STUNDENPLANPROBLEM

Graphen und Netzwerke

GLIEDERUNG

1. Problemvorstellung
2. Idee – Warum ein Färbungsproblem?
3. Unsere Umsetzung
    1. Erste Schritte
    2. Algorithmen
    3. Live-Demo
4. Laufzeitdiskussion

PROBLEMSTELLUNG

- Problembeschreibung:
  - Verantwortlichkeit für die Umsetzung eines Stundenplans
  - Verschiedene **MODULE** müssen in geeignete **BLÖCKE** unterteilt werden
  - Module werden genau einem **DOZIERENDEN** und einem **SEMESTER** zugeordnet
  - Manche Module müssen in einem bestimmten **RAUM** stattfinden
- Aufgabe:
  - **GRAPHALGORITHMUS** entwickeln, der die **MINIMALE ANZAHL AN BLÖCKEN** bestimmt

IDEE - WARUM EIN FÄRBUNGSPROBLEM?

- Wie kann daraus ein Graph entstehen?
  - Knoten: Module
  - Kanten: Zwischen den Modulen, die nicht im gleichen Block liegen dürfen

IDEE - WARUM EIN FÄRBUNGSPROBLEM?

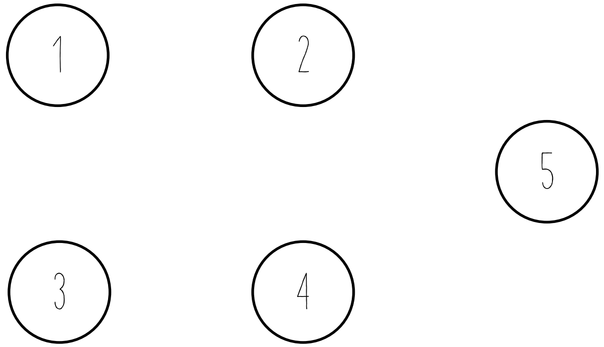
- Wie kann daraus ein Graph entstehen?
  - Knoten: Module
  - Kanten: Zwischen den Modulen, die nicht im gleichen Block liegen dürfen
- Beispiel:

Module	Dozent	Semester	Raum
1	1	1	-
2	1	2	-
3	2	1	1
4	2	1	1
5	2	2	-

IDEE - WARUM EIN FÄRBUNGSPROBLEM?

- Wie kann daraus ein Graph entstehen?
  - Knoten: Module
  - Kanten: Zwischen den Modulen, die nicht im gleichen Block liegen dürfen
- Beispiel:

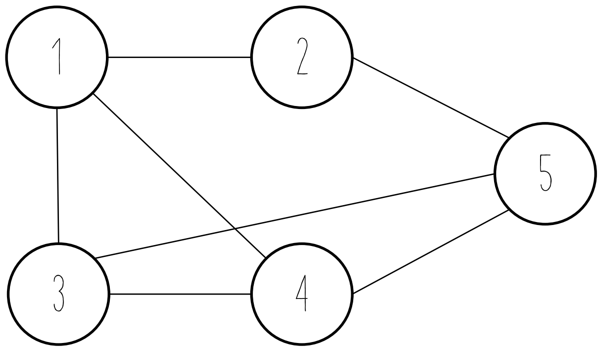
Module	Dozent	Semester	Raum
1	1	1	-
2	1	2	-
3	2	1	1
4	2	1	1
5	2	2	-



IDEE - WARUM EIN FÄRBUNGSPROBLEM?

- Wie kann daraus ein Graph entstehen?
  - Knoten: Module
  - Kanten: Zwischen den Modulen, die nicht im gleichen Block liegen dürfen
- Beispiel:

Module	Dozent	Semester	Raum
1	1	1	-
2	1	2	-
3	2	1	1
4	2	1	1
5	2	2	-

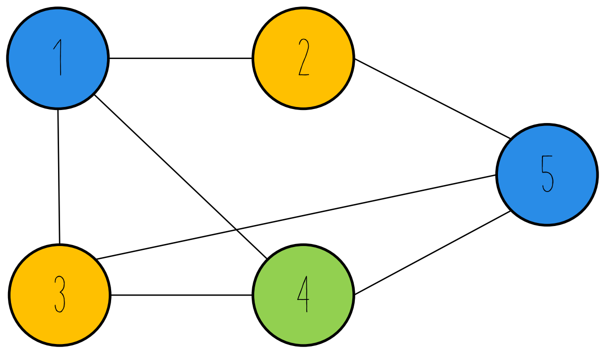


IDEE – WARUM EIN FÄRBUNGSPROBLEM?

- Beispiel:

Module	Dozent	Semester	Raum
1	1	1	–
2	1	2	–
3	2	1	1
4	2	1	1
5	2	2	–

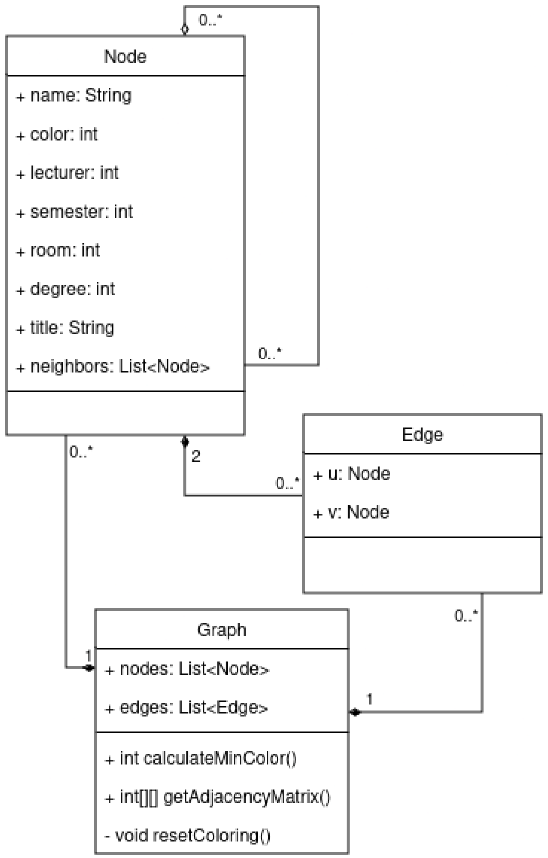
FÄRBUNG!

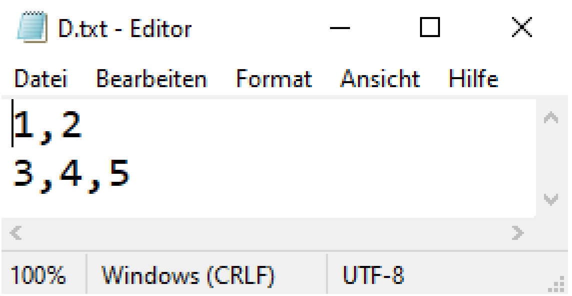


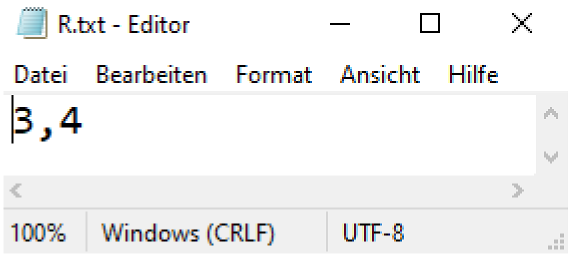
- Jede Farbe stellt einen Block dar

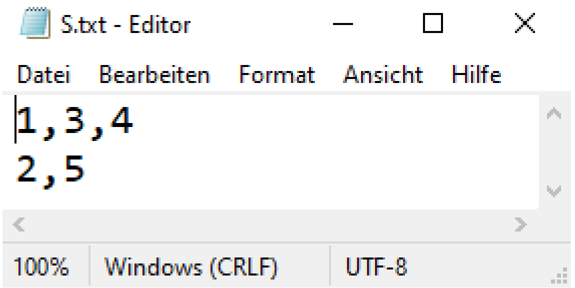
UNSERE UMSETZUNG - ERSTE SCHRITTE

- Überschaubare UML-Diagramme erstellt
- Testinstanzen angeschaut
  - 3 Textdateien für Dozenten, Semester und Raum
- Java-Klassen für Knoten, Kanten und Graphen erstellt
- Klasse CalculateTimetable erstellt
  - Textdateien einlesen
  - Graph mit entsprechenden Knoten und Kanten erstellen









UNSERE UMSETZUNG - ALGORITHMEN

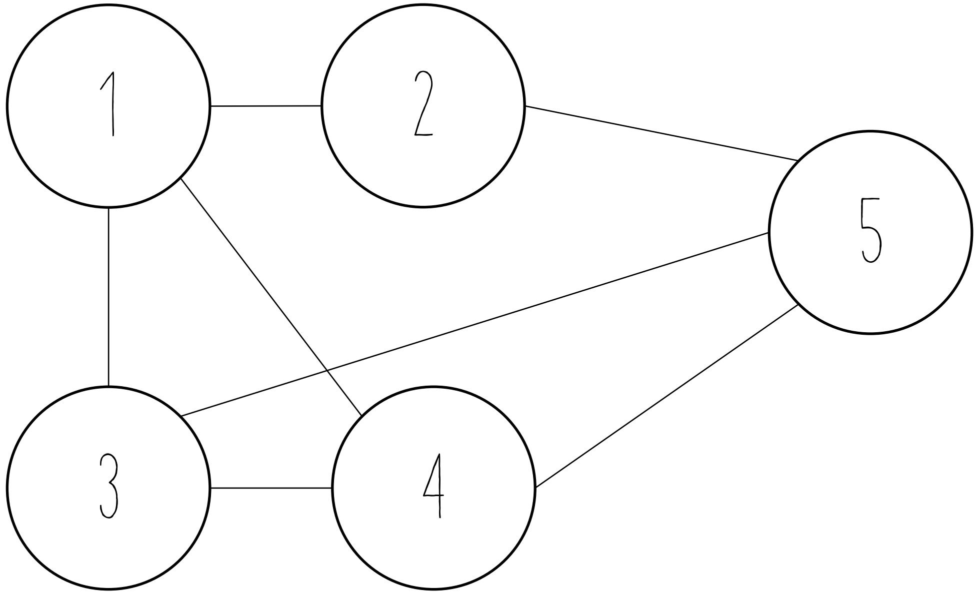
- Alle Algorithmen aus Unterricht umgesetzt:
  - Sequenzieller Algorithmus
  - Johnson Algorithmus
  - Backtracking
- Können in Klasse Graph aufgerufen werden

RECAP: SEQUENZIELLER ALGORITHMUS

Module	Farben
1	-
2	-
3	-
4	-
5	-

- Idee: Bestimme die Farbe der Knoten über die Färbung der Nachbarn

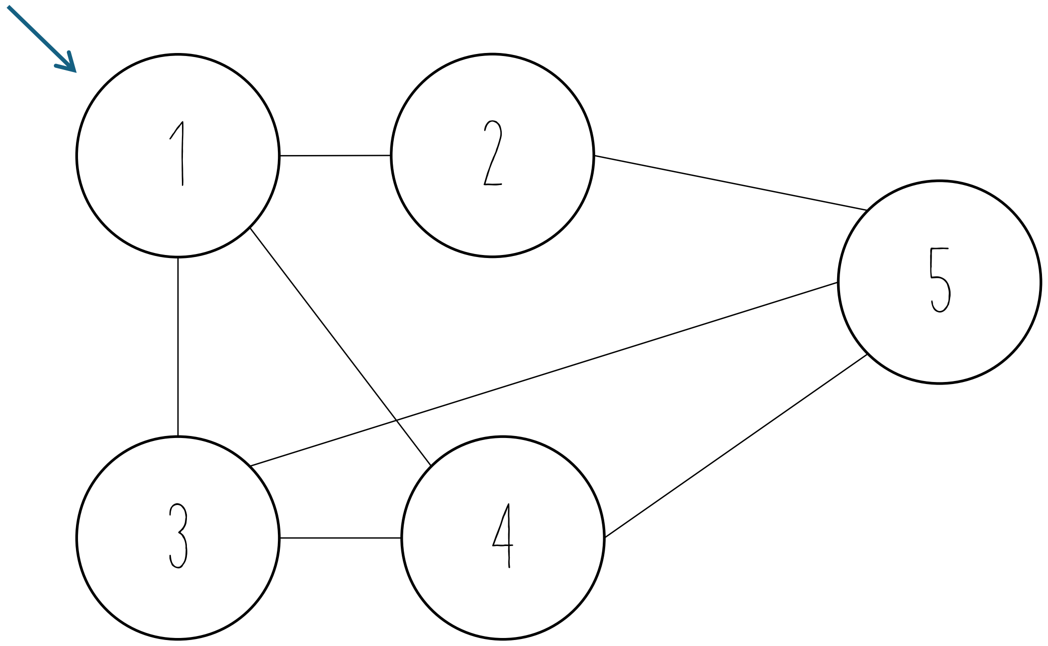
Farben = {}



RECAP: SEQUENZIELLER ALGORITHMUS

Module	Farben
1	–
2	–
3	–
4	–
5	–

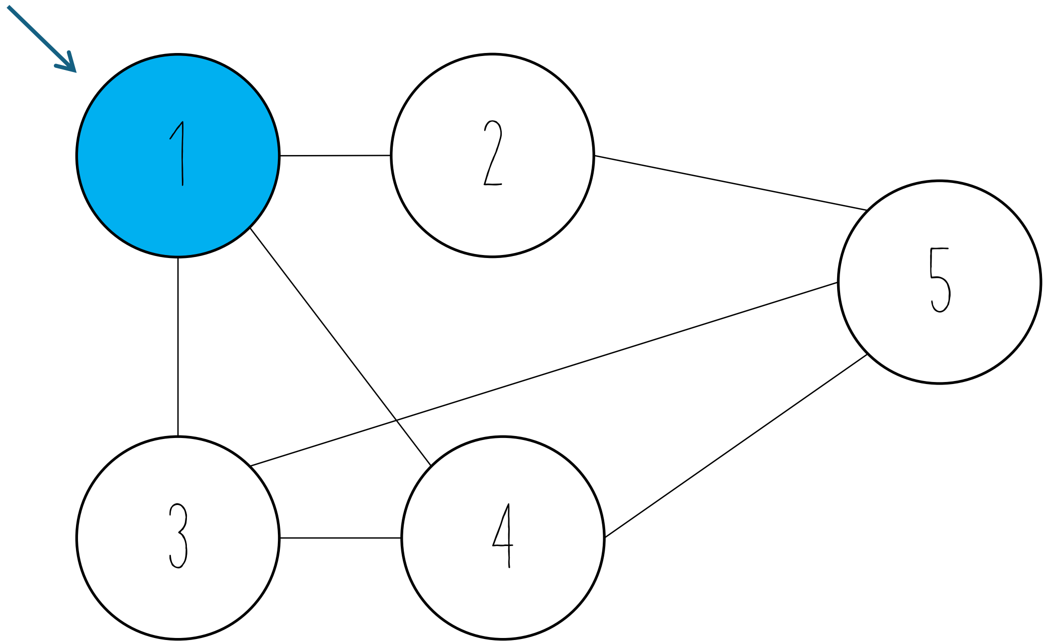
Farben = {}



RECAP: SEQUENZIELLER ALGORITHMUS

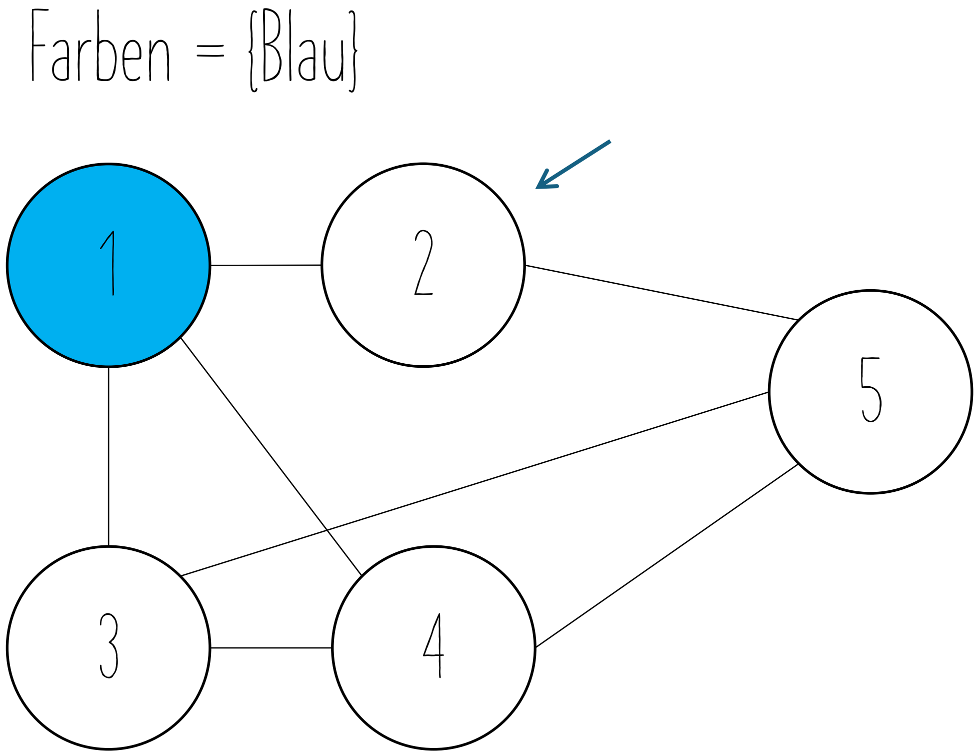
Module	Farben
1	Blau
2	-
3	-
4	-
5	-

Farben = {Blau}



RECAP: SEQUENZIELLER ALGORITHMUS

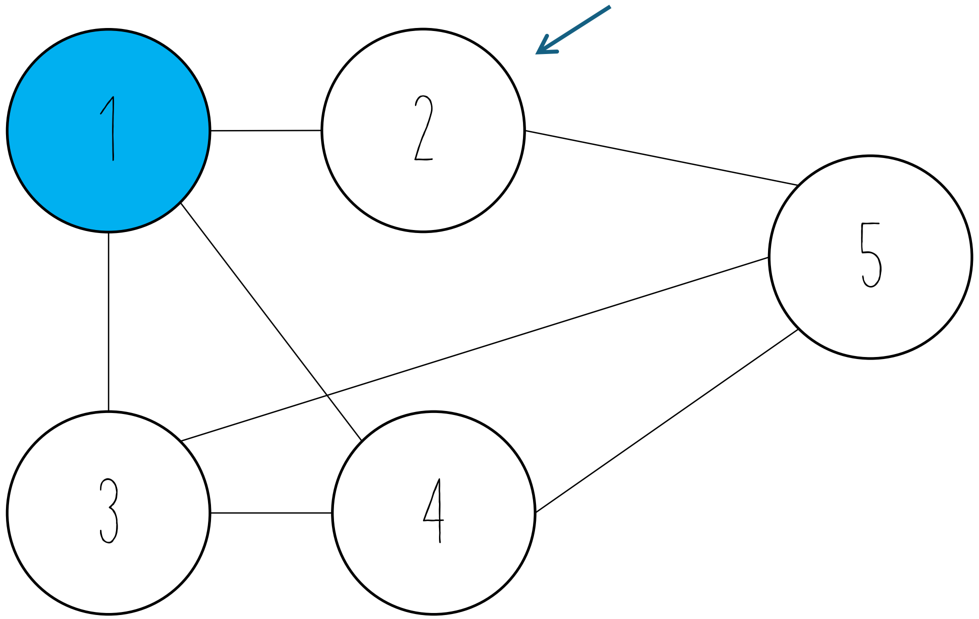
Module	Farben
1	Blau
2	-
3	-
4	-
5	-



RECAP: SEQUENZIELLER ALGORITHMUS

Module	Farben
1	Blau
2	–
3	–
4	–
5	–

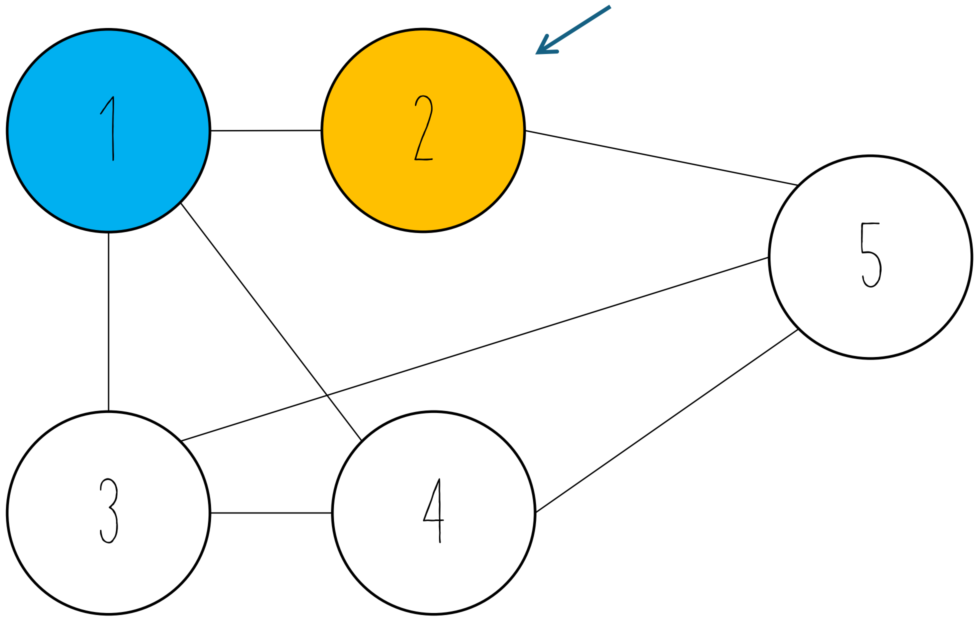
Farben = {~~Blau~~}



RECAP: SEQUENZIELLER ALGORITHMUS

Module	Farben
1	Blau
2	Orange
3	–
4	–
5	–

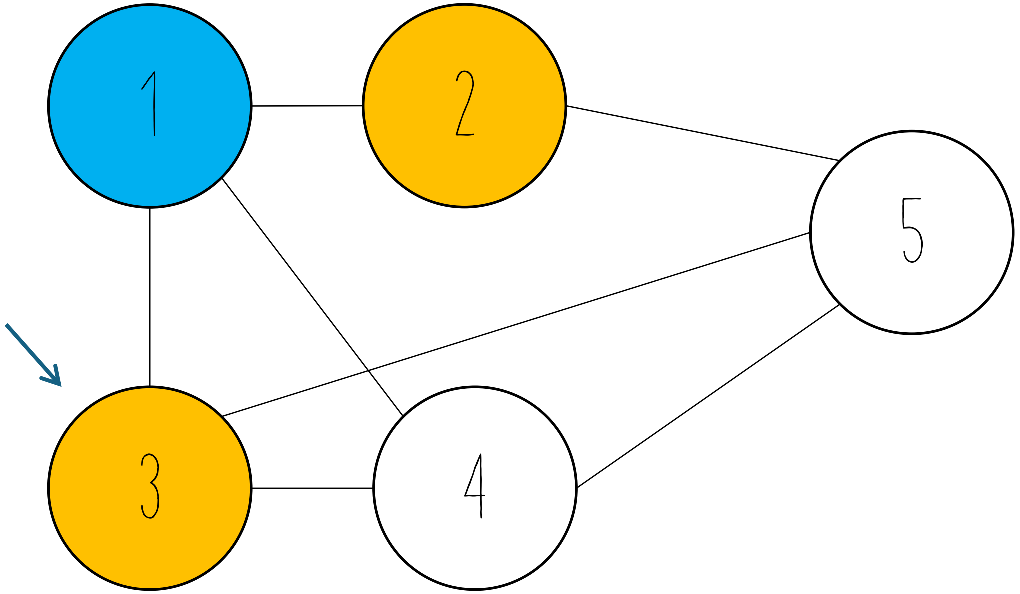
Farben = {~~Blau~~, Orange}



RECAP: SEQUENZIELLER ALGORITHMUS

Module	Farben
1	Blau
2	Orange
3	Orange
4	–
5	–

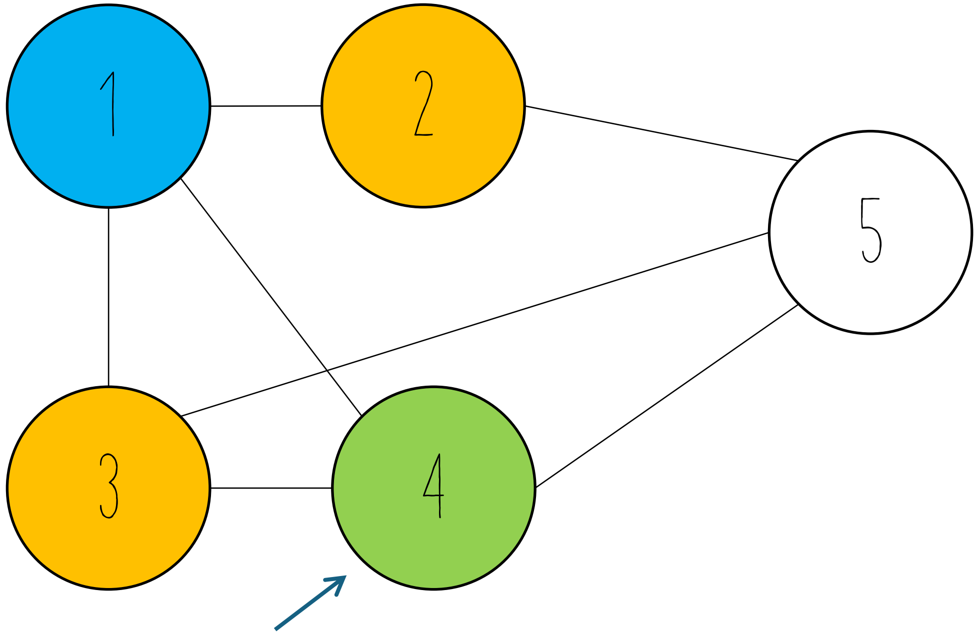
Farben = {~~Blau~~, Orange}



RECAP: SEQUENZIELLER ALGORITHMUS

Module	Farben
1	Blau
2	Orange
3	Orange
4	Grün
5	–

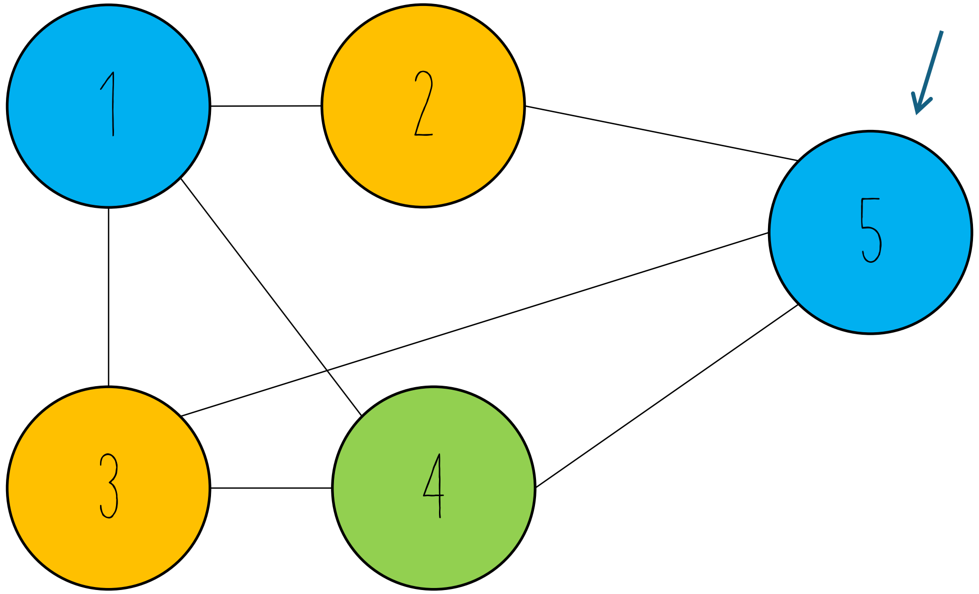
Farben = {~~Blau~~, ~~Orange~~, Grün}



RECAP: SEQUENZIELLER ALGORITHMUS

Module	Farben
1	Blau
2	Orange
3	Orange
4	Grün
5	Blau

Farben = {Blau, ~~Orange~~, ~~Grün~~}



- Nachteile:
  - Abhängig von der Reihenfolge
  - Liefert nicht die chromatische Farbe x(G)

RECAP: JOHNSON ALGORITHMUS

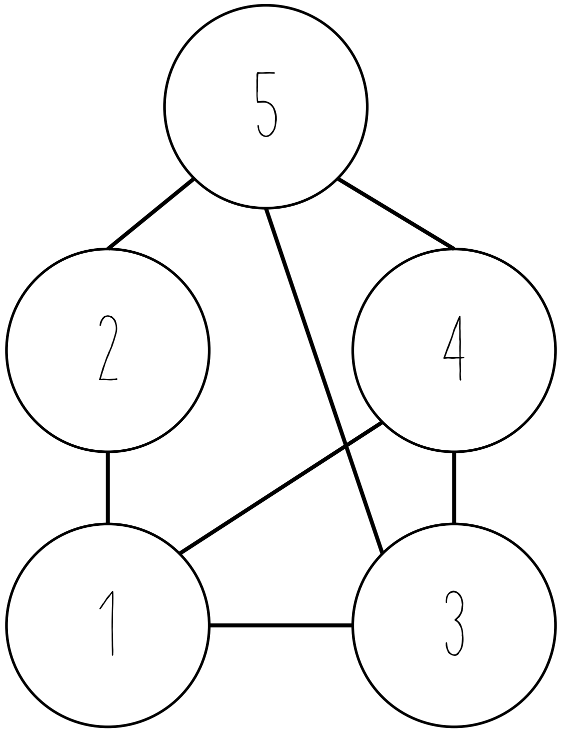
- Idee: Bestimmt die Färbung mithilfe des Knotengrads

Module	Grad	Farbe
1	3	-
2	2	-
3	3	-
4	3	-
5	3	-

Knotenmenge W

Module	Grad
1	3
2	2
3	3
4	3
5	3

Knotenmenge U



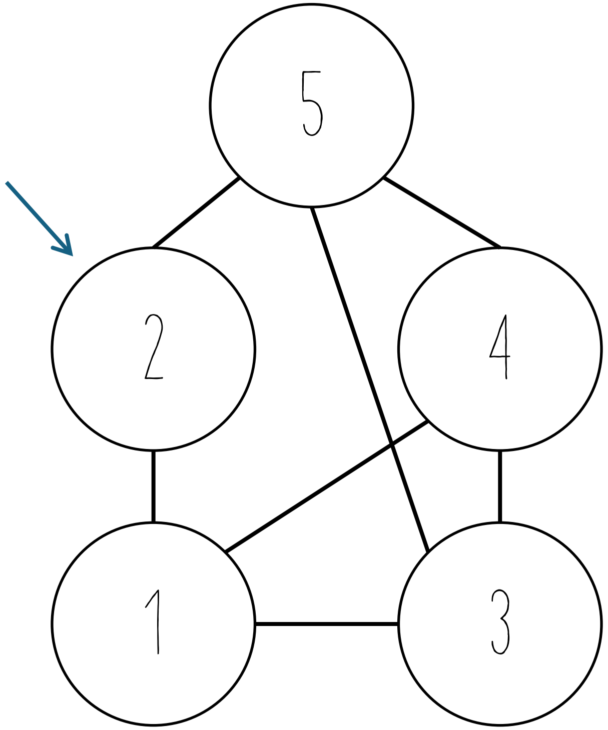
RECAP: JOHNSON ALGORITHMUS

Module	Grad	Farbe
1	3	–
2	2	–
3	3	–
4	3	–
5	3	–

Knotenmenge W

Module	Grad
1	3
2	2
3	3
4	3
5	3

Knotenmenge U



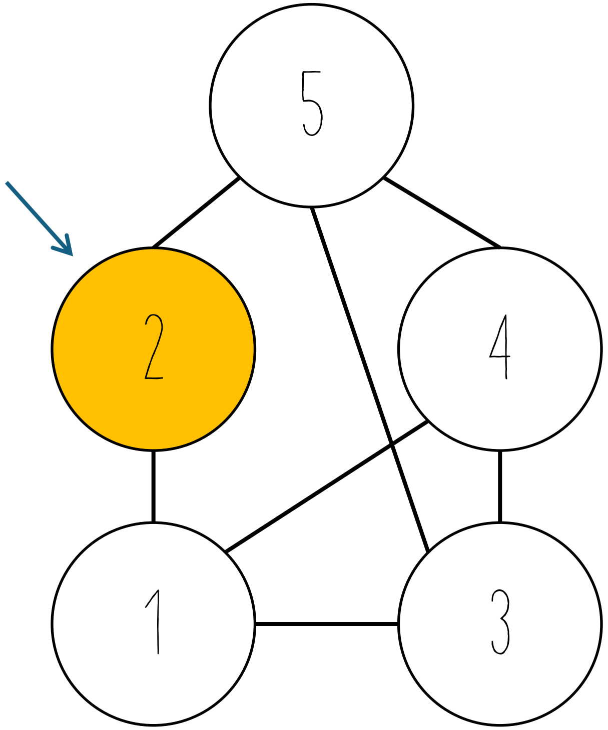
RECAP: JOHNSON ALGORITHMUS

Module	Grad	Farbe
1	3	–
2	2	Orange
3	3	–
4	3	–
5	3	–

Knotenmenge W

Module	Grad
1	3
2	2
3	3
4	3
5	3

Knotenmenge U



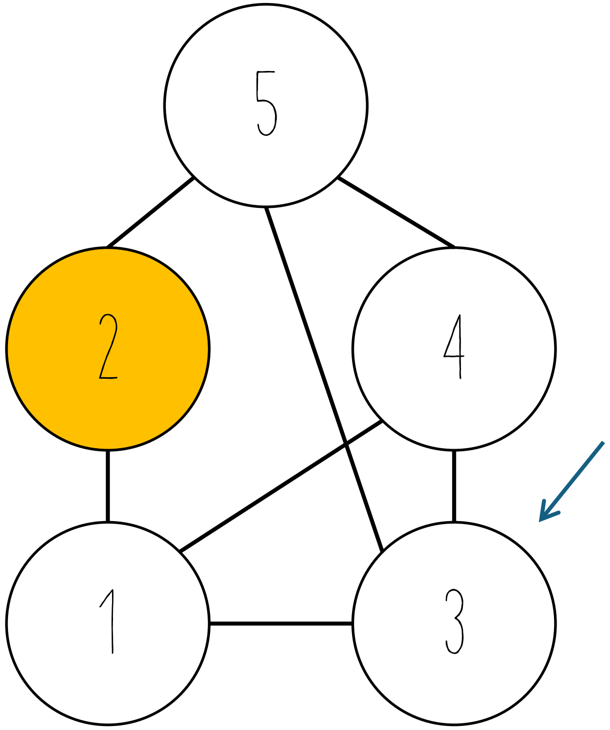
RECAP: JOHNSON ALGORITHMUS

Module	Grad	Farbe
1	2	–
2	–	Orange
3	3	–
4	3	–
5	2	–

Knotenmenge W

Module	Grad
3	1
4	1

Knotenmenge U



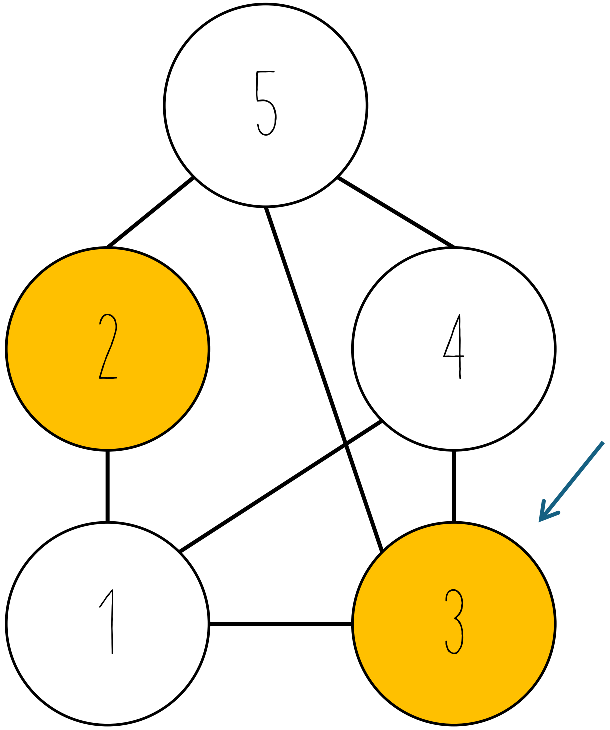
RECAP: JOHNSON ALGORITHMUS

Module	Grad	Farbe
1	2	–
2	–	Orange
3	–	Orange
4	2	–
5	1	–

Knotenmenge W

Module	Grad

Knotenmenge U



RECAP: JOHNSON ALGORITHMUS

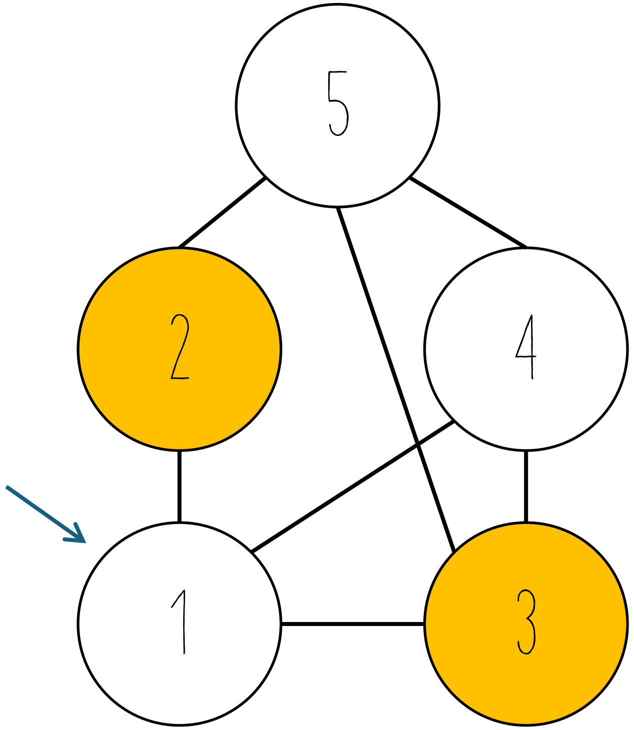
Module	Grad	Farbe
1	1	–
2	–	Orange
3	–	Orange
4	2	–
5	1	–

Knotenmenge W

Neue Farbe!

Module	Grad
1	1
4	2
5	1

Knotenmenge U



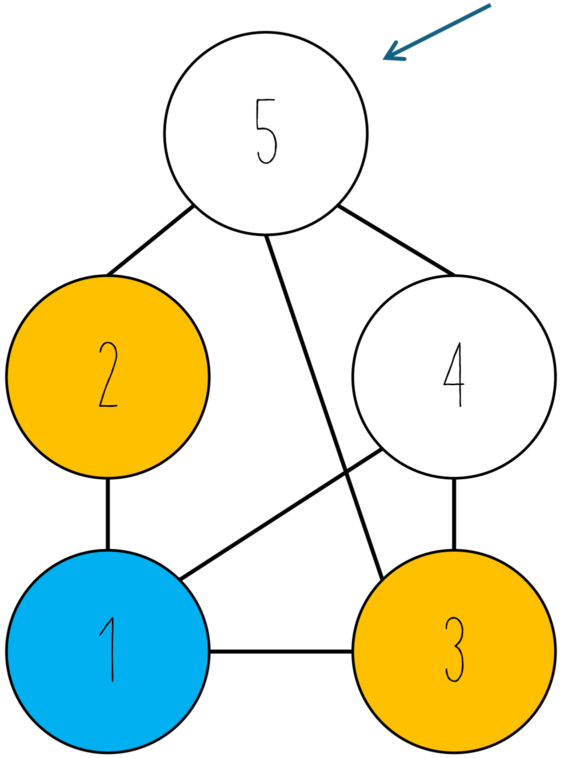
RECAP: JOHNSON ALGORITHMUS

Module	Grad	Farbe
1	–	Blau
2	–	Orange
3	–	Orange
4	1	–
5	1	–

Knotenmenge W

Module	Grad
5	0

Knotenmenge U



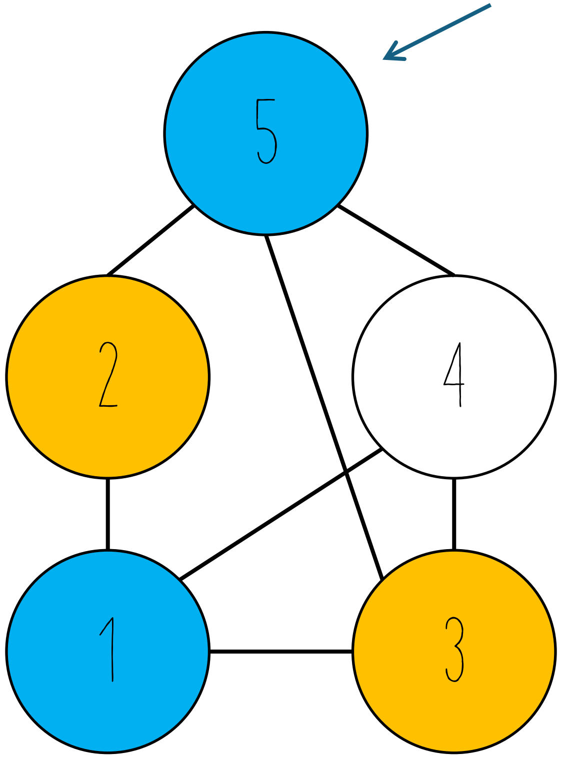
RECAP: JOHNSON ALGORITHMUS

Module	Grad	Farbe
1	–	Blau
2	–	Orange
3	–	Orange
4	0	–
5	–	Blau

Knotenmenge W

Module	Grad

Knotenmenge U



RECAP: JOHNSON ALGORITHMUS

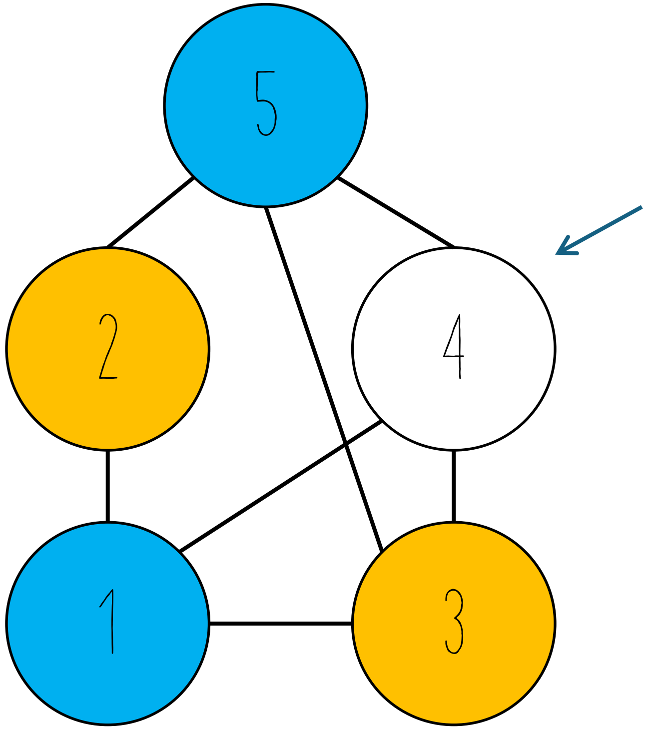
Module	Grad	Farbe
1	–	Blau
2	–	Orange
3	–	Orange
4	0	–
5	–	Blau

Knotenmenge W

Neue Farbe!

Module	Grad
4	0

Knotenmenge U



RECAP: JOHNSON ALGORITHMUS

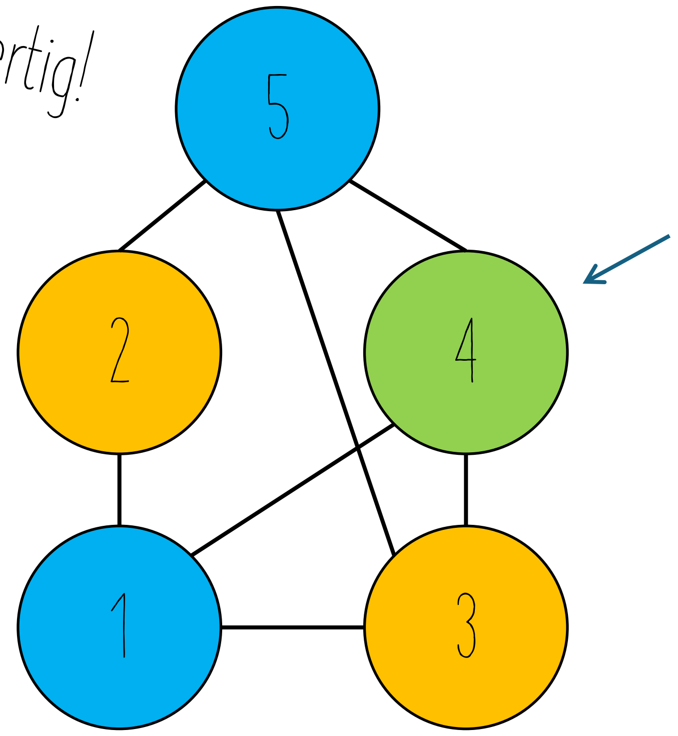
Module	Grad	Farbe
1	–	Blau
2	–	Orange
3	–	Orange
4	–	Grün
5	–	Blau

Knotenmenge W

Module	Grad

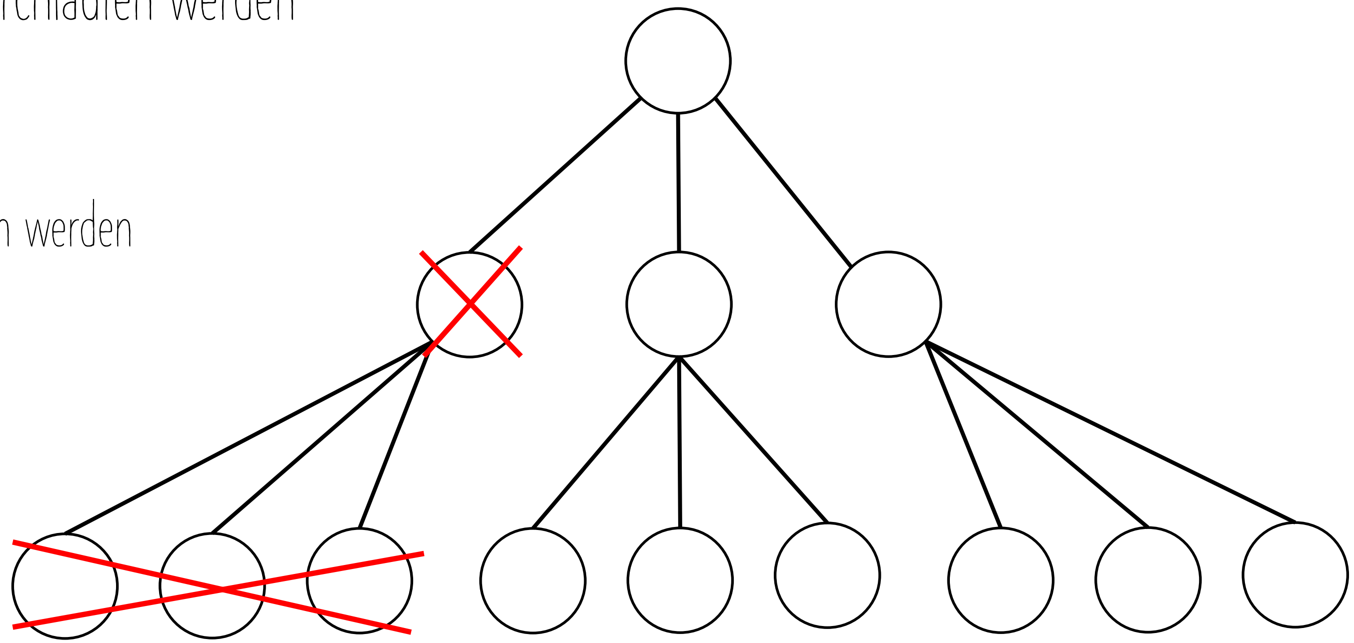
Knotenmenge U

Fertig!



RECAP: BACKTRACKING ALGORITHMUS

- Idee: Bestimme die Farbe, indem alle Möglichkeiten durchlaufen werden
- Baumstruktur stellt gesamten Lösungsraum dar
  - Wenn Möglichkeit invalide, muss Teilbaum nicht durlaufen werden
- Blätter entsprechen entweder nicht korrekter Teillösung oder Gesamtlösung
- Optimierungsproblem: Weiterführung nach Finden der Gesamtlösung



CODE & VISUALISIERUNG

Live Demo

LAUFZEITDISKUSSION

- Ab wann lohnt es sich die chromatische Farbe $\chi(G)$ zu bestimmen?
- Braucht man eine perfekte Lösung oder eine effiziente Lösung?

LAUFZEITDISKUSSION

- Ab wann lohnt es sich die chromatische Farbe x(G) zu bestimmen?
- Braucht man eine perfekte Lösung oder eine effiziente Lösung?
- Laufzeit in unseren Fällen:

Algorithmus	O-Notation	Laufzeit der Testinstanzen	Laufzeit bei 50 Modulen
Sequenzieller Algorithmus	$O(n*m) = O(n^3)$	ca. 0-2ms	ca. 0-2ms / Färbung: 7
Johnson Algorithmus	$O(n^3)$	ca. 0-2ms	ca. 0-2ms / Färbung: 7
Backtracking Algorithmus	$O(n^n)$	ca. 0-2ms	ca. 5min / Färbung: 5

LAUFZEITDISKUSSION - FAZIT

- Wahl des Algorithmus ist abhängig von Problem:
  - Einmalige oder Mehrfache Berechnungen notwendig?
  - Datengröße? Wie groß ist der Graph?
- Für unseren Fall: Backtracking Algorithmus geeignet
  - Keine großen Daten
  - Einmalige Berechnung

VIELEN DANK FÜR DIE AUFMERKSAMKEIT

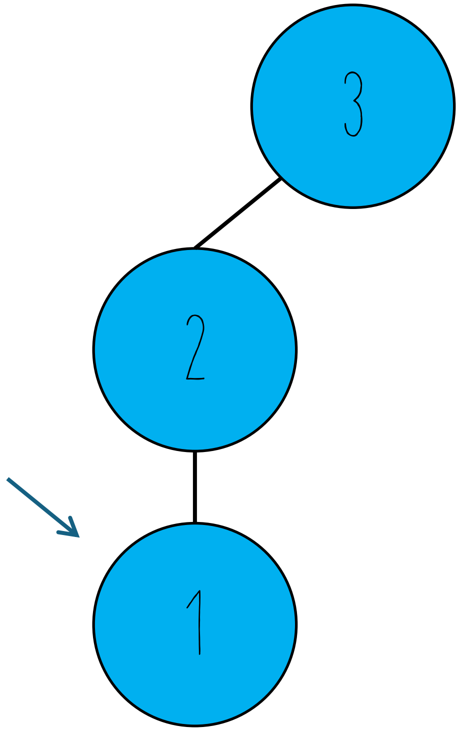
Gerne Fragen!

ZUSATZ: BACKTRACKING-VISUALISIERUNG

- Das Limit gibt die maximal möglichen Farben an

Knoten	Coloring	Maximum
1	1	1
2	1	3
3	1	3

Limit: 3

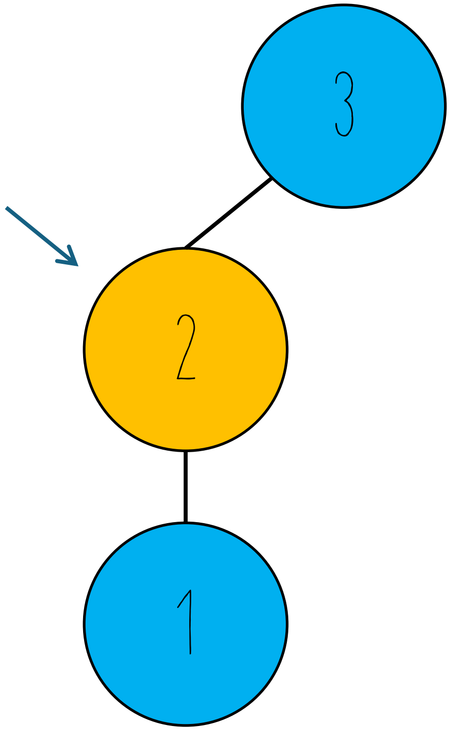


ZUSATZ: BACKTRACKING-VISUALISIERUNG

- Ist das eine korrekte Teillösung?
  - Nein: Knoten 1 und 2 haben die gleiche Farbe und sind benachbart
- Farbe für Knoten 2 wird angehoben

Knoten	Coloring	Maximum
1	1	1
2	2	3
3	1	3

Limit: 3

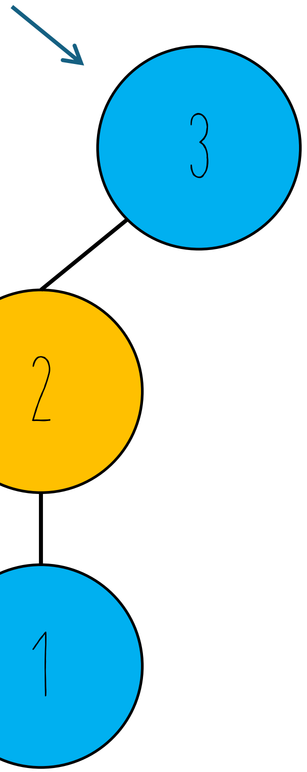


ZUSATZ: BACKTRACKING-VISUALISIERUNG

- Ist das eine korrekte Teillösung? Ja!
  - Anpassen des Maximums für Knoten 2
    - Formel: Max(coloring[k], maximum[k-1])
- Betrachten des nächsten Knotens

Knoten	Coloring	Maximum
1	1	1
2	2	2
3	1	3

Limit: 3

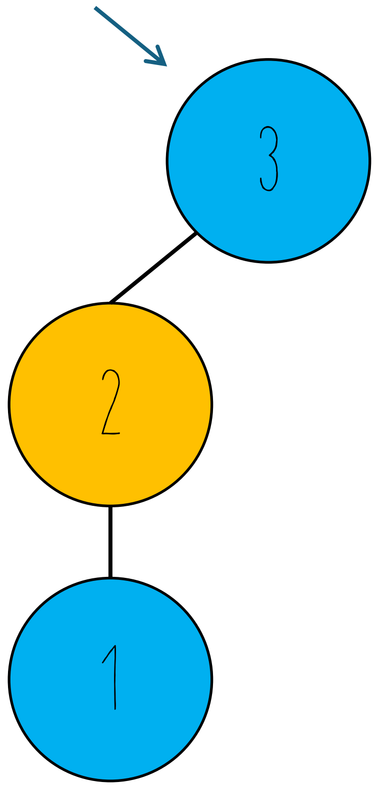


ZUSATZ: BACKTRACKING-VISUALISIERUNG

- Ist das eine korrekte Teillösung? Ja!
  - Anpassen des Maximums für Knoten 3
- Anpassen des Limits, weil mögliche Gesamtlösung
  - Limit entspricht Maximum von Knoten 3
- Gibt es eine bessere Lösung? Setze Algorithmus fort…

Knoten	Coloring	Maximum
1	1	1
2	2	2
3	1	2

Limit: 2

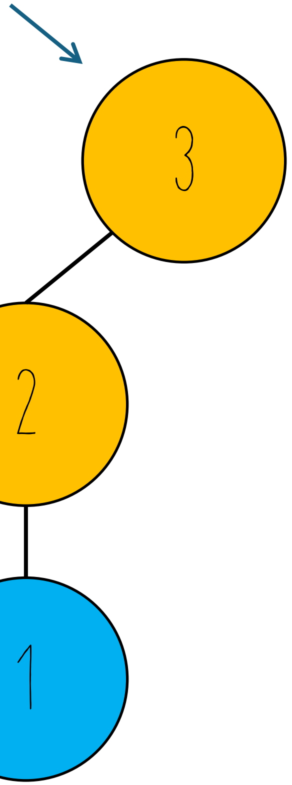


ZUSATZ: BACKTRACKING-VISUALISIERUNG

- Kann ich die Farbe für Knoten 3 noch verändern?
  - Ja, aktuelle Farbe entspricht nicht dem Maximum
  - Probieren der nächsten Farbe

Knoten	Coloring	Maximum
1	1	1
2	2	2
3	2	2

Limit: 2

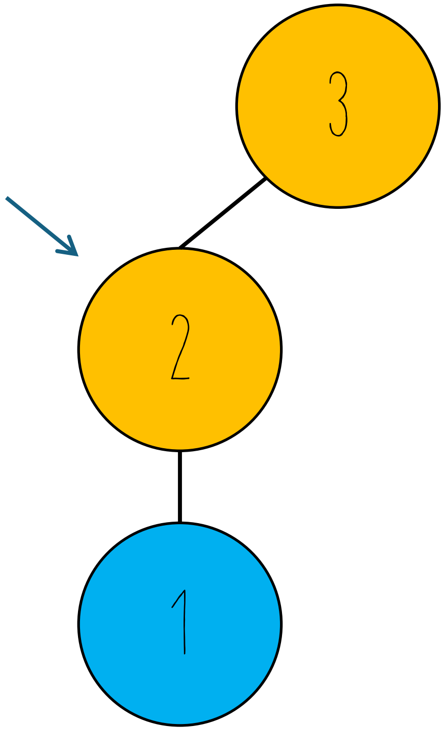


ZUSATZ: BACKTRACKING-VISUALISIERUNG

- Ist das eine korrekte Teillösung?
  - Nein: Knoten 2 und 3 haben die gleiche Farbe und sind benachbart
- Ende für Knoten 3 erreicht, betrachte Knoten 2…

Knoten	Coloring	Maximum
1	1	1
2	2	2
3	2	2

Limit: 2

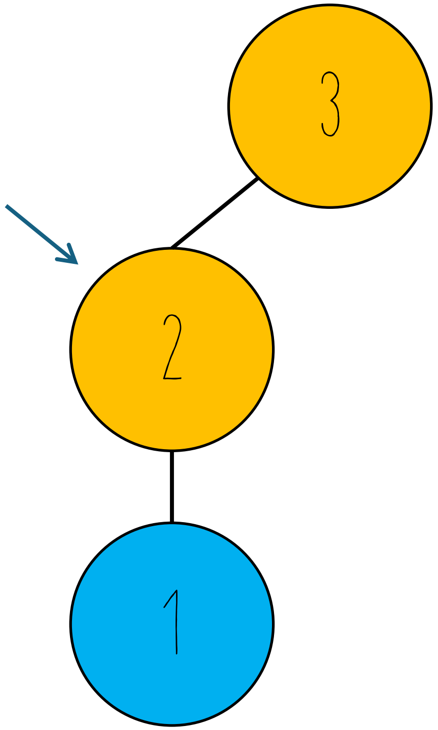


ZUSATZ: BACKTRACKING-VISUALISIERUNG

- Kann ich die Farbe für Knoten 2 noch verändern?
  - Nein: Maximum wird überschritten
- Ende für Knoten 2 erreicht
- Knoten 1 muss nicht nochmal betrachtet werden

Knoten	Coloring	Maximum
1	1	1
2	2	2
3	2	2

Limit: 2



Zusatz: Backtracking-Visualisierung

- Alle Lösungen durchgegangen
- Das ist die korrekte Lösung!

Knoten	Coloring	Maximum
1	1	1
2	2	2
3	1	2

Limit: 2