On the Complexity of Distributed Graph Coloring

Andreas Gemsa

Ausarbeitung zum Seminar Algorithmen für Sensornetze im SS 07

Institut für theoretische Informatik
Prof. Dr. Dorothea Wagner
Betreuer: Steffen Mecke

28. September 2007

Universität Karlsruhe (TH)

Inhaltsverzeichnis

1 Einleitung

Das Färbeproblem, also das Färben eines Graphen mit möglichst wenig Farben, so dass keine zwei benachbarten Knoten gleich gefärbt sind, ist schon im „normalen“ sequentiellen Fall NP-schwer. Da man für den verteilten Fall keine bessere Lösung erwarten kann, ist es nicht zu erwarten, dass man schnell eine minimale Lösung des Färbeproblems erreicht. Diese Ausarbeitung fokussiert sich auf die Betrachtung der theoretischen Leistungsfähigkeit von Algorithmen, die nur ein einziges mal ihre eigene Farbe den Nachbarknoten übermitteln und mit Hilfe der empfangenen Farben der Nachbarknoten eine neue Farbe wählen. Es geht hier also um Algorithmen, die einen sehr geringen Kommunikationsaufwand haben um eine Reduktion von Farben eines Graphen zu erreichen. Es wird weniger Wert auf die Angabe konkreter und auch praktikabler Verfahren gelegt, als auf theoretische Aspekte des verteilten Färben. Für ein besseres Verständnis wird im Verlauf ein spezieller Graph, der *Neighborhood Graph*, eingeführt mit dem man das Potential von bestimmten Färbe-Algorithmen beurteilen kann. Außerdem wird untersucht werden inwieweit eine iterative Anwendung von *One Round* Algorithmen die Anzahl der Farben verringert.
Im praktischen Bereich könnten die Farben Timeslots repräsentieren in denen drahtlose kommunizierende Sensoren Daten versenden dürfen. Bei einer gültigen Färbung senden dann auf keinen Fall direkt benachbarte Knoten Daten zur gleichen Zeit; so können direkte Kollisionen von über Funk übermittelte Daten vermieden werden.

2 Das Färbeproblem

2.1 Grundlagen

Zunächst folgen ein paar grundlegende Definitionen, die bei der Betrachtung des Färbeproblems relevant sind.

Definition 2.1 (*Gültige k-Färbung*)

Gegeben ein ungerichteter Graph $G = (V, E)$. Eine gültige k-Färbung ist eine Abbildung $\gamma : V \to \{1, \cdots, k\}$ mit $\forall (u, v) \in E : \gamma(u) \neq \gamma(v)$

Eine gültige Färbung weist also allen Knoten eines Graphen eine bestimmte Farbe zu, so dass keine adjazenten Knoten gleich eingefärbt sind. Für eine bessere Handhabung der Farben werden sie durch ganze Zahlen repräsentiert.

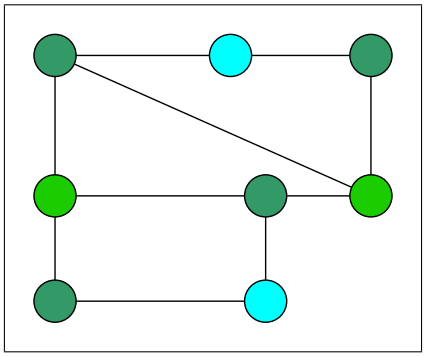

Abbildung 2.1: Gültige 3-Färbung

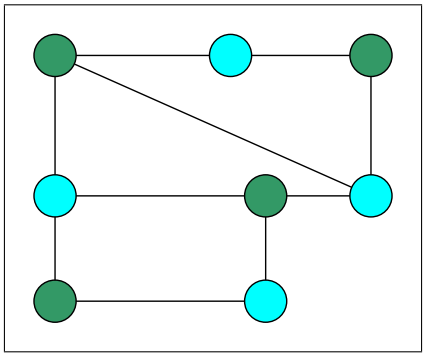

Abbildung 2.2: Gültige 2-Färbung

In Abbildung 2.1 und Abbildung 2.2 ist ein ungerichteter Graph G gegeben. Die Färbung von G in Abbildung 2.1 ist eine gültige 3-Färbung, da keine zwei adjazente Knoten gleich gefärbt sind und nur drei Farben verwendet werden. Dagegen ist in Abb. 2.2 eine gültige 2-Färbung des gleichen Graphen dargestellt. Diese Färbung stellt eine minimalgültige Färbung dar.

Die Suche nach einer minimalen gültigen Färbung wird eines Graphen wird als Färbeproblem bezeichnet.

Definition 2.2 (*Das Färbeproblem*)

Gegeben: Ein Graph $G = (V, E)$
Gesucht: Eine gültige k-Färbung, wobei k minimal ist.

Die Anzahl an Farben, die mindestens benötigt werden, um einen Graphen gültig zu färben wird als „chromatische Zahl“ bezeichnet.

Definition 2.3 (*Die chromatische Zahl χ*)

Die chromatische Zahl $\chi(G)$ eines Graphen G ist die kleinste Anzahl an verschiedenen Farben, die benötigt wird um den Graph gültig zu färben.

Bereits 1972 wurde von Karp gezeigt, dass das Finden der chromatischen Zahl eines Graphen G ein NP-schweres Problem darstellt [1]. Das heißt, falls NP $\neq$ P gilt, gibt es keinen deterministischen Algorithmus mit polynomialer Laufzeit der das Problem korrekt löst.

Außerdem kann eine obere Schranke für $\chi(G)$ für einen beliebigen Graphen G angegeben werden. Dazu wird folgende Definition benötigt.

Definition 2.4 (*Der maximale Grad eines Graphen Δ*)

Der maximale Grad, eines ungerichteten Graphen $G = (V, E)$ ist $\Delta := \max\{\text{Grad}(v)|v \in V\}$. Dabei ist der Grad eines Knoten die Anzahl der Kanten die diesen Knoten mit einem anderen verbinden.

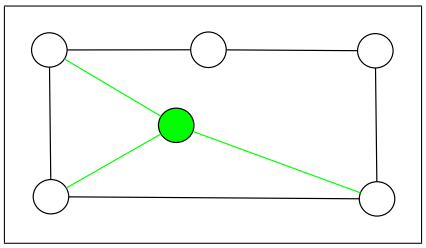

Abbildung 2.3:

Der grün markierte Knoten in Abbildung 2.3 ist durch die drei ebenfalls grün markierten Kanten mit anderen Knoten verbunden. Da kein Knoten mit mehr als drei weiteren Knoten verbunden ist, gilt für das gegebene Beispiel $\Delta = 3$.

Es kann gezeigt werden, dass $\chi(G) \leq \Delta + 1$ und es gibt Graphen mit $\chi(G) = \Delta + 1$. Ersteres gilt, denn durch einfaches sukzessives Färben mit der niedrigsten möglichen Farbe werden höchstens $\Delta + 1$ Farben benötigt. Letzteres gilt in vollständigen Graphen.

2.2 Alternative Betrachtungsweisen des Färbeproblems

2.2.1 Maximum Independent Set

Das Färbeproblen und die Suche nach einem *Maximum Independet Sets* sind zwei verwandte Probleme.

Definition 2.5 (*Independent Set*)

Ein Independet Sets ist eine Menge von Knoten eines Graphen $G = (V, E)$, bei der keine zwei Knoten aus dieser Menge miteinander verbunden sind.

Definition 2.6 (*Maximum Independent Set*)

Ein Maximum Independent Set ist ein *Independent Set* maximaler Größe.

Man kann leicht einsehen, dass die Menge der Knoten die in einer gültigen Färbung gleich gefärbt sind, ein *Independent Set* bilden.

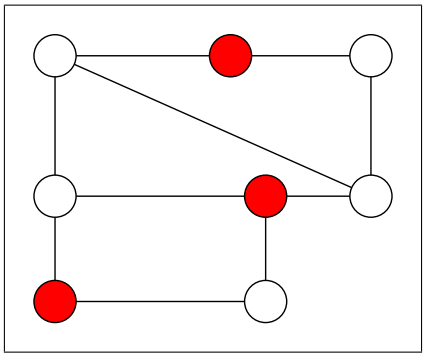

Abbildung 2.4: *Independet Set*

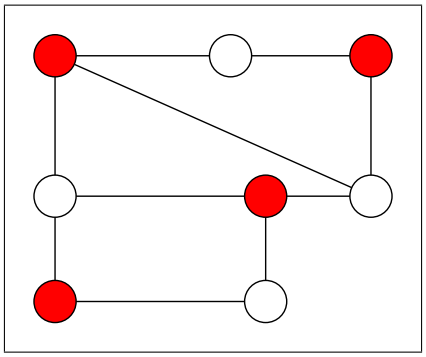

Abbildung 2.5: *Maximum Independet Sets*

Im folgenden wird gezeigt, wie man eine $\Delta + 1$ Färbung eines beliebigen Graphen $G = (V, E)$ mit Hilfe des MIS berechnen kann.

In den Zeilen 1-7 von Algorithmus 1 wird der Graph G' konstruiert. Dabei sind in G' jeweils alle Kopien eines Knoten zu einer Clique verbunden. Zwei Knoten u_i, v_i sind in G' verbunden, wenn $(u, v) \in E$ und $i = j$ gilt. Für jeden Knoten v des Graphen G kann maximal nur ein v_i in einem MIS von G' enthalten sein, da alle v_i untereinander verbunden sind. Von jedem Knoten v werden $\Delta + 1$ Kopien erzeugt und jede Kopie ist mit maximal Δ Knoten aus jeweils anderen Cliquen verbunden. Da in jeder Clique höchstens ein Knoten zum MIS gehört sind höchstens Δ Knoten, der insgesamt $\Delta + 1$ Knoten einer Clique, keine Kandidaten für das MIS. Es findet sich also immer wenigstens ein Kandidat für ein MIS. Da ein MIS ein maximales *Independent Set* ist, beinhaltet es mindestens einen Knoten aus jeder Clique. Insgesamt heißt das, dass genau ein Knoten aus jeder Clique im MIS enthalten ist. Deshalb kann der Knoten v die Farbe i annehmen. Zwei benachbarte Knoten u und v aus G können so nicht gleich gefärbt werden, da u_i

Algorithmus 1 : MIS to $\Delta + 1$ coloring

Eingabe : $G = (V, E)$

Erzeuge leeren Graph G' = (V', E')
forall $v \in V$ **do**
    Erstelle $\Delta + 1$ Kopien von jedem Knoten $v \in V$ $(v_0, \cdots, v_\Delta)$
    Verbinde alle Knoten v_i zu einer Clique
forall $(u, v) \in E$ **do**
    for $i = 0, \cdots, \Delta$ **do**
        Füge (u_i, v_i) E' hinzu
Berechne MIS von G'
forall $v \in V$ **do**
    $i \leftarrow$ das j für das gilt v_j ist im MIS enthalten
    Wähle für den Knoten v die Farbe i

und v_i für alle $i \in \{1, \cdots, \Delta\}$ in G' durch eine Kante verbunden sind und damit können sie nicht beide im selben MIS enthalten sein.

2.2.2 Minimum Set Cover

Es ist möglich die Suche nach einer minimalen Färbung eines Graphen in eine Instanz von *Minimum Set Cover* zu überführen. Dazu zunächst eine Erklärung was das *Minimum Set Cover* Problem ist.

Definition 2.7 (*Minimum Set Cover*)

Sei ein Tupel (X, F) gegeben, wobei X eine endliche Menge und F eine Familie von Teilmengen von X ist. Gesucht ist eine Teilmenge $C \subseteq F$, so dass jedes Element aus X in C enthalten ist. Zudem soll die Teilmenge C möglichst klein sein.

Die Umwandlung des Färbeproblems in eine Instanz von *Minimum Set Cover* funktioniert folgendermaßen. Man verwendet als Menge X die Menge aller Knoten des Graphen $G = (V, E)$ für den man eine minimale Färbung sucht. Die Menge F besteht aus allen *Independent Sets* des Graphen G. Findet man eine minimale Teilmenge an *Independent Sets* so erhält man automatisch eine gültige minimale Färbung des Graphen G, wenn man jedem *Indepent Set* eine Farbe zuweist. Alle Knoten in diesem *Independent Set* werden mit der zugewiesenen Farbe eingefärbt. Sollte sich ein Knoten in mehreren *Independent Sets* befinden, so wird der Knoten mit einer beliebigen Farbe, der *Independet Sets* gefärbt, in denen er enthalten ist.

3 Das Färbeproblem bei verteilten Systemen

Um verteilte Systeme besser betrachten zu können, wird das *Standard Message Passing Model* eingeführt.

3.1 Standard Message Passing Model

Beim *Standard Message Passing Model* wird jeder Sensor (z.B. von einem ad-hoc Netzwerks) als Knoten in einem Graphen $G = (V, E)$ dargestellt. Benachbarte Sensoren, also Prozessoren die direkt miteinander kommunizieren können, werden in G durch Kanten verbunden. Pro Kommunikationsrunde kann jeder Knoten mit allen direkten Nachbarn beliebig viele Daten austauschen. Es werden beim *Standard Message Passing Model* keine Beschränkungen bezüglich der lokalen Berechnungen gegeben. Außerdem wird angenommen, dass alle Sensoren synchron arbeiten, also insbesondere den Algorithmus zur gleichen Zeit starten.

3.2 One Round Algorithmen

One Round Algorithmen sind Algorithmen, die nur eine „Kommunikations-Runde“ lang laufen. Das heißt, dass alle Knoten, nach dem *Standard Message Passing Model* , nur ein einziges mal mit ihren direkten Nachbarn kommunizieren. Bei *One Round* Algorithmen für das verteilte Färben wird im Folgenden angenommen, dass bereits vor der Ausführung des *One Round* -Färbealgorithmus, alle Knoten bereits gültig gefärbt sind.

Es wird im allgemeinen Angenommen, dass alle Knoten einen einzigartigen *Identifier* haben der als „Farbe“ benutzt wird. Solch ein Fall wäre eine gültige Färbung, bei der keine zwei Knoten gleich gefärbt sind. Es wird untersucht wie viele Farben mindestens benötigt werden um einen Graph mit einem *One Round* Algorithmen gültig zu färben.

Der Neighborhood Graph

Wie bereits erwähnt, tauschen die Knoten bei *One Round* Algorithmen ihre eigene Farbe mit den Nachbarknoten aus. In Abbildung 3.1 sehen wir einen Ausschnitt eines Graphens. Die Zahl in einem Knoten entspricht der Farbe mit der der Knoten gefärbt wurde. Der

mittlere Knoten bildet nach dem Austauschen der Farben das Tupel $(1, \{2, 3, 4, 5\})$. Der Knoten rechts oben bildet dagegen $(3, \{1, \cdots, \})$. Allein auf Grundlage solch eines Tupels trifft jeder Knoten die Entscheidung welche Farbe er annimmt. Im Folgenden wird das

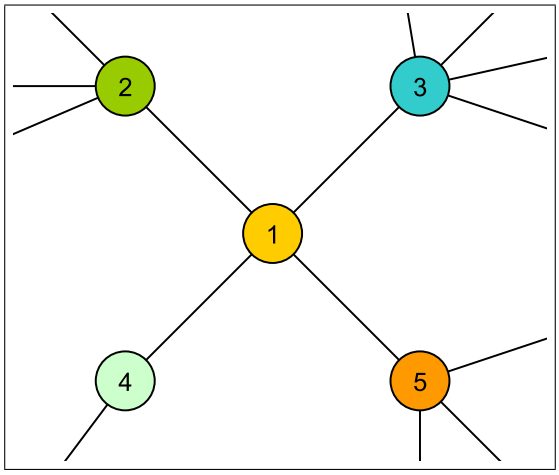

Abbildung 3.1: Ausschnitt eines Beispielsgraphens

Konzept des *Neighborhood Graph* eingeführt, dass auf [3] beruht, aber in [2] für allgemeine Graphen weiterentwickelt wurde. Die Idee hinter dem *Neighborhood Graph* ist die, dass man aus allen möglichen Tupeln (x, Γ_x) einen Graphen konstruiert. Dieser Graph, *Neighborhood Graph* genannt, lässt dann Rückschlüsse auf die Leistungsfähigkeit von *One Round* Algorithmen zu.

Definition 3.1 (*Neighborhood Graph*)

Seien $\Delta \geq 1$ *und* $m > \Delta$ *gegeben. Die Knoten des Neighborhood Graph* $N(m, \Delta)$ bestehen aus allen Paaren (x, Γ_x), für die gilt, dass $x \in \{1, \cdots, m\}$, $\Gamma_x \subset \{1, \cdots, m\}$, $\Gamma_x = \Delta$ und $x \notin \Gamma_x$. Zwei Knoten $(x, \Gamma_x), (u, \Gamma_u)$ sind genau dann durch eine Kante im *Neighborhood Graph* verbunden, wenn $x \in \Gamma_u$ und $u \in \Gamma_x$.

In Abbildung 3.2 wird der Graph $N(4, 2)$ ohne Kanten dargestellt. Im Anhang befindet sich derselbe Graph mit Kanten und einer veränderten Positionierung der Knoten.

Bemerkenswert bei dieser Definition des *Neighborhood Graph* ist, dass das die Konstruktion eines *Neighborhood Graph* nur noch indirekt von einem Graphen $G = (V_G, E_G)$ abhängt. Nur der maximale Grad von G sowie die Anzahl der Farben der ursprünglichen gültigen m-Färbung ist für den *Neighborhood Graph* entscheidend, aber nicht mehr die exakte Topologie. Es soll nun gezeigt werden, dass es ausreicht $\chi(N(m, \Delta))$ zu betrachten um die Leistungsfähigkeit von *One Round* Algorithmen zu beurteilen. Dazu wird zunächst gezeigt, dass es einen *One Round* -Algorithms gibt, mit dem man jeden gültig m-gefärbten Graph mit $q = \chi(N(m, \Delta))$ Farben färben kann. Danach wird gezeigt, dass es wenigstens einen gültig m-gefärbten Graph gibt, der sich nicht mit weniger als q Farben in einer Runde färben lässt.

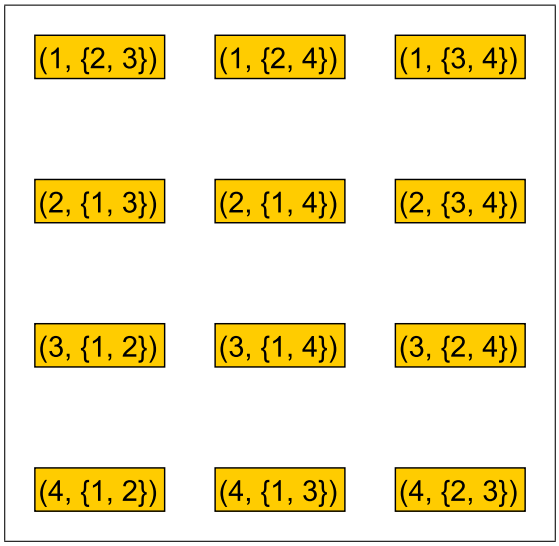

Abbildung 3.2: N(4, 2) ohne eingezeichnete Kanten

Lemma 3.1

Es gibt einen One Round Algorithmus, der einen beliebigen gültig m-gefärbten Graphen $G = (V_G, E_G)$ mit maximalem Grad Δ, in einer Runde mit höchstens $q = \chi(N(m, \Delta))$ Farben färbt.

Zum Beweis des Lemmas wird ein Verfahren angegeben mit dem man einen gültig m-gefärbten Graph in einen gültig q-gefärbten Graph überführen kann.

Algorithmus 2 : Farbwahl

Eingabe : Δ, m

1 Sende eigen Farbe an alle Nachbarn
2 Speichere alle empfangenen Farben der Nachbarn in (x, Γ_x)
3 Berechne *Neighborhood Graph* $N(m, \Delta)$
4 Berechne minimale gültige Färbung von $N(m, \Delta)$
5 **if** $\Gamma_x < \Delta$ **then**
6 fülle Γ_x zufällig mit Farben aus $\{1, \cdots, m\} \setminus \{x\}$ auf, bis $\Gamma_x = \Delta$
7 Wähle als Knotenfarbe die Farbe des Knoten im *Neighborhood Graph* (x, Γ_x)

Beweis: In den ersten beiden Schritten tauschen die Knoten die eigene Farbe mit ihren jeweiligen Nachbarknoten aus. Der *Neighborhood Graph* kann von jedem Knoten berechnet werden, sofern m und Δ zur Verfügung stehen, was hier angenommen wird. Entscheidend sind die Schritte in den Zeilen 5-7. Sollte $|\Gamma_x|$ in einem Knoten kleiner als Δ sein, dann werden per Zufall Farben aus $\{1, \cdots, m\} \setminus \{x\}$ ausgewählt und zu Γ_x hinzugefügt, bis $|\Gamma_x|$ den Wert Δ erreicht hat. Dieser Fall tritt dann ein, wenn zwei Nachbarknoten gleich gefärbt sind, oder wenn der Knoten v gar nicht Grad Δ hat. Zum Schluss wählt der Algorithmus für

den Knoten die Farbe des Knotens (x, Γ_x) des bereits minimal gültig gefärbten *Neighborhood Graph* aus.

Zu zeigen bleibt, dass die erhaltene q-Färbung von G auch gültig ist.
Seien u und v zwei in G benachbarte Knoten und y und x die Farben der beiden Knoten u und v bevor Algorithmus 1 angewandt wurde. Die beiden Farben x und y müssen verschieden sein, da zu der Graph G zunächst gültig m-gefärbt war. Nach Anwendung der zweiten Zeile des Algorithmus 1 hat der Knoten u das Tupel (y, Γ_y) mit $x \in \Gamma_y$ und der Knoten v entsprechend (x, Γ_x) mit $y \in \Gamma_x$ gebildet. Das bedeutet aber, dass die Knoten (y, Γ_y) und (x, Γ_x) im *Neighborhood Graph* durch eine Kante verbunden sind. Das heißt, dass nachdem beide Knoten durch die in Schritt 4 berechne minimale gültige Färbung, gefärbt wurde, beide Knoten unterschiedliche Farben besitzen. Da u bzw. v die Farben von (y, Γ_y) bzw. (x, Γ_x) übernehmen, können sie nicht gleich gefärbt sein.

Es folgt jetzt, dass sich jeder gültig m-gefärbte Graph mit maximalem Grad Δ mit maximal $q = \chi(N(m, \Delta))$ Farben Färben lässt. ■

Lemma 3.2

Sei $\Diamond$ die Menge aller gültig m-gefärbten Graphen mit maximalem Grad Δ. Für jeden One-Round Algorithmus existiert ein Graph in $\Diamond$, der sich nicht mit weniger als $q = \chi(N(m, \Delta))$ Farben färben lässt.

Beweis : Sei ein Anzahl an Farben $t < q$ gegeben. Jeder *One-Round* Algorithmus weist jedem Knoten u, abhängig von der Menge der Farben der Nachbarknoten Γ_u, eine Farbe zu. Dies entspricht einer Abbildung von den Tupeln (x, Γ_x) in die Menge der Farben $\{1, \cdots, t\}$. Man kann sie auch als Funktion γ auffassen, die aus der Menge der Knoten eines *Neighborhood Graph* in die Menge der Farben abbildet. Wenn man den *Neighborhood Graph* mit Hilfe der Funktion γ mit t Farben färben will, so werden mindestens zwei benachbarte Knoten (x, Γ_x), (y, Γ_y) gleich gefärbt. Es lässt sich ein Graph in $\Diamond$ finden mit Knoten u und Farbe x, dessen Nachbarn die Farben Γ_x haben und der verbunden ist zu einem Knoten v mit Farbe y mit Nachbarn die mit den Farben aus Γ_y gefärbt sind. Der zu der Funktion γ gehörende Algorithmus wird die benachbarten Knoten u und v gleich färben. Es entsteht keine gültige Färbung. ■

Es wurde gezeigt, dass es einen Algorithmus gibt, der einen beliebigen Graphen mit Hilfe des *Neighborhood Graph* mit maximal $q = \chi(N(m, \Delta))$ Farben färben kann. Und es wurde gezeigt, dass es zu jedem Algorithmus einen gültig m-gefärbten Graph mit maximalem Grad Δ gibt der sich nicht mit weniger als q Farben gültig färben lässt. Das heißt, dass es ausreicht $\chi(N(m, \Delta))$ zu betrachten, wenn man die Leistungsfähigkeit von *One Round* Algorithmen beurteilen will.
Die Anzahl der Knoten eines *Neighborhood Graph* ergibt sich nach dem folgenden Lemma.

Lemma 3.3

Die Anzahl Ψ der Knoten in einem Neighborhood Graph entspricht

$$\begin{pmatrix} m-1 \\ \Delta \end{pmatrix} \cdot m = \frac{m!}{\Delta! \cdot (m-1-\Delta)!}$$

Beweis: Sei eine Menge $F := \{1, \cdots, m\}$ gegeben. Der *Neighborhood Graph* $N(m, \Delta)$ besteht gemäß Definition aus allen Paaren (x, Γ_x), für die gilt, dass $x \in \{1, \cdots, m\}$, $\Gamma_x \subset \{1, \cdots, m\}$, $\Gamma_x = \Delta$ und $x \notin \Gamma_x$. Das heißt, dass die Menge der Knoten in $N(m, \Delta)$ dem Produkt, aus der Anzahl aller Möglichkeiten Γ_x zu bilden und der Anzahl der verschiedenen Möglichkeiten x zu wählen, entspricht.

$$\begin{pmatrix} m-1 \\ \Delta \end{pmatrix} \cdot m = \frac{(m-1)!}{\Delta! \cdot (m-1-\Delta)!} \cdot m = \frac{m!}{\Delta! \cdot (m-1-\Delta)!}$$

■

3.3 Deterministische One-Round Algorithms

Zunächst wird eine obere Schranke für die maximale Farbanzahl die benötigt wird um einen Graphen in einer Runde gültig zu färben. Dabei ist die obere Schranke eine Verbesserung der besten bekannten aus [3], allerdings unterscheiden sich beide nur um einen konstanten Faktor.

3.3.1 Obere Schranke

Für die Untersuchung der oberen Schranke benötigen wir die Defintion des *fractional covering* und der *fractional chromatic number* .

Definition 3.2 (*fractional covering*)

Weise jedem Independet Set S aus einem Graphen G ein Gewicht x_S zu, so dass für jeden Knoten gilt, dass die Summe aller Gewichte aller *Independent Sets*, in denen dieser Knoten enthalten ist, mindestens 1 ist.

Definition 3.3 (*fractional chromatic number*)

Das kleinst mögliche fractional covering wird als *fractional chromatic number* bezeichnet.

Es wird in [2] gezeigt, dass jeder Knoten aus $N(m, \Delta)$ in $m!/(\Delta + 1)$ verschiedenen *Independent Sets* der Größe $\Psi/(\Delta + 1)$. Dabei ist Ψ die Anzahl der Knoten in einem *Neighborhood Graph* (s. Lemma 3.3). Außerdem wird in [2] gezeigt, dass es in einem *Neighborhood Graph* genau $m!$ *Maximum Independent Sets* gibt.

Lemma 3.4

Wenn $m > \Delta$, dann hat die fractional chromatic number $\chi_f(N(m, \Delta))$ des Graphen $N(m, \Delta)$ den Wert $\Delta + 1$.

Beweis: Um ein *fractional covering* der Knoten zu erhalten muss für jeden Knoten v aus dem *Neighborhood Graph* allen *Independent Sets* ein Gewicht x_s zuweisen, so dass die Summe der Gewichte aller *Independent Sets* in denen v enthalten ist mindestens 1 ist. Damit das erreicht wird, wird jedem der $m!$ verschiedenen *Independent Sets* das Gewicht $x_s = (\Delta + 1)/m!$ zugewiesen. Das Erhaltene *fractional covering* hat die Größe $m! \cdot (\Delta + 1)/m! = \Delta + 1$. Da

jeder Knoten in $K/(\Delta+1)$ *Independent Sets* enthalten ist, beträgt die Summe der Gewichte der *Independent Sets* in denen ein Knoten enthalten, ist immer genau 1. Damit wurde ein minimales *fractional covering* erreicht. ■

Lemma 3.5

Für alle $m > \Delta$ ist $\chi(N(m,\Delta)) \leq (\Delta+1)^2(\ln m + 1)$

Beweis: Die chromatische Zahl eines Graphen G entspricht der Anzahl der *Independent Sets* die mindestens benötigt werden damit jeder Knoten aus G wenigstens in einem dieser *Independent Sets* enthalten ist (vgl. Abschnitt 2.2.2). Also ist $\chi(G)$ die Lösung einer Instanz des Problems *Minimum Set Cover* . Die beste Lösung eines *fractional covering* und der besten ganzzahligen Lösung unterscheiden sich höchstens um den Faktor $\ln(\alpha(G))+1$, wobei $\alpha(G)$ die Größe des größten *Independet Sets* von G ist [2]. Also gilt: $\chi(G) \leq (\ln(\alpha(G))+1)\chi_f(G)$. Die Anzahl der Knoten in dem *Neighborhood Graph* $N(m,\Delta)$ entspricht $m!/(\Delta!(m-1-\Delta)!)$. Da das *Maximum Independet Sets* von $N(m,\Delta) = \Psi/(\Delta+1)$, folgt für $\alpha(N(m,\Delta))$:

$$\begin{aligned}
\alpha(N(m,\Delta)) &= \frac{m!}{\Delta! \cdot (m-\Delta-1)!} \cdot \frac{1}{\Delta+1} \\
&= \frac{m!}{(\Delta+1)! \cdot (m-\Delta-1)!} \\
&= \frac{m!}{(\Delta+1)! \cdot (m-(\Delta+1)!)} \\
&= \begin{pmatrix} m \\ \Delta+1 \end{pmatrix} < m^{\Delta+1}
\end{aligned}$$

Also folgt für $\chi(N(m,\Delta))$:

$$\begin{aligned}
\chi(N(m,\Delta)) &\leq (\ln(\alpha(N(m,\Delta))) + 1) \cdot \chi_f(N(m,\Delta)) \\
&\leq (\ln(m^{\Delta+1}) + 1) \cdot (\Delta+1) \\
&= ((\Delta+1)\ln(m) + 1)(\Delta+1) = (\Delta+1)^2(\ln(m) + 1/(\Delta+1)) \\
&\leq (\Delta+1)^2(\ln(m)+1)
\end{aligned}$$

Damit folgt die Behauptung. ■

Die Autoren von [2] haben sich dazu entschlossen, die etwas größere obere Schranke $(\Delta+1)^2(\ln(m)+1)$ zu nutzen. Schon $(\Delta+1)^2(\ln(m)+1/(\Delta+1))$ ist eine gültige Abschätzung. Die Verschlechterung der oberen Grenze aus [2] ist jedoch nur geringfügig.

3.3.2 Untere Schranke

Die grundlegende Idee für den Beweis der unteren Schranke für die Anzahl der Farben, die von einem *One Round* Algorithmus mindestens benötigt werden um einen *Neighborhood Graph* zu färben, beruht auf *Independent Sets* . Sind q verschiedene *Independent Sets* $(S_1, \cdots, S_q)$ von $N(m,\Delta)$ gegeben und jeder Knoten ist in wenigstens einem der S_i enthalten, dann ist die chromatische Zahl von $N(m,\Delta)$ höchstens q. Was gezeigt werden

soll ist, wenn q klein genug ist, dann ist wenigstens ein Knoten in keinem *Independet Set* S_i vertreten, egal wie wir die *Independent Sets* wählen. Dazu definieren wir zu einer Farbe x folgendes: $\Phi_S(x) := \{y|\ y \notin \Gamma_u\ \forall(u,\Gamma_u) \in S\}$, sowie $\phi_S(x) := |\Phi_S(x)|$. Dabei bezeichnet $\Phi_S(x)$ die Menge der Farben, die nicht in Γ_u für alle $(u,\Gamma_u) \in S$, auftreten und $\phi_S(x)$ bezeichnet die Anzahl dieser Farben. Wir wollen zeigen, dass für ein q, das klein genug ist, es $\Delta + 1$ Farben $x, y_1, \cdots, y_\Delta$ gibt, so dass für eine beliebige Wahl von den *Independent Sets* $S_1, \cdots, S_q$ gilt:

$$\forall i \in \{1, \cdots, q\},\ \exists j \in \{1, \cdots, \Delta\} : y_j \in \Phi_{S_i}(x)$$

Daraus folgt, dass der Knoten $(x, \{y_1, \cdots, y_\Delta\} \setminus \{y_j\})$ in keinem *Independet Set* enthalten ist, also auch nicht mit einer der q Farben gefärbt wird. Also bilden diese q *Independent Sets* keine gültige Färbung für $N(m, \Delta)$.
Die Suche nach der Farbe y_j lässt sich auch als Instanz des *Minimum Set Cover*-Problems betrachten. Dabei sind die Farben $\{1, \cdots, m\} \setminus \{x\}$ die Mengen und die *Independent Sets* $(S_1, \cdots, S_q)$ sind die Elemente. Ein Element S_i sei gemäß dem *Minimum Set Cover*-Problems in einer Menge y_j enthalten, wenn $y_j \in \Phi_{S_i}$ gilt.
Für die folgenden Lemmata ist es notwendig, alle $\phi_{S_i}(x)$, $i \in \{1, \cdots q\}$ für alle Farben x nach ihrem Wert aufsteigend zu sortieren. Der j-te Wert dieser Sortierung wird bezeichnet als $h_j(x)$.

Lemma 3.6

Sei ein ganzzahliges t gegeben. Wenn $t((m-q)t - q) > 2q(m-1)$, dann existiert ein $x \in \{1, \cdots, m\}$ für das gilt:

$$\forall i \in \{1, \cdots, t\} : h_i(x) > 0$$

Beweis s. [2]

Lemma 3.7

Sei ein ganzzahliges t gegeben. Wenn $t((m-q)t - q) > 2q(m-1)$ gilt, dann gibt es eine Farbe $x \in \{1, \cdots, m\}$, für die das beschriebene Minimum Set Cover -Problem eine fractional covering Lösung hat, die höchstens t ist.

Beweis : Es muss jeder Farbe $y \in \{1, \cdots, m\} \setminus \{x\}$ ein Wert λ_y zugewiesen werden, so dass:

$$\forall i \in \{1, \cdots, q\} : \sum_{y \in \Phi_{S_i}} \lambda_y \geq 1$$

Sei $\lambda_y := 1/(\min\{\phi_{S_i}(x)|\ y \in \Phi_{S_i}(x)\})$.
Es haben genau $h_1(x)$ Farben y den Wert $\lambda_y = 1/h_1(x)$. Das folgt direkt aus der Definition von h_i. Denn $h_1(x)$ ist die kleinste Anzahl an Farben die nicht in Γ_x für alle $(u,\Gamma_u) \in S_j$ für ein $j \in \{1, \cdots, q\}$ auftreten. Analog kann man argumentieren, dass maximal $h_i(x)$ Farben den Wert $1/h_i(x)$ annehmen können, für alle $i > 1$.
Also ist die Summe aller λ_y mindestens 1 und es gilt folgdens:

$$\sum_{y \in \Phi_{S_i}} \lambda_y \leq \sum_{i=1}^{t} h_i(x) \frac{1}{h_i(x)} = \sum_{i=1}^{t} 1 = t$$

■

Nun wird eine untere Schranke für $\chi(N(m,\Delta)) > q$ für gewisse q ermittelt, mit der dann letztendlich auf eine untere Schranke in Abhängigkeit von Δ geschlossen wird.

Lemma 3.8

Es gilt $\chi(N(m,\Delta)) > q$ genau dann, wenn

$$\Delta\left(\Delta - \frac{q\ln(eq)}{m-q}\right) > \frac{2(m-1)q(\ln(eq)^2)}{m-q} \tag{3.1}$$

Beweis: Es ist bekannt, dass die beste ganzzahlige Lösung für eine Instanz des *Minimum Set Cover* Problems höchstens um den Faktor $\ln(\alpha(G))+1$ größer ist, als die beste *fractional covering* Lösung. Wobei $\alpha(G)$ die größe des größten *Independent Sets* von G darstellt. Das größtmögliche *Independet Set* des beschriebenen *Minimum Set Cover* Problems kann höchstens den Wert q annehmen. Es wurde gezeigt: Wenn $t((m-q)t-q) > 2q(m-1)$ gilt, dann gibt es eine Farbe, für die die *fractional covering* Lösung des *Minimum Set Cover* Problems höchstens t ist. Außerdem wurde beschrieben, dass wenn die Lösung des *Minimum Set Cover* den Wert Δ hat, es wenigstens einen Knoten des *Neighborhood Graph* gibt, der in keinem der q *Independent Sets* enthalten ist und damit nicht gefärbt wird. Also folgt insgesamt, dass $\chi(N(m,\Delta)) > q$ gilt, wenn

$$\Delta \geq t\cdot(\ln(q)+1)\ ,\ t((m-q)t-q) > 2q(m-1)$$

$$\Delta \geq t\cdot(\ln(q)+1) = t\ln(eq) \Rightarrow \frac{\Delta}{\ln(eq)} \geq t$$

Ersetzt man nun t durch $\Delta/\ln(eq)$ dann erhält man

$$\begin{aligned}
&\frac{\Delta}{\ln(eq)}\left((m-q)\frac{\Delta}{\ln(eq)} - q\right) \geq t((m-q)t-q) > 2q(m-1)\\
&\Rightarrow \frac{\Delta}{\ln(eq)}\left((m-q)\frac{\Delta}{\ln(eq)} - q\right) > 2q(m-1)\\
&\Rightarrow (m-q)\left(\frac{\Delta}{\ln(eq)}\right)^2 - q\frac{\Delta}{\ln(eq)} > 2q(m-1)\\
&\Rightarrow (m-q)\Delta^2 - q\Delta\ln(eq) > 2q(m-1)(\ln(eq))^2\\
&\Rightarrow \Delta^2 - \frac{q\Delta\ln(eq)}{m-q} > \frac{2q(m-1)(\ln(eq))^2}{m-q}\\
&\Rightarrow \Delta\left(\Delta - \frac{q\ln(eq)}{m-q}\right) > \frac{2q(m-1)(\ln(eq))^2}{m-q}
\end{aligned}$$

■

Mit Hilfe des Lemmas 3.8 ist es möglich eine untere Schranke für $\chi(N(m,\Delta))$ anzugeben, wobei diese Schranke nur noch von Δ abhängt.

Lemma 3.9

Wenn $m \in \Omega(\Delta^2/\log\Delta)$, dann ist die chromatische Zahl für den Neighborhood Graph

$$\chi(N(m,\Delta)) \in \Omega\left(\frac{\Delta^2}{(\log\Delta)^2}\right)$$

Es gilt zu zeigen, dass die Ungleichung (3.1) für ein $q \in \Omega(\Delta^2/(\log \Delta)^2)$ gilt, wenn $m \in \Omega(\Delta^2/\log \Delta)$.

Beweis: Sei $m = d\Delta^2/\ln \Delta$ und $q = c\Delta^2/(\ln \Delta)^2$, wobei d und c eine Konstanten sind. Gesucht ist ein passendes c, so dass Ungleichung (3.1) gilt. Zunächst kann für $m - q$ eine Abschätzung angegeben werden, die die späteren Umformungen erleichtert. Für $c < d$:

$$\begin{aligned} m - q &= \frac{d\Delta^2}{\ln \Delta} - \frac{c\Delta^2}{(\ln \Delta)^2} = \frac{\Delta^2}{\ln \Delta}\left(d - \frac{c}{\ln \Delta}\right) \\ &= \frac{\Delta^2}{\ln \Delta} \cdot \frac{d \ln \Delta - c}{\ln \Delta} \leq \frac{\Delta^2}{\ln \Delta} \cdot \frac{\ln \Delta - 1}{\ln \Delta} \end{aligned}$$

Also gilt für die linke Seite der Ungleichung (3.1):

$$\Delta\left(\Delta - \frac{q \ln(eq)}{m-q}\right) \geq \Delta\left(\Delta - \frac{q \ln(eq)}{\frac{\Delta^2}{\ln \Delta} \cdot \frac{\ln \Delta - 1}{\ln \Delta}}\right) = \Delta\left(\Delta - \frac{q \ln(eq)(\ln \Delta)^2}{\Delta^2(\ln \Delta - 1)}\right)$$

Setzt man nun $q = c\Delta^2/(\ln \Delta)^2$ in die obige Gleichung ein erhält man:

$$\begin{aligned} &\Delta\left(\Delta - \frac{c\Delta^2}{(\ln \Delta)^2} \cdot \frac{\ln\left(e^{\frac{c\Delta^2}{(\ln \Delta)^2}}\right)(\ln \Delta)^2}{\Delta^2(\ln \Delta - 1)}\right) \\ &= \Delta\left(\Delta - c \cdot \frac{\ln\left(e^{\frac{c\Delta^2}{(\ln \Delta)^2}}\right)}{\ln \Delta - 1}\right) \\ &= \Delta\left(\Delta - c \cdot \frac{\ln(ec) + 2\ln \Delta - (\ln \Delta)^2}{\ln \Delta - 1}\right) \\ &\geq \Delta(\Delta - 1) \end{aligned}$$

Für ein entsprechend kleines c gilt also:

$$\Delta\left(\Delta - \frac{q \ln(eq)}{m-q}\right) \geq \Delta(\Delta - 1) \tag{3.2}$$

Außerdem kann man eine Konstante c' finden damit

$$\begin{aligned} \frac{m-1}{m-q} &\leq \frac{(m-1) \cdot (\ln \Delta)^2}{\Delta^2(\ln \Delta - 1)} \\ &= \frac{d\Delta^2 \cdot (\ln \Delta)^2 - 1}{\ln \Delta \cdot \Delta^2(\ln \Delta - 1)} = \frac{d \ln \Delta - 1}{\ln \Delta - 1} \leq c' \end{aligned}$$

gilt.
Für die rechte Seite der Ungleichung (3.1) gilt:

$$\frac{2(m-1)q(\ln(eq))^2}{m-q} = \frac{m-1}{m-q} \cdot 2q(\ln(eq))^2 \leq c' \cdot 2q(\ln(eq))^2$$

Ersetzt man nun q mit $c\Delta^2/(\ln \Delta)^2$ dann folgt:

$$c' \cdot 2q(\ln(eq))^2 = 2c' \cdot \frac{c\Delta^2}{(\ln \Delta)^2}\left(\ln\left(e^{\frac{c\Delta^2}{(\ln \Delta)^2}}\right)\right)^2 = \frac{2c'c\Delta^2(\ln(ec) + \ln \Delta - (\ln \Delta)^2)}{(\ln \Delta)^2}$$

Für ein c das klein genug ist, gilt dann:

$$\frac{2c'c\Delta^2(\ln(ec) + \ln\Delta - (\ln\Delta)^2)}{(\ln\Delta)^2} < \Delta(\Delta - 1)$$

Also gilt:

$$\frac{2(m-1)q(\ln(eq))^2}{m-q} < \Delta(\Delta - 1) \tag{3.3}$$

Kombiniert man die Gleichungen (3.2) und (3.3) zeigt sich, dass Gleichung (3.1) für ein passendes c erfüllt ist, wenn $m = d\Delta^2/\ln\Delta$ und $q = c\Delta^2/(\ln\Delta)^2$. Nach Lemma 3.8 folgt $\chi(N(m,\Delta)) > q = c\Delta^2/(\ln\Delta) \in \Omega(\Delta^2/(\ln\Delta))$. Also wurde Lemma 3.9 und damit eine untere Schranke für $\chi(N(m,\Delta))$ gezeigt. ■

3.4 Iterative Anwendung von One Round Algorithmen

Will man in einer verteilten Umgebung eine Färbung eines Graphen mit möglichst wenig Farben erzeugen, so kann man iterativ *One Round* Algorithmen auszuführen, die pro Runde die Anzahl der benötigten Farben reduzieren. Es wird nun untersucht wie stark sich die $\chi(N(m,\Delta))$ dadurch reduzieren lässt. Dabei kann man sich nicht auf das bisherigen Ergebnisse für die chromatische Zahl des *Neighborhood Graph* stützen, da gefordert wurde, dass $m \in \Omega(\Delta^2/\log\Delta)$ gilt und man bei iterativer Anwendung auch beliebig kleine m betrachten muss.

Obere Schranke

Für eine genauere Analyse wird eine Methode vorgestellt, wie man eine beliebige gültige m-Färbung in eine gültige q-Färbung, für gewisse q, umwandeln kann. Dieses q wird dann später für die obere Schranke genutzt.

Lemma 3.10

Zu einem gegebenen Graph $G = (V, E)$, mit maximalem Grad Δ und einer gültigen m-Färbung, lässt sich ein q-Färbung konstruieren, für alle q, welche $q + q/(\Delta+1) \geq m$ genügen.

Beweis: Alle Knoten, die mit einer Farbe $x \leq q$ gefärbt sind, behalten ihre Farben. Nun müssen nur noch neue Farben aus $\{1, \cdots, q\}$ für alle Knoten $v \in V$ gefunden werden die eine Farbe haben, die größer als q ist. Sei die Differenz von m und q definiert als t, also $t := m - q$. Laut Voraussetzung gilt $m \leq q + q/(\Delta+1)$. Daraus folgt, dass für t gilt: $t \leq q + q/(\Delta+1) - q = q/(\Delta+1)$. Die Farben, die nicht in $\{1, \cdots, q\}$ aber in der m-Färbung enthalten sind werden als Farben $x_0, \cdots, x_{t-1}$ bezeichnet. Ein Knoten aus G mit Farbe x_i wählt dann eine Farbe aus der Menge $F(i) := \{i(\Delta+1)+1, \cdots, i(\Delta+1)+\Delta+1\}$, die keiner der Farben der Nachbarknoten entspricht.
Es muss gezeigt werden, dass $i(\Delta+1)+\Delta+1 \leq q$, da nur eine Farbe $\leq q$ für eine gültige

q-Färbung gewählt werden darf. Die Variable i kann maximal den Wert $t-1$ annehmen; für t gilt $t \leq q/(\Delta+1)$. Es ist also zu zeigen, dass folgendes gilt:

$$i(\Delta+1)+\Delta+1 \leq q, \text{ für } t \leq \frac{q}{\Delta+1} \text{ und } i = t-1$$

Es gilt:

$$\begin{aligned} t &\leq \frac{q}{\Delta+1} \Rightarrow t(\Delta+1) \leq q \\ &\Rightarrow (t-1+1)\cdot(\Delta+1) \leq q \\ &\Rightarrow (i+1)\cdot(\Delta+1) \leq q \\ &\Rightarrow i\Delta+i+\Delta+1 \leq q \\ &\Rightarrow i(\Delta+1)+\Delta+1 \leq q \end{aligned}$$

Es wurde also gezeigt, dass die größtmögliche Farbe $i(\Delta+1)+\Delta+1$ höchstens q ist. Da $|F(i)| = \Delta+1$, kann der Knoten eine Farbe finden die kein bereits gefärbter Knoten besitzt. Es bleibt noch zu prüfen ob zwei Knoten u, v, die durch eine Kante in G verbunden sind, die gleiche Farbe wählen können. Sei die Farbe von u x_i und die Farbe von v x_j. Da beide Knoten in G benachbart sind und der Graph bereits gültig gefärbt wurde ist offensichtlich, dass $x_i \neq x_j$ (oder $i \neq j$). Da aber $F(i) \cup F(j) = \emptyset$ können beide Knoten nicht dieselbe Farbe wählen. Die entstandene Färbung ist also eine gültige q-Färbung. ■

Lemma 3.11

Für ein beliebiges m gilt:

$$\chi(N(m,\Delta)) \leq \left\lceil m \cdot \frac{\Delta+1}{\Delta+2} \right\rceil = \left\lceil m \cdot \left(1 - \frac{1}{\Delta+2}\right) \right\rceil$$

Beweis: Der *Neighborhood Graph* $N(m,\Delta)$ besitzt bereits eine gültige m-Färbung. In Lemma 3.10 wurde gezeigt, wie man eine gültige m-Färbung in eine gültige q-Färbung umwandelt, wenn $q + q/(\Delta+1) \geq m$. Die Behauptung dieses Lemmas folgt, also wenn $\left\lceil m \cdot \frac{\Delta+1}{\Delta+2} \right\rceil = q$ der Gleichung $q + q/(\Delta+1) \geq m$ genügt. Dazu zunächst ein paar Umformungen bezüglich $q + q/(\Delta+1)$

$$q + \frac{q}{\Delta+1} = q\left(1 + \frac{1}{\Delta+1}\right) = q\left(\frac{\Delta+2}{\Delta+1}\right)$$

Jetzt setzen wir $\left\lceil m \cdot \frac{\Delta+1}{\Delta+2} \right\rceil$ für q in die obige Gleichung ein.

$$\left\lceil m \cdot \frac{\Delta+1}{\Delta+2} \right\rceil \frac{\Delta+2}{\Delta+1} \geq m \cdot \frac{\Delta+1}{\Delta+2} \cdot \frac{\Delta+2}{\Delta+1} = m$$

Damit folgt, die Behauptung des Lemmas. ■

Es ist also möglich eine gültige m-Färbung in eine $(\Delta+1)$-Färbung in $O(\Delta \log(m/\Delta))$ aufeinander folgenden Schritten zu transferieren.

Untere Schranke

Es soll nun gezeigt werden wie viele iterative Anwendungen eines *One Round* Algorithmus benötigt werden um eine gültige m-Färbung eines Graphen um einen konstanten Faktor zu verringern. Dazu benötigen wir Lemma 3.8 das besagt, dass $\chi(N(m,\Delta)) > q$, wenn q Gleichung (3.1) genügt.

Lemma 3.12

Wenn $m = \rho\Delta$ für ein $\rho > 1$ lässt sich ein ϵ finden, so dass:

$$\chi(N(m,\Delta)) \geq (1-\epsilon)\cdot m$$

Beweis: Nach Lemma 3.8 ist $\chi(N(m,\Delta)) > q$, wenn

$$\begin{aligned}\Delta\left(\Delta - \frac{q\ln(eq)}{m-q}\right) &> \frac{2(m-1)q(\ln(eq))^2}{m-q}\\ \Rightarrow \Delta(\Delta(m-q) - q\ln(eq) &> 2(m-1)q(\ln(eq))^2\end{aligned}$$

Substituiert man nun $\rho\Delta$ für m und $(1-\epsilon)m = (1-\epsilon)\rho\Delta$ für q erhält man:

$$\begin{aligned}&\Rightarrow \Delta(\Delta(\rho\Delta - (1-\epsilon)\rho\Delta) - (1-\epsilon)\rho\Delta\ln(e(1-\epsilon)\rho\Delta)) > 2(\rho\Delta-1)(1-\epsilon)\rho\Delta(\ln(e(1-\epsilon)\rho\Delta))^2\\ &\Rightarrow \Delta(\Delta(\underbrace{1-(1-\epsilon)}_{=\epsilon}) - (1-\epsilon)\ln(e(1-\epsilon)\rho\Delta)) > 2(\rho\Delta-1)(1-\epsilon)(\ln(e(1-\epsilon)\rho\Delta))^2\\ &\Rightarrow \Delta(\epsilon\Delta - (1-\epsilon)\ln(e(1-\epsilon)\rho\Delta)) > 2(\rho\Delta-1)(1-\epsilon)(\ln(e(1-\epsilon)\rho\Delta))^2\end{aligned}$$

Wir können o.B.d.A. annehmen, dass $\rho \in O(\Delta/(\log(\Delta))^2)$. Also $\rho \leq k\Delta/(\log(\Delta))^2)$ für eine Konstante k. Es lässt sich ein c finden, für das gilt

$$\ln(e(1-\epsilon)\rho\Delta) \leq c\ln\Delta.$$

Nun folgen zwei weitere Abschätzungen, die mit Hilfe von $\ln(e(1-\epsilon)\rho\Delta) \leq c\ln\Delta$ erreicht werden können.

$$\begin{aligned}&\Delta(\epsilon\Delta - (1-\epsilon)\ln(e(1-\epsilon)\rho\Delta))\\ &\geq \Delta(\epsilon\Delta - \underbrace{(1-\epsilon)}_{\leq 1}c\ln\Delta)\\ &\geq \Delta(\epsilon\Delta - c\ln\Delta)\end{aligned}$$

$$\begin{aligned}&2(\rho\Delta-1)(1-\epsilon)(\ln(e(1-\epsilon)\rho\Delta))^2\\ &\leq 2(\rho\Delta-1)\underbrace{(1-\epsilon)}_{\leq 1}(c\ln\Delta)^2\\ &\leq 2(\rho\Delta)(c\ln\Delta)^2 \leq 2c^2\rho\Delta(\ln\Delta)^2\end{aligned}$$

Das heißt, wenn

$$\Delta(\epsilon\Delta - c\ln\Delta) \geq 2c^2\rho\Delta(\ln\Delta)^2 \tag{3.4}$$

gilt, dann gilt auch

$$\Delta(\Delta(\rho\Delta - (1-\epsilon)\rho\Delta) - (1-\epsilon)\rho\Delta\ln(e(1-\epsilon)\rho\Delta)) > 2(\rho\Delta-1)(1-\epsilon)\rho\Delta(\ln(e(1-\epsilon)\rho\Delta))^2$$

bzw.

$$\Delta\left(\Delta - \frac{q\ln(eq)}{m-q}\right) > \frac{2(m-1)q(\ln(eq))^2}{m-q}$$

für $m = \rho\Delta$ und $q = (1-\epsilon)\rho\Delta$. Daraus folgt, dass es ein passendes ϵ gibt.

$$\chi(N(m,\Delta)) > q = (1-\epsilon)\rho\Delta = (1-\epsilon)m$$

■

Nun wird eine untere Schranke für das ϵ aus Lemma 3.12 gezeigt.

Lemma 3.13

Wenn $m = \rho\Delta$ für ein $\rho > 1$, gilt:

$$\chi(N(m,\Delta)) \geq \left(1 - O\left(\frac{\rho(\log\Delta)^2}{\Delta}\right)\right) \cdot m$$

Beweis: Dass $\chi(N(m,\Delta)) \geq (1-\epsilon)\cdot m$ gilt, wurde bereits in Lemma 3.12 gezeigt, wenn Gleichung (3.4) erfüllt ist.

$$\begin{aligned}
&\Delta(\epsilon\Delta - c\ln\Delta) \geq 2c^2\rho\Delta(\ln\Delta)^2 \\
&\Rightarrow \epsilon\Delta^2 - c\ln\Delta \geq 2c^2\rho\Delta(\ln\Delta)^2 \\
&\Rightarrow \epsilon \geq \frac{2c^2\rho\Delta(\ln\Delta)^2}{\Delta^2 - c\ln\Delta} \in O\left(\frac{\rho(\log\Delta)^2}{\Delta}\right)
\end{aligned}$$

Damit wurde eine untere Schranke für ϵ gefunden. Aus Lemma 2.13 ist bekannt, dass

$$\chi(N(m,\Delta)) > (1-\epsilon)\cdot m$$

und da

$$\epsilon \geq O\left(\frac{\rho(\log\Delta)^2}{\Delta}\right)$$

folgt

$$\chi(N(m,\Delta)) > (1-\epsilon)\cdot m \geq \left(1 - O\left(\frac{\rho(\log\Delta)^2}{\Delta}\right)\right)\cdot m$$

und damit das Lemma. ■

Insgesamt wurde nun gezeigt, dass man ausgehend von einer $\rho\Delta$-Färbung mindestens $\Omega(\Delta/(\rho\log\Delta)^2)$ iterative Schritte benötigt um die ursprüngliche Färbung um einen konstanten Faktor zu verringern.
In [3] wurde gezeigt, dass $\Omega(\log^* m)$ als untere Schranke gilt. Kombiniert man beide Ergebnisse ergibt sich die leicht verbesserte untere Schranke $\Omega(\Delta/(\rho\log\Delta)^2 + \log^* m)$. Das bedeutet auch, dass ein $O(\Delta)$ Färbe-Algorithmus, der auf iterativen *One Round* Algorithmen beruht, mindestens $\Omega(\Delta/(\log\Delta)^2 + \log^* m)$ Schritte benötigt.

4 Zusammenfassung

Die gezeigten Erkenntnisse der Autoren Kuhn und Wattenhofer betrachten die Leistungsfähigkeit von *One Round* Algorithmen. Der eingeführte *Neighborhood Graph* und der Beweis, dass es ausreicht, die chromatische Zahl des *Neighborhood Graph* zu betrachten um das Potential der *One Round* Algorithmen zu beurteilen, hat die Analyse maßgeblich erleichtert. Es konnte gezeigt werden, dass man in einer Runde die Farben eines gültig m-gefärbten Graphen auf höchstens $(\Delta+1)^2(\ln m+1)$ reduzieren kann und das man für entsprechend große m die Anzahl der Farben höchstens auf $\Omega(\Delta^2/(\log\Delta)^2)$ vermindern kann. Außerdem konnte gezeigt werden wie stark sich die Anzahl der Farben durch eine wiederholte Anwendung von *One Round* Algorithmen verringern lässt bzw. wie viele Iterationen notwendig sind um eine $O(\Delta)$-Färbung zu erhalten.

Der große Vorteil der betrachteten *One Round* Algorithmen liegt darin, dass sie ein Netzwerk nicht durch viele Kommunikationsnachrichten belasten. Die Nachrichtengröße ist ebenfalls sehr gering, da nur die eigene Farbe übermittelt werden muss. Allerdings ist eine optimale Lösung, also eine minimale Färbung, nicht zu erwarten, was aber in gewissen Anwendungsbereichen auch nicht notwendig ist.

In Zukunft wäre es wünschenswert, dass man Algorithmen entwickelt die nicht auf dem *Neighborhood Graph* basieren, da dieser äußerst aufwändig zu konstruieren ist und hohe Rechen- und Speicherkapazitäten in den Sensoren voraussetzt.

Literaturverzeichnis

[1] R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*. Plenum Press, New York, 1972. 2.1

[2] Fabian Kuhn and Roger Wattenhofer. On the complexity of distributed graph coloring. In *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 7–15, New York, NY, USA, 2006. ACM Press. 3.2, 3.3.1, 3.3.1, 3.3.2

[3] Nathan Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992. 3.2, 3.3, 3.4

5 Anhang

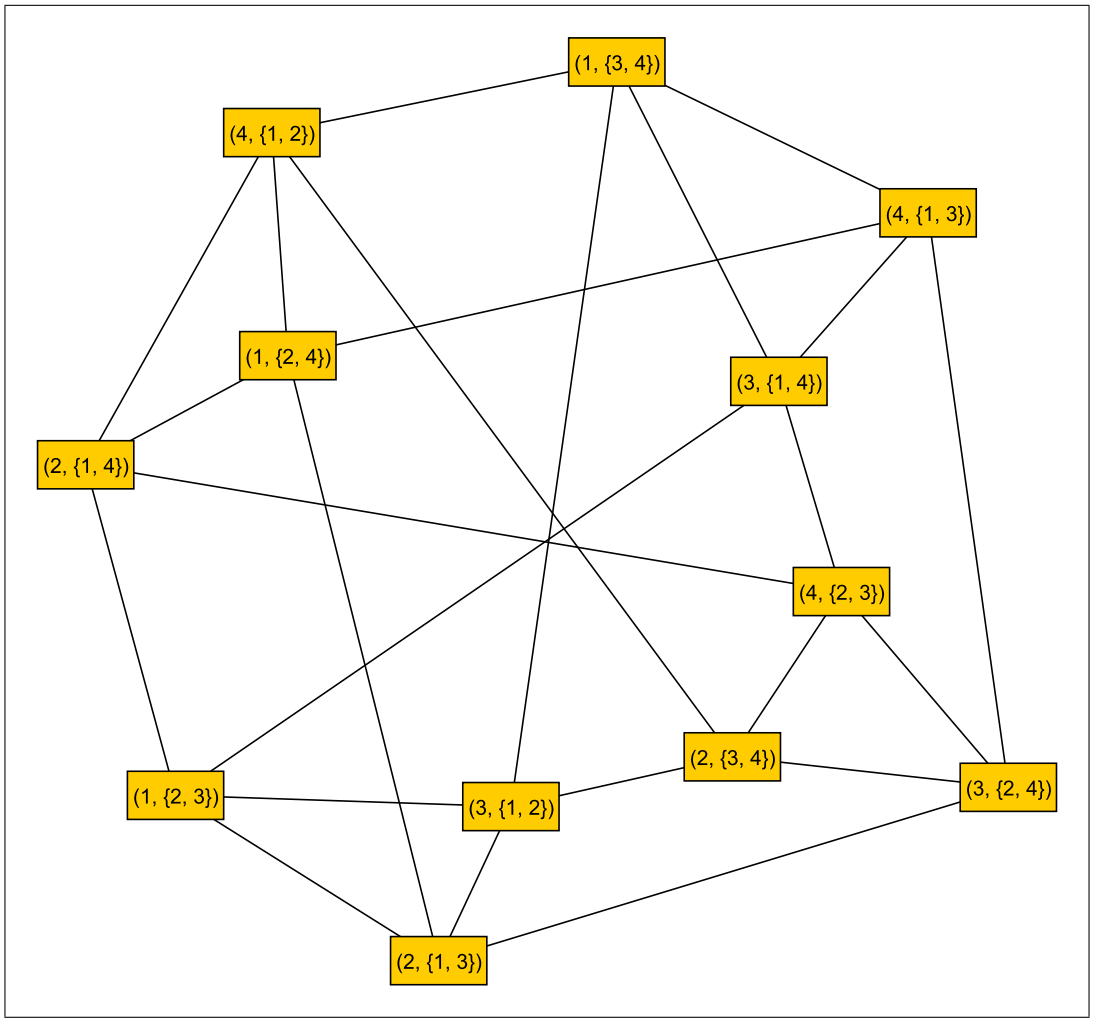

Abbildung 5.1: N(4, 2)