

# Distributed Node Coloring for Directed Graphs

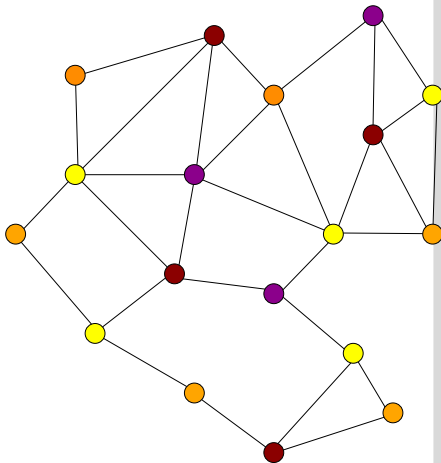
Bachelorarbeit, Betreuer: Fabian Fuchs, Roman Prutkin

Manuel Schweigert | 6. Juni 2014

INSTITUT FÜR THEORETISCHE INFORMATIK

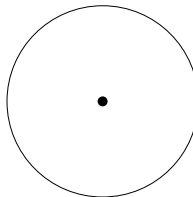


- Knotenfärbung klassisch
- Wofür braucht man das heute? (TDMA)
- Was sind die theoretischen Modelle? (SINR, Congest)



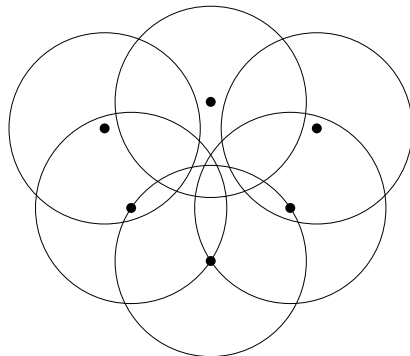
## Kommunikationsgraph

- Knoten mit Sendereichweite
- Bisher immer uniform
- Resultiert in einem ungerichteten Kommunikationsgraph
- Szenario: verschieden starke Sendeleistung
- Resultat: gerichteter Kommunikationsgraph



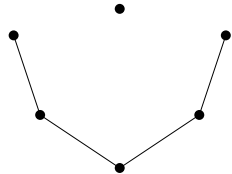
## Kommunikationsgraph

- Knoten mit Sendereichweite
- Bisher immer uniform
- Resultiert in einem ungerichteten Kommunikationsgraph
- Szenario: verschieden starke Sendeleistung
- Resultat: gerichteter Kommunikationsgraph



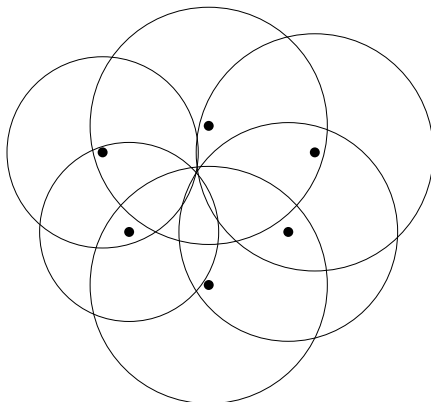
## Kommunikationsgraph

- Knoten mit Sendereichweite
- Bisher immer uniform
- Resultiert in einem ungerichteten Kommunikationsgraph
- Szenario: verschieden starke Sendeleistung
- Resultat: gerichteter Kommunikationsgraph



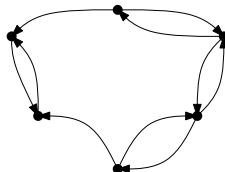
## Kommunikationsgraph

- Knoten mit Sendereichweite
- Bisher immer uniform
- Resultiert in einem ungerichteten Kommunikationsgraph
- Szenario: verschieden starke Sendeleistung
- Resultat: gerichteter Kommunikationsgraph



## Kommunikationsgraph

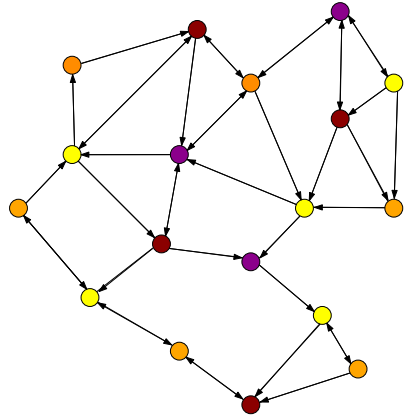
- Knoten mit Sendereichweite
- Bisher immer uniform
- Resultiert in einem ungerichteten Kommunikationsgraph
- Szenario: verschieden starke Sendeleistung
- Resultat: gerichteter Kommunikationsgraph



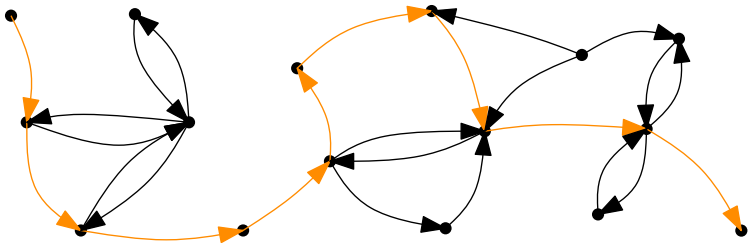
- 1 Einführung
- 2 Definitionen
- 3 Algorithmen
  - Mit Initialisierung
  - Ohne Initialisierung
- 4 Untere Schranke
- 5 Experiment
- 6 Fazit



- $\Delta :=$  maximaler Eingangsgrad
- valide Knoten-Färbung im gerichteten Graph  
:= Nachbarn eingehender Kanten haben andere Farbe als Knoten



- Dominierender Knoten  $:=$  Knoten  $u$ , sodass  $\exists$  Knoten  $v$  mit  $(u, v) \in E$  aber  $(v, u) \notin E$
- $l :=$  Länge des längsten dominierenden Pfades im Graph



## Grundidee:

- In jeder Runde:
- Jeder Knoten wählt sich zufällig eine Farbe
- Wenn konfliktfrei  $\Rightarrow$  „reserviere“ Farbe und terminiere
- Nachbarknoten respektieren reservierte Farben in der Zukunft

## Grundidee:

- In jeder Runde:
- Jeder Knoten wählt sich zufällig eine Farbe
- Wenn konfliktfrei  $\Rightarrow$  „reserviere“ Farbe und terminiere
- Nachbarknoten respektieren reservierte Farben in der Zukunft

## Grundidee:

- In jeder Runde:
- Jeder Knoten wählt sich zufällig eine Farbe
- Wenn konfliktfrei  $\Rightarrow$  „reserviere“ Farbe und terminiere
- Nachbarknoten respektieren reservierte Farben in der Zukunft

## Grundidee:

- In jeder Runde:
- Jeder Knoten wählt sich zufällig eine Farbe
- Wenn konfliktfrei  $\Rightarrow$  „reserviere“ Farbe und terminiere
- Nachbarknoten respektieren reservierte Farben in der Zukunft

## Skizze

- Finde dominierende Knoten ( $D_v$ )
- Hauptschleife
  - Mit Wahrscheinlichkeit  $1/2$  setze aus
  - Wähle zufällige Farbe von  $\Delta + 1$  Farben
  - Broadcaste die Farbe an alle Nachbarn
  - Wenn kein Nachbar die gleiche Farbe hat und alle Knoten in  $D_v$  fertig sind
    - sende die gewählte Farbe als finale Farbe und terminiere

## Skizze

- Finde dominierende Knoten ( $D_V$ )
- Hauptschleife
  - Mit Wahrscheinlichkeit  $1/2$  setze aus
  - Wähle zufällige Farbe von  $\Delta + 1$  Farben
  - Broadcaste die Farbe an alle Nachbarn
  - Wenn kein Nachbar die gleiche Farbe hat und alle Knoten in  $D_V$  fertig sind
    - sende die gewählte Farbe als finale Farbe und terminiere



## Skizze

- Finde dominierende Knoten ( $D_V$ )
- Hauptschleife
  - Mit Wahrscheinlichkeit  $1/2$  setze aus
  - Wähle zufällige Farbe von  $\Delta + 1$  Farben
  - Broadcaste die Farbe an alle Nachbarn
  - Wenn kein Nachbar die gleiche Farbe hat und alle Knoten in  $D_V$  fertig sind
    - sende die gewählte Farbe als finale Farbe und terminiere

## Skizze

- Finde dominierende Knoten ( $D_v$ )
- Hauptschleife
  - Mit Wahrscheinlichkeit  $1/2$  setze aus
  - Wähle zufällige Farbe von  $\Delta + 1$  Farben
  - Broadcaste die Farbe an alle Nachbarn
  - Wenn kein Nachbar die gleiche Farbe hat und alle Knoten in  $D_v$  fertig sind
    - sende die gewählte Farbe als finale Farbe und terminiere

## Skizze

- Finde dominierende Knoten ( $D_v$ )
- Hauptschleife
  - Mit Wahrscheinlichkeit  $1/2$  setze aus
  - Wähle zufällige Farbe von  $\Delta + 1$  Farben
  - Broadcaste die Farbe an alle Nachbarn
  - Wenn kein Nachbar die gleiche Farbe hat und alle Knoten in  $D_v$  fertig sind
    - sende die gewählte Farbe als finale Farbe und terminiere

## Skizze

- Finde dominierende Knoten ( $D_V$ )
- Hauptschleife
  - Mit Wahrscheinlichkeit  $1/2$  setze aus
  - Wähle zufällige Farbe von  $\Delta + 1$  Farben
  - Broadcaste die Farbe an alle Nachbarn
  - Wenn kein Nachbar die gleiche Farbe hat und alle Knoten in  $D_V$  fertig sind
    - sende die gewählte Farbe als finale Farbe und terminiere

## Subgraph-Hierarchie:

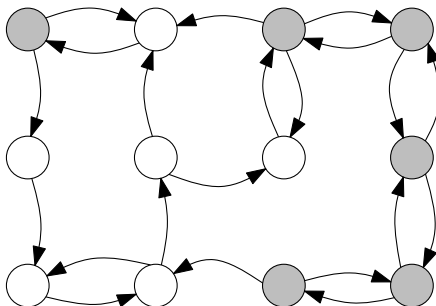


Abbildung: Subgraph  $G_1$ , alle nicht-dominierten Knoten

## Subgraph-Hierarchie:

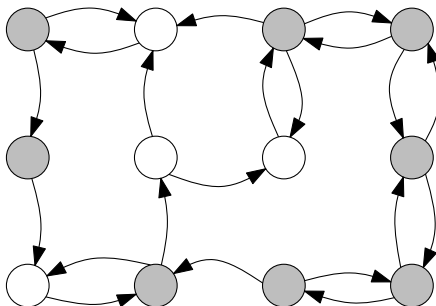


Abbildung: Subgraph  $G_2$ , alle Knoten, die nur aus Knoten aus  $G_1$  dominiert werden

## Subgraph-Hierarchie:

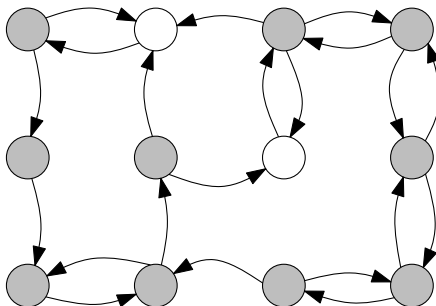


Abbildung: Subgraph  $G_3$ , alle Knoten, die nur aus Knoten aus  $G_2$  dominiert werden

## Subgraph-Hierarchie:

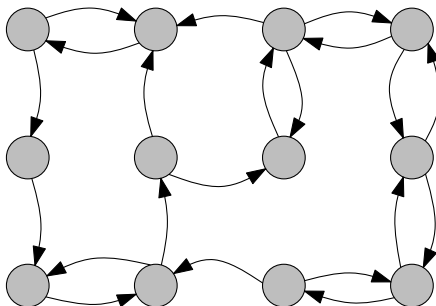


Abbildung: Subgraph  $G_4 = G$ , alle Knoten, die nur aus Knoten aus  $G_3$  dominiert werden



- Subgraph-Hierarchie
- Finde kleinstes  $l$  sodass  $G_{l+1} = G$
- Zeige induktiv, dass Färbung von  $G_k \rightarrow G_{k+1}$  max.  $O(\log n)$  benötigt (mit hoher Wahrscheinlichkeit)
- Laufzeit setzt sich aus Initialisierung ( $O(\Delta)$ ) und  $(l + 1) \cdot O(\log n)$  zusammen

$\Rightarrow$  Laufzeit  $O(\Delta + l \log n)$

- Subgraph-Hierarchie
- Finde kleinstes  $l$  sodass  $G_{l+1} = G$
- Zeige induktiv, dass Färbung von  $G_k \rightarrow G_{k+1}$  max.  $O(\log n)$  benötigt (mit hoher Wahrscheinlichkeit)
- Laufzeit setzt sich aus Initialisierung ( $O(\Delta)$ ) und  $(l + 1) \cdot O(\log n)$  zusammen

$\Rightarrow$  Laufzeit  $O(\Delta + l \log n)$

- Subgraph-Hierarchie
- Finde kleinstes  $l$  sodass  $G_{l+1} = G$
- Zeige induktiv, dass Färbung von  $G_k \rightarrow G_{k+1}$  max.  $O(\log n)$  benötigt (mit hoher Wahrscheinlichkeit)
- Laufzeit setzt sich aus Initialisierung ( $O(\Delta)$ ) und  $(l + 1) \cdot O(\log n)$  zusammen

⇒ Laufzeit  $O(\Delta + l \log n)$

- Subgraph-Hierarchie
- Finde kleinstes  $l$  sodass  $G_{l+1} = G$
- Zeige induktiv, dass Färbung von  $G_k \rightarrow G_{k+1}$  max.  $O(\log n)$  benötigt (mit hoher Wahrscheinlichkeit)
- Laufzeit setzt sich aus Initialisierung ( $O(\Delta)$ ) und  $(l + 1) \cdot O(\log n)$  zusammen

$\Rightarrow$  Laufzeit  $O(\Delta + l \log n)$

- Betrachte Knoten  $v \in G_{k+1}$ , der in der letzten Runde in einem Konflikt war und Münzwurf gewinnt
- Wahrscheinlichkeit für  $v$  wieder im Konflikt ist
$$\leq (\Delta + 1 - |F_v|) \frac{1}{2(\Delta+1-|F_v|)} = \frac{1}{2}$$
- $\Rightarrow v$  terminiert mit Wahrscheinlichkeit  $\geq (1/2 \cdot 1/2) = 1/4$
- In Runde  $i$  hat  $v$  mit Wahrscheinlichkeit  $\leq (1 - 1/4)^i$  nicht terminiert
- $\Rightarrow$  Nach  $(c + 1)4 \log n$  Runden hat  $v$  mit Wahrscheinlichkeit  $\leq 1 - 1/n^c$  nicht terminiert

- Betrachte Knoten  $v \in G_{k+1}$ , der in der letzten Runde in einem Konflikt war und Münzwurf gewinnt
- Wahrscheinlichkeit für  $v$  wieder im Konflikt ist
$$\leq (\Delta + 1 - |F_v|) \frac{1}{2(\Delta+1-|F_v|)} = \frac{1}{2}$$
- $\Rightarrow v$  terminiert mit Wahrscheinlichkeit  $\geq (1/2 \cdot 1/2) = 1/4$
- In Runde  $i$  hat  $v$  mit Wahrscheinlichkeit  $\leq (1 - 1/4)^i$  nicht terminiert
- $\Rightarrow$  Nach  $(c + 1)4 \log n$  Runden hat  $v$  mit Wahrscheinlichkeit  $\leq 1 - 1/n^c$  nicht terminiert

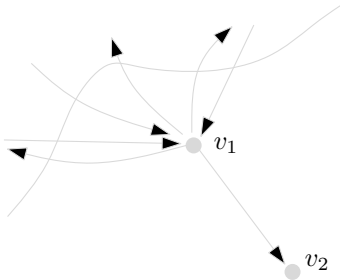
- Betrachte Knoten  $v \in G_{k+1}$ , der in der letzten Runde in einem Konflikt war und Münzwurf gewinnt
- Wahrscheinlichkeit für  $v$  wieder im Konflikt ist
$$\leq (\Delta + 1 - |F_v|) \frac{1}{2(\Delta+1-|F_v|)} = \frac{1}{2}$$
- $\Rightarrow v$  terminiert mit Wahrscheinlichkeit  $\geq (1/2 \cdot 1/2) = 1/4$
- In Runde  $i$  hat  $v$  mit Wahrscheinlichkeit  $\leq (1 - 1/4)^i$  nicht terminiert
- $\Rightarrow$  Nach  $(c + 1)4 \log n$  Runden hat  $v$  mit Wahrscheinlichkeit  $\leq 1 - 1/n^c$  nicht terminiert

- Betrachte Knoten  $v \in G_{k+1}$ , der in der letzten Runde in einem Konflikt war und Münzwurf gewinnt
- Wahrscheinlichkeit für  $v$  wieder im Konflikt ist
$$\leq (\Delta + 1 - |F_v|) \frac{1}{2(\Delta+1-|F_v|)} = \frac{1}{2}$$
- $\Rightarrow v$  terminiert mit Wahrscheinlichkeit  $\geq (1/2 \cdot 1/2) = 1/4$
- In Runde  $i$  hat  $v$  mit Wahrscheinlichkeit  $\leq (1 - 1/4)^i$  nicht terminiert
- $\Rightarrow$  Nach  $(c+1)4 \log n$  Runden hat  $v$  mit Wahrscheinlichkeit  $\leq 1 - 1/n^c$  nicht terminiert

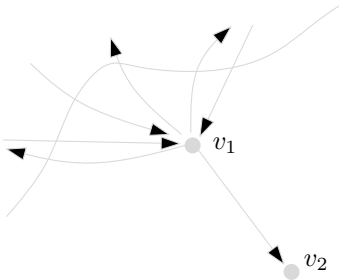


- Betrachte Knoten  $v \in G_{k+1}$ , der in der letzten Runde in einem Konflikt war und Münzwurf gewinnt
- Wahrscheinlichkeit für  $v$  wieder im Konflikt ist
$$\leq (\Delta + 1 - |F_v|) \frac{1}{2(\Delta+1-|F_v|)} = \frac{1}{2}$$
- $\Rightarrow v$  terminiert mit Wahrscheinlichkeit  $\geq (1/2 \cdot 1/2) = 1/4$
- In Runde  $i$  hat  $v$  mit Wahrscheinlichkeit  $\leq (1 - 1/4)^i$  nicht terminiert
- $\Rightarrow$  Nach  $(c + 1)4 \log n$  Runden hat  $v$  mit Wahrscheinlichkeit  $\leq 1 - 1/n^c$  nicht terminiert

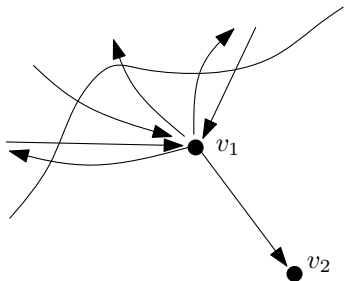
- Initialisierung beansprucht  $O(\Delta)$  Broadcasts
- Versuch: Algorithmus ohne Initialisierung
- Problem: Knoten kann sich nicht mehr sicher sein, ob er fertig ist oder nicht
- Resultat: Nachbarn respektieren und Farben halten



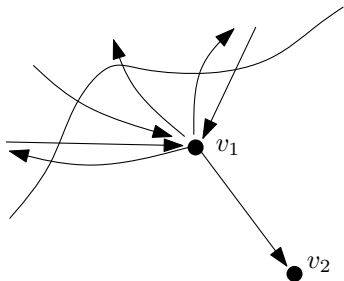
- Initialisierung beansprucht  $O(\Delta)$  Broadcasts
- Versuch: Algorithmus ohne Initialisierung
- Problem: Knoten kann sich nicht mehr sicher sein, ob er fertig ist oder nicht
- Resultat: Nachbarn respektieren und Farben halten



- Initialisierung beansprucht  $O(\Delta)$  Broadcasts
- Versuch: Algorithmus ohne Initialisierung
- Problem: Knoten kann sich nicht mehr sicher sein, ob er fertig ist oder nicht
- Resultat: Nachbarn respektieren und Farben halten



- Initialisierung beansprucht  $O(\Delta)$  Broadcasts
- Versuch: Algorithmus ohne Initialisierung
- Problem: Knoten kann sich nicht mehr sicher sein, ob er fertig ist oder nicht
- Resultat: Nachbarn respektieren und Farben halten



## Skizze

### ■ Hauptschleife

- Behalte die Farbe aus der letzten Runde (initial „0“)
- Broadcaste aktuelle Farbe (Nachbarfarben dieser Runde  $\rightarrow T$ )
- Bei Konflikt
  - Mit Wahrscheinlichkeit  $1/2$  wähle „0“ als Farbe, sonst
  - Wähle neue Farbe aus  $[2\Delta + 1] \setminus T$

## Skizze

- Hauptschleife
  - Behalte die Farbe aus der letzten Runde (initial „0“)
  - Broadcaste aktuelle Farbe (Nachbarfarben dieser Runde  $\rightarrow T$ )
  - Bei Konflikt
    - Mit Wahrscheinlichkeit  $1/2$  wähle „0“ als Farbe, sonst
    - Wähle neue Farbe aus  $[2\Delta + 1] \setminus T$

## Skizze

- Hauptschleife
  - Behalte die Farbe aus der letzten Runde (initial „0“)
  - Broadcaste aktuelle Farbe (Nachbarfarben dieser Runde  $\rightarrow T$ )
  - Bei Konflikt
    - Mit Wahrscheinlichkeit  $1/2$  wähle „0“ als Farbe, sonst
    - Wähle neue Farbe aus  $[2\Delta + 1] \setminus T$



## Skizze

- Hauptschleife
  - Behalte die Farbe aus der letzten Runde (initial „0“)
  - Broadcaste aktuelle Farbe (Nachbarfarben dieser Runde  $\rightarrow T$ )
  - Bei Konflikt
    - Mit Wahrscheinlichkeit  $1/2$  wähle „0“ als Farbe, sonst
    - Wähle neue Farbe aus  $[2\Delta + 1] \setminus T$

## Skizze

- Hauptschleife
  - Behalte die Farbe aus der letzten Runde (initial „0“)
  - Broadcaste aktuelle Farbe (Nachbarfarben dieser Runde  $\rightarrow T$ )
  - Bei Konflikt
    - Mit Wahrscheinlichkeit  $1/2$  wähle „0“ als Farbe, sonst
    - Wähle neue Farbe aus  $[2\Delta + 1] \setminus T$

## Skizze

- Hauptschleife
  - Behalte die Farbe aus der letzten Runde (initial „0“)
  - Broadcaste aktuelle Farbe (Nachbarfarben dieser Runde  $\rightarrow T$ )
  - Bei Konflikt
    - Mit Wahrscheinlichkeit  $1/2$  wähle „0“ als Farbe, sonst
    - Wähle neue Farbe aus  $[2\Delta + 1] \setminus T$

- Gleiche Graph-Hierarchie
- Statt terminiertem Knoten beweisen wir, dass Knoten „stabil“ werden
- Induktionsschritt weitestgehend gleich
- Farben müssen erweitert werden, da alle Nachbarfarben als „final“ angesehen werden
- Laufzeit:  $O(l \log n)$

- Gleiche Graph-Hierarchie
- Statt terminiertem Knoten beweisen wir, dass Knoten „stabil“ werden
- Induktionsschritt weitestgehend gleich
- Farben müssen erweitert werden, da alle Nachbarfarben als „final“ angesehen werden
- Laufzeit:  $O(l \log n)$

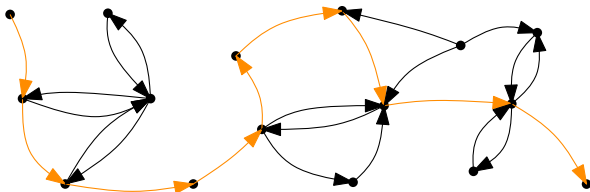
- Gleiche Graph-Hierarchie
- Statt terminiertem Knoten beweisen wir, dass Knoten „stabil“ werden
- Induktionsschritt weitestgehend gleich
- Farben müssen erweitert werden, da alle Nachbarfarben als „final“ angesehen werden
- Laufzeit:  $O(l \log n)$

- Gleiche Graph-Hierarchie
- Statt terminiertem Knoten beweisen wir, dass Knoten „stabil“ werden
- Induktionsschritt weitestgehend gleich
- Farben müssen erweitert werden, da alle Nachbarfarben als „final“ angesehen werden
- Laufzeit:  $O(l \log n)$

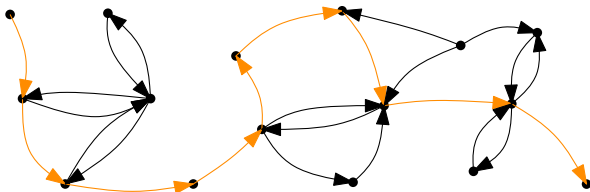
- Gleiche Graph-Hierarchie
- Statt terminiertem Knoten beweisen wir, dass Knoten „stabil“ werden
- Induktionsschritt weitestgehend gleich
- Farben müssen erweitert werden, da alle Nachbarfarben als „final“ angesehen werden
- Laufzeit:  $O(l \log n)$



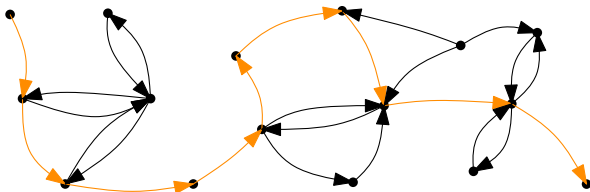
- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein



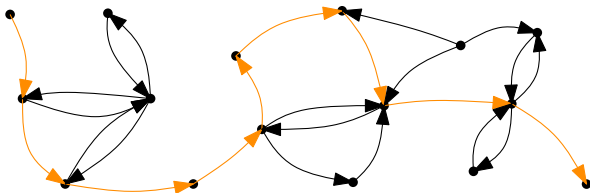
- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein



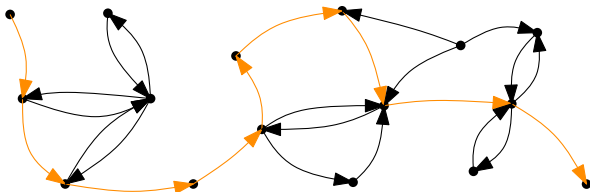
- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein



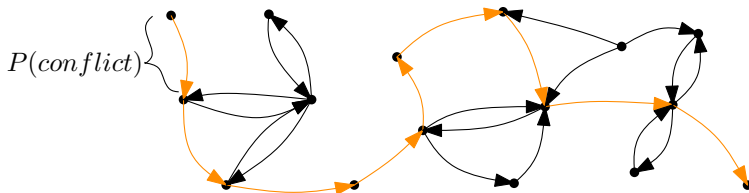
- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein



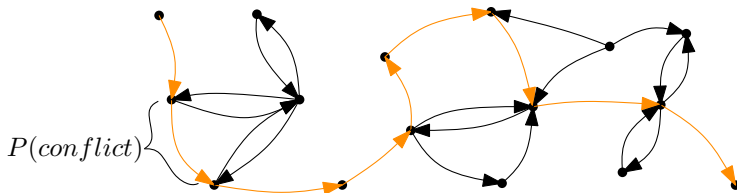
- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein



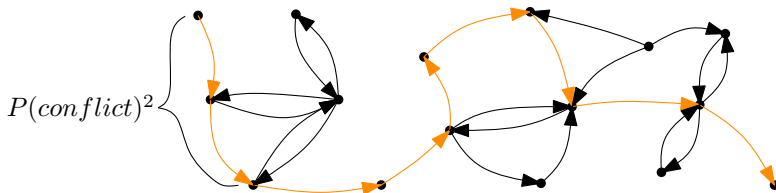
- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein



- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein

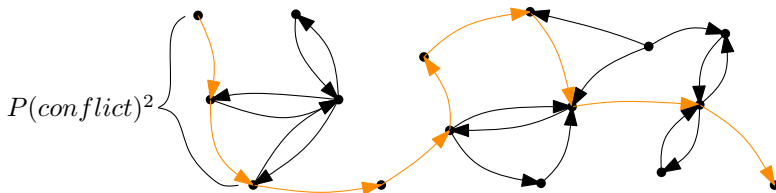


- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein

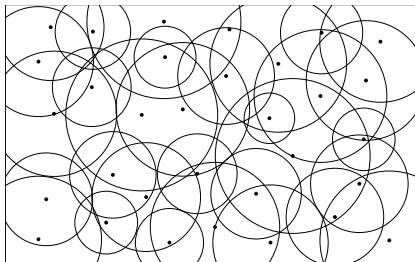




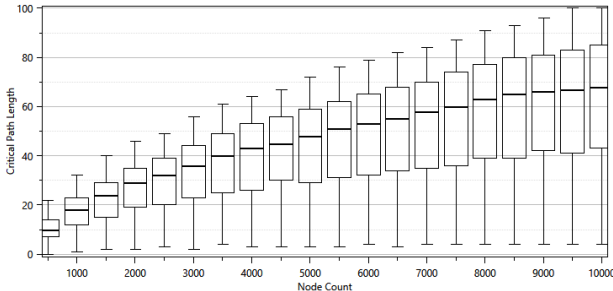
- Für allgemeine Färbealgorithmen mit  $O(\Delta)$  Farben:
- Deterministische Algorithmen haben untere Schranke  $\Omega(l)$  (Kritischer Pfad)
- Randomisierte Algorithmen haben untere Schranke  $\Omega(\min(l, \log n))$
- Beweisidee:
  - Betrachte wie ein Konflikt propagiert
  - Konfliktwahrscheinlichkeit kann erst nach  $\Omega(\log n)$  Schritten „klein“ ( $\leq 1 - 1/n^c, c > 1$ ) sein



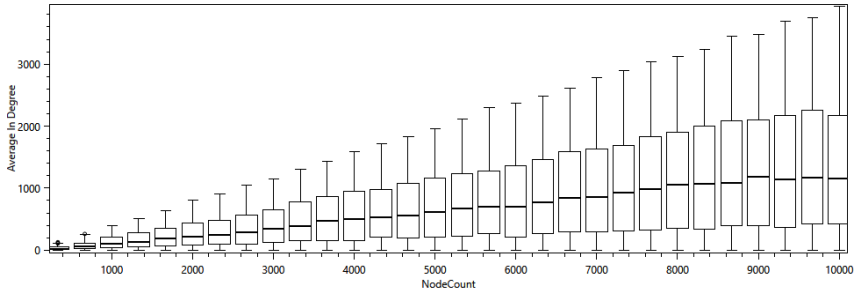
- Simulation von Kommunikationsgraphen
- 100 bis 10.000 Knoten
- $X, Y \in [0, 1]$
- Sendereichweiten im Schnitt von  $\frac{1}{200}$  bis  $\frac{1}{2}$
- Über 50.000 Simulationen
- Gleichverteilung von  $n$  und den Sendereichweiten



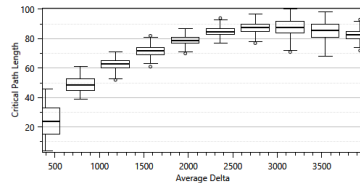
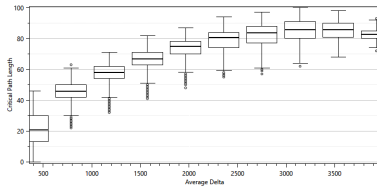
- $l$  steigt sublinear mit  $n$  an
- Aber:  $\Delta$  steigt linear mit  $n$  an und  $l$  steigt ebenfalls sublinear mit  $\Delta$  an
- Mit festem  $n$  verhält sich  $l$  gegenüber  $\Delta$  genauso wie mit allen  $n$
- $\Rightarrow$  Abhängigkeit zu  $n$  bedingt durch höheres  $\Delta$



- $f$  steigt sublinear mit  $n$  an
- Aber:  $\Delta$  steigt linear mit  $n$  an und  $f$  steigt ebenfalls sublinear mit  $\Delta$  an
- Mit festem  $n$  verhält sich  $f$  gegenüber  $\Delta$  genauso wie mit allen  $n$
- $\Rightarrow$  Abhängigkeit zu  $n$  bedingt durch höheres  $\Delta$

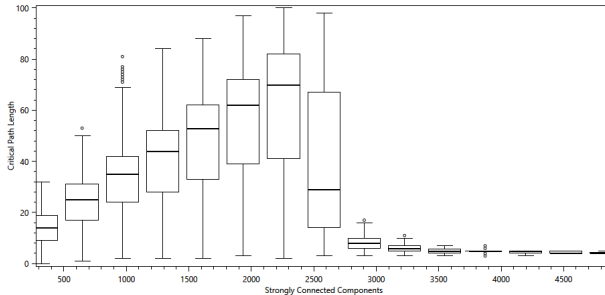


- $l$  steigt sublinear mit  $n$  an
- Aber:  $\Delta$  steigt linear mit  $n$  an und  $l$  steigt ebenfalls sublinear mit  $\Delta$  an
- Mit festem  $n$  verhält sich  $l$  gegenüber  $\Delta$  genauso wie mit allen  $n$
- $\Rightarrow$  Abhängigkeit zu  $n$  bedingt durch höheres  $\Delta$

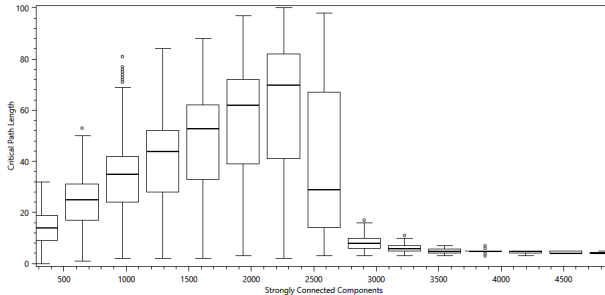


- $f$  steigt sublinear mit  $n$  an
- Aber:  $\Delta$  steigt linear mit  $n$  an und  $f$  steigt ebenfalls sublinear mit  $\Delta$  an
- Mit festem  $n$  verhält sich  $f$  gegenüber  $\Delta$  genauso wie mit allen  $n$
- $\Rightarrow$  Abhängigkeit zu  $n$  bedingt durch höheres  $\Delta$

- $\ell$  steigt mit der Sendereichweite (bis die Fläche abgedeckt ist)
- Eine höhere Sendereichweite bewirkt ebenfalls ein höheres  $\Delta$
- Im Experiment bewirkt die höhere durchschnittliche Sendereichweite eine höhere Varianz
- Interpretation:  $\ell$  steigt mit der Dichte des Graphen ( $\Delta$ ) sowie der Varianz der Sendereichweiten

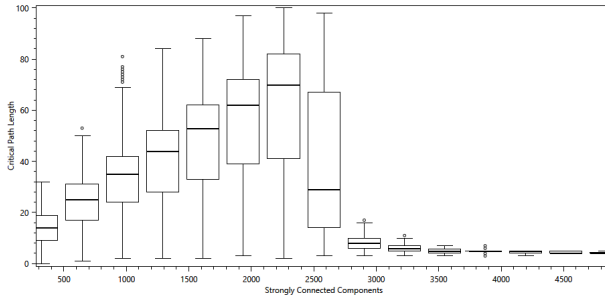


- $\ell$  steigt mit der Sendereichweite (bis die Fläche abgedeckt ist)
- Eine höhere Sendereichweite bewirkt ebenfalls ein höheres  $\Delta$
- Im Experiment bewirkt die höhere durchschnittliche Sendereichweite eine höhere Varianz
- Interpretation:  $\ell$  steigt mit der Dichte des Graphen ( $\Delta$ ) sowie der Varianz der Sendereichweiten

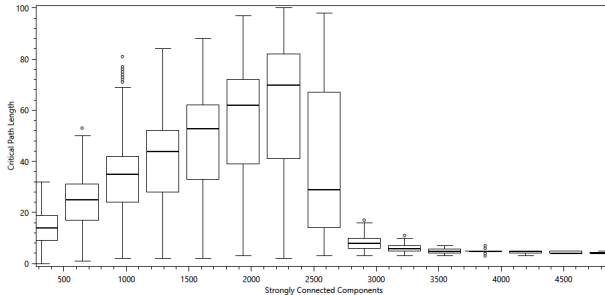




- $\ell$  steigt mit der Sendereichweite (bis die Fläche abgedeckt ist)
- Eine höhere Sendereichweite bewirkt ebenfalls ein höheres  $\Delta$
- Im Experiment bewirkt die höhere durchschnittliche Sendereichweite eine höhere Varianz
- Interpretation:  $\ell$  steigt mit der Dichte des Graphen ( $\Delta$ ) sowie der Varianz der Sendereichweiten



- $l$  steigt mit der Sendereichweite (bis die Fläche abgedeckt ist)
- Eine höhere Sendereichweite bewirkt ebenfalls ein höheres  $\Delta$
- Im Experiment bewirkt die höhere durchschnittliche Sendereichweite eine höhere Varianz
- Interpretation:  $l$  steigt mit der Dichte des Graphen ( $\Delta$ ) sowie der Varianz der Sendereichweiten



- Algorithmen können unidirektionale Kanten ohne viel Komplexität berücksichtigen
- Zwei Algorithmen vorgestellt:
  - $\Delta + 1$  Färbung in  $O(\Delta + l \log n)$  (mit Initialisierung) und
  - $2\Delta + 1$  Färbung in  $O(l \log n)$
- Mit den vorgestellten Algorithmen können realitätsnahe Szenarien abgedeckt werden
- Je größer die Sendereichweitenvarianz, um so größer wird  $l$  und somit die erwartete Laufzeit
- Ausblick: Differenz zwischen unteren Schranken und Laufzeiten verbessern

- Algorithmen können unidirektionale Kanten ohne viel Komplexität berücksichtigen
- Zwei Algorithmen vorgestellt:
  - $\Delta + 1$  Färbung in  $O(\Delta + l \log n)$  (mit Initialisierung) und
  - $2\Delta + 1$  Färbung in  $O(l \log n)$
- Mit den vorgestellten Algorithmen können realitätsnahe Szenarien abgedeckt werden
- Je größer die Sendereichweitenvarianz, um so größer wird  $l$  und somit die erwartete Laufzeit
- Ausblick: Differenz zwischen unteren Schranken und Laufzeiten verbessern

- Algorithmen können unidirektionale Kanten ohne viel Komplexität berücksichtigen
- Zwei Algorithmen vorgestellt:
  - $\Delta + 1$  Färbung in  $O(\Delta + l \log n)$  (mit Initialisierung) und
  - $2\Delta + 1$  Färbung in  $O(l \log n)$
- Mit den vorgestellten Algorithmen können realitätsnahe Szenarien abgedeckt werden
- Je größer die Sendereichweitenvarianz, um so größer wird  $l$  und somit die erwartete Laufzeit
- Ausblick: Differenz zwischen unteren Schranken und Laufzeiten verbessern

- Algorithmen können unidirektionale Kanten ohne viel Komplexität berücksichtigen
- Zwei Algorithmen vorgestellt:
  - $\Delta + 1$  Färbung in  $O(\Delta + l \log n)$  (mit Initialisierung) und
  - $2\Delta + 1$  Färbung in  $O(l \log n)$
- Mit den vorgestellten Algorithmen können realitätsnahe Szenarien abgedeckt werden
- Je größer die Sendereichweitenvarianz, um so größer wird  $l$  und somit die erwartete Laufzeit
- Ausblick: Differenz zwischen unteren Schranken und Laufzeiten verbessern

- Algorithmen können unidirektionale Kanten ohne viel Komplexität berücksichtigen
- Zwei Algorithmen vorgestellt:
  - $\Delta + 1$  Färbung in  $O(\Delta + l \log n)$  (mit Initialisierung) und
  - $2\Delta + 1$  Färbung in  $O(l \log n)$
- Mit den vorgestellten Algorithmen können realitätsnahe Szenarien abgedeckt werden
- Je größer die Sendereichweitenvarianz, um so größer wird  $l$  und somit die erwartete Laufzeit
- Ausblick: Differenz zwischen unteren Schranken und Laufzeiten verbessern