n \log^2 n)$ Zeit konturiert werden.

Theorem (vorhin gesehen)
COLORFUL LONGEST PATH kann in $O(2^k km)$ Zeit gelöst werden.

Derandomisierung

Idee

- ausreichend viele zufällige Färbungen $\rightarrow$ eine ist gut (mit hoher Wkt)
- stattdessen: wähle clever mehrere Färbungen, sodass eine gut ist

Wünsch dir was

- Menge von Färbungen der n Elemente (Knoten) mit k Farben
- jede k-elementige Teilmenge ist in mindestens einer Färbung bunt
- nennt man auch: (n, k) perfekte Familie von Hash-Funktionen

Theorem (ohne Beweis)
Eine (n, k) perfekte Familie von Hash-Funktionen der Größe $O(6{,}4^k \log^2 n)$ kann in $O(6{,}4^k n \log^2 n)$ Zeit konturiert werden.

Theorem (vorhin gesehen)
Colorful Longest Path kann in $O(2^k km)$ Zeit gelöst werden.

Theorem (folgt direkt)
Longest Path kann in $O(12{,}8^k km \log^2 n)$ Zeit gelöst werden.

Zusammenfassung: Color Coding

Grundsätzliches Vorgehen

- definiere gefärbte Problemvariante (mit Präfix COLORFUL oder MULTICOLORED)
- entwickle FPT-Algorithmus für gefärbtes Problem
- zeige:
  - nein-Instanz wird zu gefärbter nein-Instanz
  - ja-Instanz wird zu gefärbter ja-Instanz für mindestens eine Färbung in der perfekten Familie von Hash-Funktionen

Zusammenfassung: Color Coding

Grundsätzliches Vorgehen

- definiere gefärbte Problemvariante (mit Präfix COLORFUL oder MULTICOLORED)
- entwickle FPT-Algorithmus für gefärbtes Problem
- zeige:
  - nein-Instanz wird zu gefärbter nein-Instanz
  - ja-Instanz wird zu gefärbter ja-Instanz für mindestens eine Färbung in der perfekten Familie von Hash-Funktionen

Tipp fürs Color Coding

- betrachte Lösung einer Probleminstanz
- suche Zeugen der Größe $f(k)$ dafür, dass es eine Lösung ist
- fordere, dass dieser Zeuge bunt ist

Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Beispiel

$U = \{a, b, c, d\}$

b a

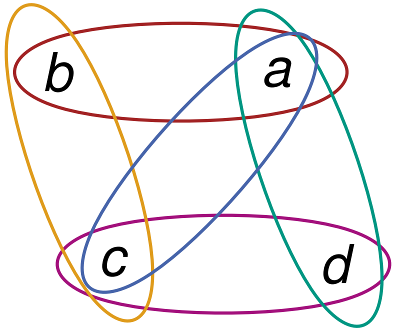
c d

Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Beispiel

$U = \{a, b, c, d\}$ $\mathcal{S} = \{\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{c, d\}\}$



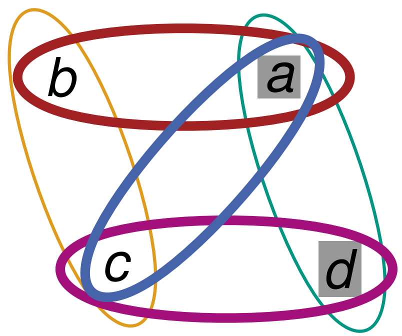
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Beispiel

$U = \{a, b, c, d\}$ $\mathcal{S} = \{\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{c, d\}\}$

$X = \{a, d\}$



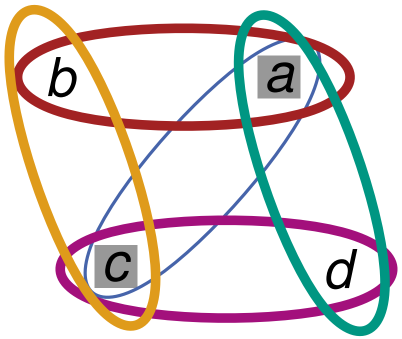
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Beispiel

$U = \{a, b, c, d\}$ $\mathcal{S} = \{\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{c, d\}\}$

$X = \{a, c\}$

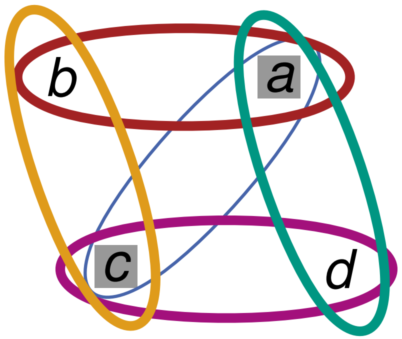


Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Beispiel

$U = \{a, b, c, d\}$ $\mathcal{S} = \{\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{c, d\}\}$
$X = \{a, c\}$



Anmerkung

- Set Splitting ist im Prinzip Max Cut auf Hypergraphen

Set Splitting

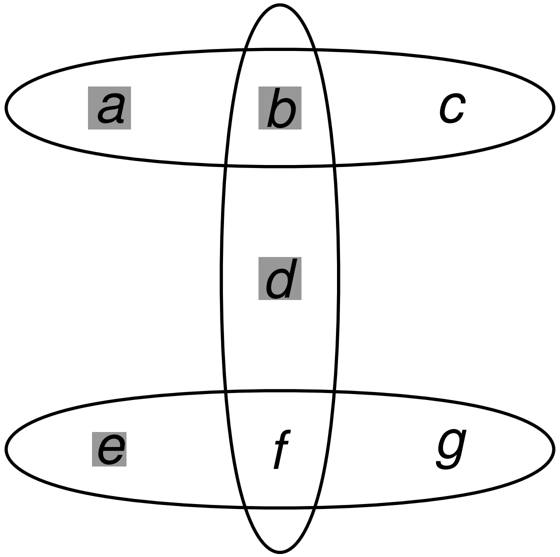
Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X$

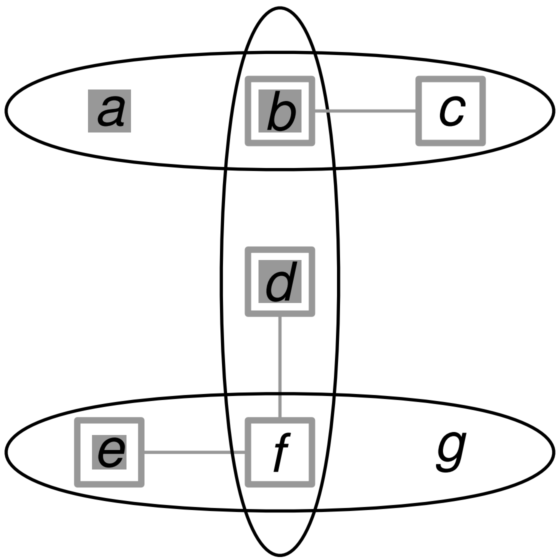


$X = \{a, b, d, e\}$ $\quad U \setminus X = \{c, f, g\}$

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X \rightarrow$ $\boxed{\text{Zeuge der Größe} \leq 2k}$



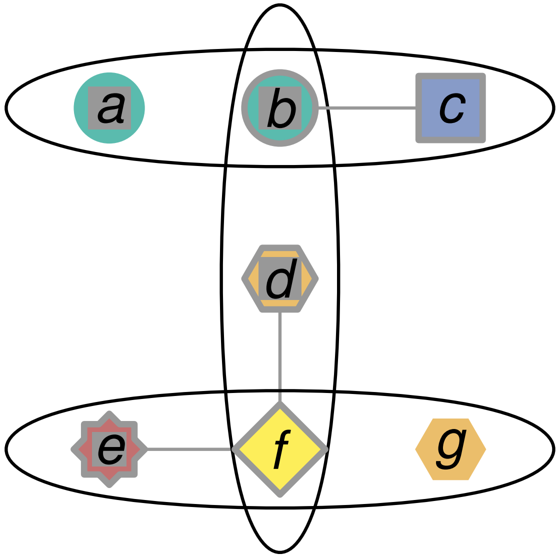
$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X \rightarrow$ $\boxed{\text{Zeuge der Größe} \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)



$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

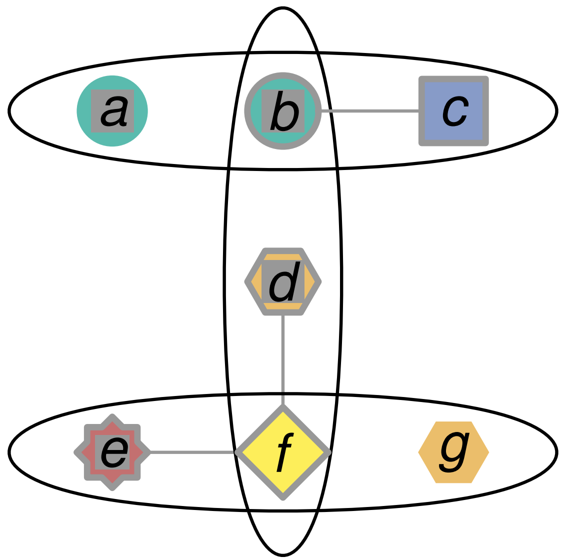
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X$ → $\boxed{\text{Zeuge der Größe} \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)

Wie können wir Colorful Set Splitting lösen?



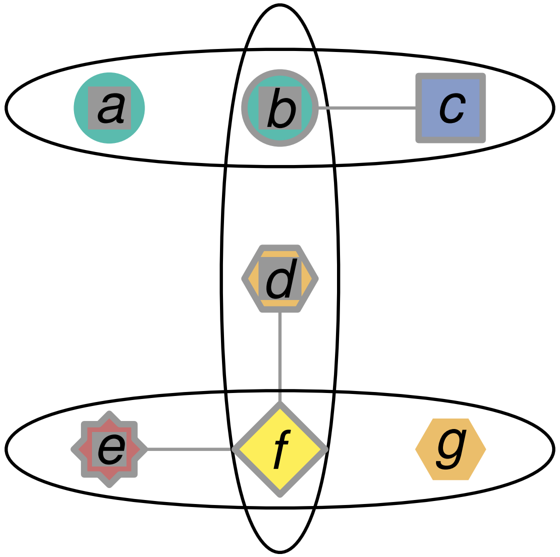
$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X \rightarrow$ $\boxed{\text{Zeuge der Größe } \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)
- für jede Teilmenge der Farben: wähle alle Elemente mit diesen Farben



$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

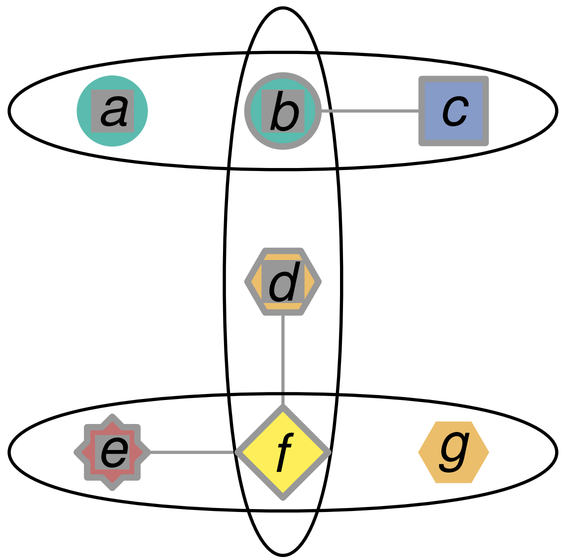
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X \rightarrow$ $\boxed{\text{Zeuge der Größe} \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)
- für jede Teilmenge der Farben: wähle alle Elemente mit diesen Farben
- $\Rightarrow$ Zeuge wird irgendwann korrekt getrennt

(hier: $X = \{$, , $\}$, $U \setminus X = \{$, $\}$)



$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

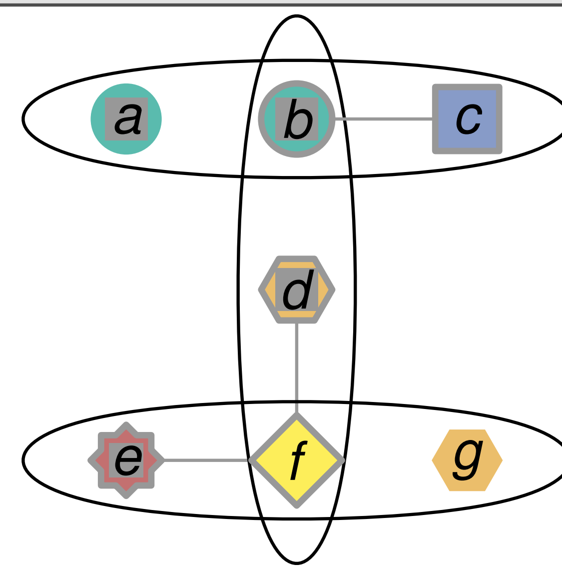
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X$ → $\boxed{\text{Zeuge der Größe} \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)
- für jede Teilmenge der Farben: wähle alle Elemente mit diesen Farben
- $\Rightarrow$ Zeuge wird irgendwann korrekt getrennt

(hier: $X = \{ , , \}$, $U \setminus X = \{ , \}$)



$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

Warum so kompliziert?

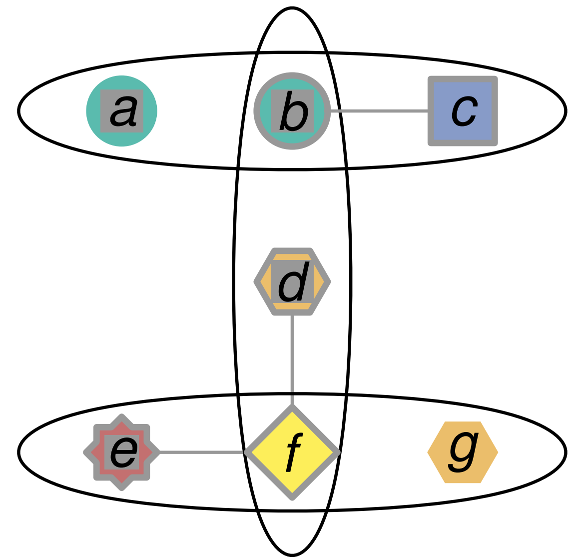
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X \rightarrow$ $\boxed{\text{Zeuge der Größe} \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)
- für jede Teilmenge der Farben: wähle alle Elemente mit diesen Farben
- $\Rightarrow$ Zeuge wird irgendwann korrekt getrennt

(hier: $X = \{$, , $\}$, $U \setminus X = \{$, $\}$)



$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

Warum so kompliziert?

- eigentlich brauchen wir nur zwei Farben: für X und für $U \setminus X$

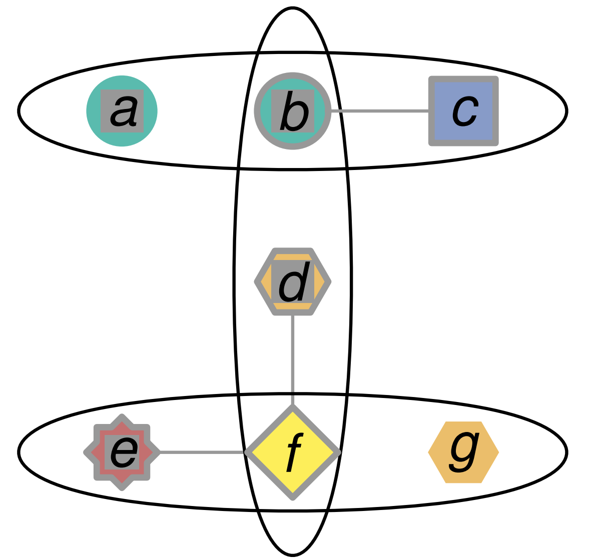
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X \rightarrow$ $\boxed{\text{Zeuge der Größe} \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)
- für jede Teilmenge der Farben: wähle alle Elemente mit diesen Farben
- $\Rightarrow$ Zeuge wird irgendwann korrekt getrennt

(hier: $X = \{$, , $\}$, $U \setminus X = \{$, $\}$)



$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

Warum so kompliziert?

- eigentlich brauchen wir nur zwei Farben: für X und für $U \setminus X$
- Wunsch: in einer der 2-Färbungen wird der $2k$ große Zeuge gut geteilt

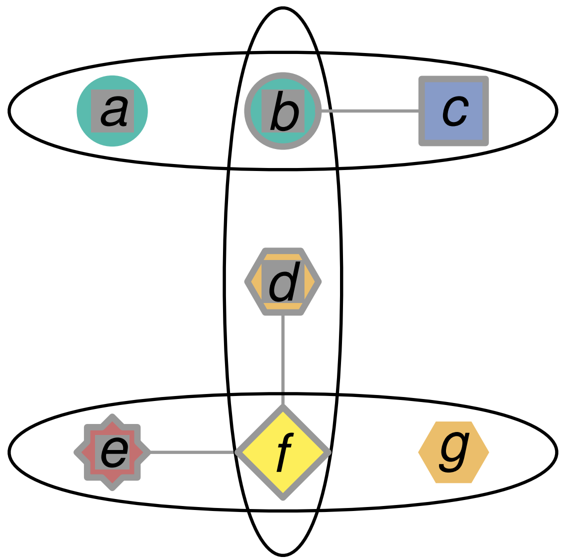
Set Splitting

Problem: Set Splitting
Gegeben eine Grundmenge U, eine Menge $\mathcal{S} \subseteq 2^U$ von Teilmengen und einen Parameter k. Gibt es eine Menge $X \subset U$, die mindestens k Mengen in $\mathcal{S}$ zerschneidet? (X zerschneidet $S \in \mathcal{S}$, wenn $S \cap X \neq \emptyset$ und $S \not\subseteq X$)

Idee für Color Coding

- jede zerschnittene Menge enthält Element aus X und aus $U \setminus X$ → $\boxed{\text{Zeuge der Größe} \leq 2k}$
- Zeuge ist bunt für eine der Färbungen mit $2k$ Farben (($n, 2k$) perfekte Familie von Hash-Funktionen)
- für jede Teilmenge der Farben: wähle alle Elemente mit diesen Farben
- $\Rightarrow$ Zeuge wird irgendwann korrekt getrennt

(hier: $X = \{\ ,\ ,\ \}$, $U \setminus X = \{\ ,\ \}$)



$X = \{a, b, d, e\}$ $U \setminus X = \{c, f, g\}$

Warum so kompliziert?

- eigentlich brauchen wir nur zwei Farben: für X und für $U \setminus X$
- Wunsch: in einer der 2-Färbungen wird der $2k$ große Zeuge gut geteilt
- dafür gibt es universelle Mengen

Universelle Mengen

Universelle Mengen

Wünsch dir was

- Menge von Färbungen der n Elemente mit 2 Farben
- für jede k-elementige Teilmenge kommt jede der 2^k Färbungen mindestens einmal vor

Universelle Mengen

Wünsch dir was

- Menge von Färbungen der n Elemente mit 2 Farben
- für jede k-elementige Teilmenge kommt jede der 2^k Färbungen mindestens einmal vor
- nennt man auch: (n, k) universelle Menge

Theorem (ohne Beweis)
Eine (n, k) universelle Menge kann in $O(n2^{k+o(k)})$ konstruiert werden.

Universelle Mengen

Wünsch dir was

- Menge von Färbungen der n Elemente mit 2 Farben
- für jede k-elementige Teilmenge kommt jede der 2^k Färbungen mindestens einmal vor
- nennt man auch: (n, k) universelle Menge

Theorem (ohne Beweis)
Eine (n, k) universelle Menge kann in $O(n2^{k+o(k)})$ konstruiert werden.

Was heißt das für SET SPLITTING?

- nutze Färbung mit $(n, 2k)$ universeller Menge

Universelle Mengen

Wünsch dir was

- Menge von Färbungen der n Elemente mit 2 Farben
- für jede k-elementige Teilmenge kommt jede der 2^k Färbungen mindestens einmal vor
- nennt man auch: (n, k) universelle Menge

Theorem (ohne Beweis)
Eine (n, k) universelle Menge kann in $O(n2^{k+o(k)})$ konstruiert werden.

Was heißt das für Set Splitting?

- nutze Färbung mit $(n, 2k)$ universeller Menge
- Lösung $(X, Y = U \setminus X)$ für Set Splitting $\to$ Zeuge der Größe $2k$

Universelle Mengen

Karlsruher Institut für Technologie

Wünsch dir was

- Menge von Färbungen der n Elemente mit 2 Farben
- für jede k-elementige Teilmenge kommt jede der 2^k Färbungen mindestens einmal vor
- nennt man auch: (n, k) universelle Menge

Theorem (ohne Beweis)
Eine (n, k) universelle Menge kann in $O(n2^{k+o(k)})$ konstruiert werden.

Was heißt das für Set Splitting?

- nutze Färbung mit $(n, 2k)$ universeller Menge
- Lösung $(X, Y = U \setminus X)$ für Set Splitting $\rightarrow$ Zeuge der Größe $2k$
- mindestens eine Färbung spaltet den Zeugen entsprechend (X, Y) auf

Universelle Mengen

Wünsch dir was

- Menge von Färbungen der n Elemente mit 2 Farben
- für jede k-elementige Teilmenge kommt jede der 2^k Färbungen mindestens einmal vor
- nennt man auch: (n, k) universelle Menge

Theorem (ohne Beweis)
Eine (n, k) universelle Menge kann in $O(n2^{k+o(k)})$ konstruiert werden.

Was heißt das für Set Splitting?

- nutze Färbung mit $(n, 2k)$ universeller Menge
- Lösung $(X, Y = U \setminus X)$ für Set Splitting $\rightarrow$ Zeuge der Größe $2k$
- mindestens eine Färbung spaltet den Zeugen entsprechend (X, Y) auf

  $\Rightarrow$ korrekter Algo: teste für jede Färbung, ob sie k Mengen zerteilt

Universelle Mengen

Wünsch dir was

- Menge von Färbungen der n Elemente mit 2 Farben
- für jede k-elementige Teilmenge kommt jede der 2^k Färbungen mindestens einmal vor
- nennt man auch: (n, k) universelle Menge

Theorem (ohne Beweis)
Eine (n, k) universelle Menge kann in $O(n2^{k+o(k)})$ konstruiert werden.

Was heißt das für Set Splitting?

- nutze Färbung mit $(n, 2k)$ universeller Menge
- Lösung $(X, Y = U \setminus X)$ für Set Splitting $\to$ Zeuge der Größe $2k$
- mindestens eine Färbung spaltet den Zeugen entsprechend (X, Y) auf
  $\Rightarrow$ korrekter Algo: teste für jede Färbung, ob sie k Mengen zerteilt

Theorem
Set Splitting kann in $n^{O(1)}4^{k+o(k)}$ gelöst werden.

Zusammenfassung

Randomisierung

- kann mit FPT kombiniert werden

Zusammenfassung

Randomisierung

- kann mit FPT kombiniert werden
- Randomisierung ist oft nur Zwischenschritt $\rightarrow$ Derandomisierung

Zusammenfassung

Randomisierung

- kann mit FPT kombiniert werden
- Randomisierung ist oft nur Zwischenschritt $\rightarrow$ Derandomisierung
- Raten zusätzlicher Struktur eröffnet neue Lösungsansätze

Zusammenfassung

Randomisierung

- kann mit FPT kombiniert werden
- Randomisierung ist oft nur Zwischenschritt $\rightarrow$ Derandomisierung
- Raten zusätzlicher Struktur eröffnet neue Lösungsansätze

Color Coding

- Färbung bzw. Partitionierung gibt nützliche zusätzliche Struktur

Zusammenfassung

Randomisierung

- kann mit FPT kombiniert werden
- Randomisierung ist oft nur Zwischenschritt $\rightarrow$ Derandomisierung
- Raten zusätzlicher Struktur eröffnet neue Lösungsansätze

Color Coding

- Färbung bzw. Partitionierung gibt nützliche zusätzliche Struktur
- zentrale Tools: perfekte Familien von Hash-Funktionen und universelle Mengen

Zusammenfassung

Randomisierung

- kann mit FPT kombiniert werden
- Randomisierung ist oft nur Zwischenschritt $\rightarrow$ Derandomisierung
- Raten zusätzlicher Struktur eröffnet neue Lösungsansätze

Color Coding

- Färbung bzw. Partitionierung gibt nützliche zusätzliche Struktur
- zentrale Tools: perfekte Familien von Hash-Funktionen und universelle Mengen
- formal ist Color Coding eine Art Turing-Reduktion