Untersuchung von Graphreduktionsregeln beim Knotenüberdeckungsproblem

Bachelorarbeit

zur Erlangung des akademischen Grades
Bachelor (B.Sc.)

Universität Trier
FB IV - Informatikwissenschaften
Lehrstuhl für theoretische Informatik

Gutachter:	Prof. Dr. Henning Fernau
	Prof. Dr. Stefan Näher
Betreuer:	Prof. Dr. Henning Fernau

Vorgelegt am xx.xx.xxxx von:

Benedikt Lüken-Winkels
Bahnhofstraße 32
54292 Trier
s4beluek@uni-trier.de
Matr.-Nr. 1138844

Zusammenfassung

Hier steht eine Kurzzusammenfassung (Abstract) der Arbeit. Stellen Sie kurz und präzise Ziel und Gegenstand der Arbeit, die angewendeten Methoden, sowie die Ergebnisse der Arbeit dar. Halten Sie dabei die ersten Punkten eher kurz und fokussieren Sie die Ergebnisse. Bewerten Sie auch die Ergebnissen und ordnen Sie diese in den Kontext ein.

Die Kurzzusammenfassung sollte maximal 1 Seite lang sein.

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Motivation 1
1.2 Problemstellung 2
1.3 Zielsetzung 2
1.4 Gliederung/Aufbau der Arbeit 2

2 Grundlagen **3**
2.1 Knotenüberdeckung 3
2.2 Nemhauser-Trotter Reduktionsregeln 3
2.3 Kronenregel 3
2.4 Einfache Reduktionsregeln 3

3 Analyse **4**
3.1 Anforderungen 4
3.2 Bewertung der Reduktionsregeln 4
3.3 Anwendung der Reduktionsregeln 5
3.4 Interpretation der Ergebnisse 10
3.5 Zusammenfassung 10

4 Implementierung **14**
4.1 Kronenregel 14
4.2 Nemhauser-Trotter-Regel 14

5 Diskussion und Ausblick **16**

Literaturverzeichnis **17**

Abbildungsverzeichnis

1.1 Graph eines Stromnetzes 1

3.1 Graph_3 nach Anwendung von Grad_1-Regel und Kronenregel. 7

3.2 Anwendung der Kronenregel. 8

3.3 Einfache Krone 8

3.4 Anwendung von Nemhauser-Trotter-Regel, Kronenregel und Grad_1-Regel. 11

Tabellenverzeichnis

3.1 Bewertung der Reduktionsregeln 5
3.2 Anwendung einzelner Reduktionsregeln 6
3.3 Besondere Graphen für die Kronenregel 9
3.4 Anwendung kombinierter Reduktionsregeln 11
3.5 Besondere Graphen für die Dreierkombinationen von Regeln 12
3.6 Statistiken über Graph_5 13

4.1 Mindestens ein Knoten mit Einschränkung 15
4.2 Beide Knoten mit Einschränkung 15

1. Einleitung

Die Einleitung besteht aus der Motivation, der Problemstellung, der Zielsetzung und einem erster Überblick über den Aufbau der Arbeit.

1.1 Motivation

- Definition
- Wo findet das Knotenüberdeckungsproblem Anwendung?
- Woher kommt die Komplexität?
- Was macht eine schwere Instanz aus?
- Wie sieht eine schwere Instanz aus?

Das *Knotenüberdeckungsproblem*, oder englisch Vertex Cover, ist ein nachgewiesen NP-vollständiges Problem [10]. Um das Problem zu erklären, betrachten wir ein Netz von Haushalten, bei dem wir die Möglichkeit haben, in jedem Haushalt einen Stromgenerator zu platzieren, sodass durch alle mit diesem Haus verbundenen Leitungen Strom fließt. Ziel ist, ein stabiles Stromnetz zu schaffen, bei dem jede Leitung an mindestens eine Stromquelle angeschlossen ist. Nun ergibt sich das Problem, dass die Kosten für Stromgeneratoren erheblich sind und daher nur maximal k Geräte angeschafft werden können. Es gilt also, aus den n Häusern k oder weniger auszuwählen, sodass jede Leitung von einem der Häuser in der Auswahl versorgt wird. In

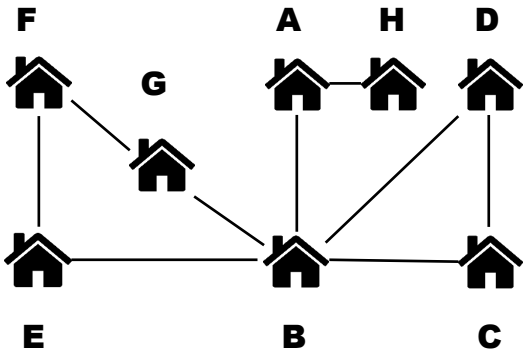

Abbildung 1.1: Graph eines Stromnetzes

Abbildung 1.1[1] sieht man ein Beispiel für eine solche Probleminstanz. Die Haushalte sind jeweils mit einem Buchstaben gekennzeichnet; jede Linie, beziehungsweise

[1] Icons in der Graphik von https://www.flaticon.com/

Kante eine Leitung zum Nachbarhaus. Hier ist es möglich eine Lösung für $k = 4$ zu finden, also 4 Häuser mit Generatoren auszustatten, sodass alle Leitungen versorgt sind. Es ist leicht zu sehen, dass es keine Lösung mit weniger Häusern gibt. Um 4 Häuser, die das Problem lösen zu finden, betrachtet man nun jede Leitung und entscheidet, welches der beiden Häuser aufgenommen werden soll, da mindestens eins der beiden Häuser in der Lösungsmenge ist. Werden alle 2^8 Möglichkeiten ausprobiert, finden sich die Lösung, dass wenn die Haushalte F, B, A und C oder F, B, A und D mit einem Generator ausgestattet werden, durch alle Leitungen Strom fließt. Für eine Problemgröße wie sie hier geschildert wird, bietet die *Brute-Force*-Methode eine Lösung. Wenn nun allerdings ein Häusernetz mit mehreren Tausend Häusern und entsprechend vielen Leitungen zur Diskussion steht, ergibt sich so eine Menge an Möglichkeiten, die nicht in absehbarer Zeit verarbeitet werden kann. Für ein $n = 2000$ bedeutet das $1.148 \cdot 10^{602}$ mögliche Kombinationen. Um die Explosion der Laufzeit zu verhindern

1.2 Problemstellung

- Effekt von Graphreduktionsalgorithmen auf die Problemkomplexität

1.3 Zielsetzung

- Kategorisierung der Regeln?
- Bewertungskriterien für einen GRalgorithmus
  - Laufzeit (Parametrisierung)
  - Erwartete Reduktion/Wie oft wird die Regel angewandt
  - Ressourcenverbrauch
  - Wie gut ist das Ergebnis im Vergleich zu anderen Algorithmen?
- Wie funktionieren die GRA in Kombination?
- Wie sehen Graphen aus, auf die keine Regel anwendbar ist?
- Wie sehen Graphen aus, auf die genau eine Regel anwendbar ist?
- Welche Regeln werden untersucht?

1.4 Gliederung/Aufbau der Arbeit

Was enthalten die weiteren Kapitel? Wie ist die Arbeit aufgebaut? Welche Methodik wird verfolgt?

2. Grundlagen

Beschreibung der verschiedenen Reduktionsregeln und wie funktionieren. Mit graphischen Beispielen und Pseudocode?

2.1 Knotenüberdeckung

...

2.2 Nemhauser-Trotter Reduktionsregeln

...

2.3 Kronenregel

...

2.4 Einfache Reduktionsregeln

...

3. Analyse

3.1 Anforderungen

Um einen Graphen, der zur Diskussion steht auf einen Problemkern zu reduzieren können verschiedene Reduktionsregeln verwendet werden.
Zunächst werden die Regeln anhand der folgenden Kriterien beurteilt:

- Laufzeit
- Erwartete Reduktion
- Ressourcenverbrauch

Dann wird die Anwendung und der Effekt der Regeln auf das eigens erstellte Testset untersucht, um folgende Fragen zu beantworten:

- Wie effektiv sind die Reduktionsregeln in der Anwendung?
  - Wie oft sind die Regeln anwendbar?
  - Wie viel wird reduziert?
- Wie (gut) funktionieren die Reduktionsregeln in Kombination?
- Wie sehen Graphen aus, auf die keine Regel anwendbar ist?
- Wie sehen Graphen aus, auf die genau eine Regel anwendbar ist?

3.2 Bewertung der Reduktionsregeln

Um die verwendeten Reduktionsregeln zu bewerten, beziehungsweise zu vergleichen, betrachten wir auf der einen Seite die theoretisch mögliche Reduktion und auf der anderen, wie sich der implementierte Algortihmus verhält.

In Tabelle 3.1 stehen die Laufzeiten in der O-Notation und erwartete Größe der reduzierten Problemkerns der Nemhauser-Trotter-Regel [1, S67], der Kronenregel [1, S71] [2], die der Buss Regel [1, S67] und von einfachen Reduktionsregeln (Grad 0 und Grad 1) gegenüber. Die Laufzeiten der einfachen Reduktionsregeln ergeben sich folgt:

Tabelle 3.1: Bewertung der Reduktionsregeln

Reduktionsregel	**Laufzeit**	**Reduktion**
Nemhauser-Trotter	$O(k^3)$	$\leq 2k$
Kronenregel	$O(\sqrt{\|V\|}\|E\|)$	$\leq 3k$
Buss	$O(k\|V\|)$	$\leq k^2$
Grad 0	$O(\|V\|)$	*
Grad 1	$O(\|V\|)$	*

* Die Reduktion dieser Regeln hängt stark von der Struktur des Graphen ab. Eine formale Bestimmung der zu erwartenden Reduktion steht noch aus.
k beschreibt jeweils den Eingabeparameter der Knotenüberdeckung für einen Graphen $G = (V, E)$

Beweis. Sei Graph $G = (V, E)$

Grad $0 -$ *Reduktion*

$\underline{1.Fall:}$ $|E| = 0$

jeder Knoten aus V *muss betrachtet werden.*

$\Rightarrow$ *Laufzeit:* $O(|V|)$

$\underline{2.Fall:}$ $|E| > 0$

Sobald bei der Iteration ein Knoten mit $Grad(v) > 0$ $(v \in V)$ *ausgewählt wird, können alle Nachbarn ignoriert werden.*

$\Rightarrow$ *Laufzeit:* $O(x|V|)$ *mit* $(x < 1)$

□

Beweis. Sei Graph $G = (V, E)$

Grad $1 -$ *Reduktion*

Sobald bei der Iteration ein Knoten mit $Grad(v) = 1$ $(v \in V)$ *ausgewählt wird, wird der Nachbar in die Knotenberdeckung aufgenommen.*

$\Rightarrow$ *Laufzeit:* $O(x|V|)$ *mit* $(x < 1)$

□

TODO was bringt die Bewertung?

3.3 Anwendung der Reduktionsregeln

Das Testset an dem die Algorithmen angewandt wurden besteht aus ungerichteten Graphen, die jeweils eine Knotenmenge von 1000 Knoten und eine Kantenanzahl von

1000 bis 10000 (aufsteigend in 200er Schritten) umfassen. Von jeder *Graphklasse*, die sich durch die Kantenmenge auszeichnet gibt es 20 Exemplare. Dadurch entsteht ein Testset von 900 Graphen. Zum Erstellen der Graphen wurde die LEDA-Funktion random_simple_undirected_graph(|V|,|E|) [2] verwendet. Da Zufallsgraphen verwendet werden, gibt es keine Vorgabe für den Parameter k. Die obere Beschränkung der Kantenmenge (10000) hat sich bei den Tests ergeben, da ab einer bestimmten Dichte, beziehungsweise Knotenanzahl keine Reduktionsregel mehr erfolgreich ist, beziehungsweise keine Änderung am eingegebenen Graphen mehr erzielte. Dies kann mit der erwarteten Reduktion (siehe Tabelle 3.1) erklärt werden: Je dichter der Graph wird, desto größer wird k und sobald $k > \frac{|V|}{2}$ wird, ist bei der Nemhauser-Trotter-Regel und somit auch bei den anderen die theoretische Größe des reduzierten Problemkerns größer oder gleich der Menge der Knoten, was bedeutet, dass theoretisch keine Reduktion stattfinden kann. Daher eignen sich viele Benchmarktests [1] [2] nicht, da sich diese durch sehr schwere Probleme mit dichten Graphen und Werten für k, die sich der Größe der Knotenmenge im jeweiligen Problem annähern, auszeichnen. Eine Knotenmenge von 1000 Knoten pro Graph für das Testset erwies sich bei der Anwendung als ausreichend hoher Wert, um das Einsetzen der Reduktionsregeln zu beobachten.
Die Reduktionsregeln wurden jeweils solange auf einen Graphen angewandt, bis sich keine Änderungen mehr ergeben haben. Erzielte mindestens eine der Regeln eine Reduktion, wurde der gesamte Vorgang wiederholt. In der Anwendung sorgte dies dafür, dass für manche Regeln einige weitere Iterationen mögich wurden. In Tabelle 3.3 kann man diesen Effekt beobachten. Die Anwendung der Grad_1-Regel nach der Kronenregel sorgte dafür, dass bei Graph_1 3 und bei Graph_2 sogar 5 weitere Iterationen der Kronenregel eine Reduktion erzielten. Ausgeführt wurden die Experimente auf einem 2.6 GHz, Zweikern, Intel Core i5-3320M mit 8 GB Arbeitsspeicher. In der Tabelle 3.2 werden die durchschnittliche Anzahl der Anwendungen pro Graph, die durschnittliche Reduktion (Anzahl der Knoten, die aus dem Graphen entfernt wurden), und die durchschnittliche CPU-Zeit, die pro Graph aufgebracht wurde. Die Buss Regel wurde von der Untersuchung ausgeschlossen, da kein Wert für k vorliegt, welcher für deren Anwendung essentiell ist, während die restlichen Regeln auch ohne diesen Parameter verwendbar sind.
Tabelle 3.2 spiegelt die erwarteten Werte (3.1) in soweit wieder, als das die Nemhauser-Trotter-Regel eine deutlich bessere Reduktion als die Kronenregel erreicht, zumindest, wenn sie isoliert angewandt wird. Dabei werden weniger Durchläufe, allerdings ein höherer Rechenaufwand benötigt.

Tabelle 3.2: Anwendung einzelner Reduktionsregeln

Reduktionsregel	**Anwendungen**	**Reduktion**	**CPU-Zeit**
Nemhauser-Trotter	0.27	50.3	0.014s
Kronenregel	0.46	19.77	0.005s
Grad 1	1.32	99.06	0.001s

EINFÜGEN

[1] http://dimacs.rutgers.edu/pub/challenge/graph/benchmarks/
[2] http://sites.nlsde.buaa.edu.cn/ kexu/benchmarks/graph-benchmarks.htm

Während sich die Reduktion durch die Nemhauser-Trotter und die Grad 1 Regel bei der Anwendung an dichteren Graphen, also Graphen mit einer höheren Kantenzahl, erwartungsgemäß stetig verringerte, zeigten sich bei der Kronenregel einige Ausnahmen. Wie in Abbildung 3.2 zu sehen ist, setzten sich diese deutlich vom Durchschnitt ab. In Tabelle 3.3 werden die drei Graphen mit einer Kantenmenge über (beziehungsweise gleich) 3000 und einer durch die Kronenregel erreichten Reduktion > 480 Knoten betrachtet. Keiner dieser Graphen ist bipartit. Bei allen drei Graphen führte die Anwendung der Nemhauser-Trotter-Regel zu keinerlei nennenswertem Effekt, weder isoliert, noch in Kombination mit den anderen Regeln. Ähnlich verhielt sich zunächst die Grad_1-Regel, wohingegen die Anwendung in Kombination einen großen Einfluss auf die Reduktion hatte. Kombiniert man die Regeln, scheint die Reihenfolge, in der sie eingesetzt werden schwer ins Gewicht zu fallen. Besonders fiel das bei Graph_1 (3200 Kanten) auf. Wurde zuerst die Grad_1 und dann die Kronenregel verwendet, war jeweils lediglich ein Durchlauf (Anwendung) zu beobachten. Die Grad_1-Regel scheint den Graphen derart zu verändern, dass die Stuktur keine Reduktion durch die Kronenregeln mehr zulässt. Betrachtet man dagegen das Ergebnis von Graph_2, erzeugte die Anwendung in gerade dieser Reihenfolge eine Lösung des Problems: 1000 von 1000 Konten reduziert, wovon 619 eine Knotenüberdeckung für Graph_2 bilden. Auch hier war die Reihenfolge wieder wichtig, wie man bei der Reduktionsmenge beim entgegengesetzten Experiment (erst Kronenregel, dann Grad_1-Regel) sieht, wo 858 Knoten bei der Reduktion aus dem Graphen entfernt wurden. Bei Graph_1 und Graph_3 zeichnet sich der übrig gebliebene Problemkern nach der jeweils größten Reduktion dadurch aus, dass der Großteil der Knoten vom Grad 2 ist. Der Problemkern G'_3 von Graph_3, zu sehen in Abbildung 3.1, lässt sich in die Knotenmengen V_1 und V_2 mit $|V_1| = |V_2| = 5$ aufteilen. Bis auf die Knoten $v_1 \in V_1$ und $v_2 \in V_2$, welche vom Grad 3 sind, hat jeder andere Knoten in G'_3 durch die vorherige Reduktion den Grad 2. Für v_1 und v_2 existiert eine Kante (v_1, v_2) in G'_3, welche die Knotenmengen verbindet. Innerhalb der Knotenmengen existiert einen Zyklus (Kreis), mit ungerader Knotenzahl, woraus sich folgern lässt, dass G'_3 nicht bipartit ist. Hier ist keine der Reduktionsregeln mehr erfolgreich.

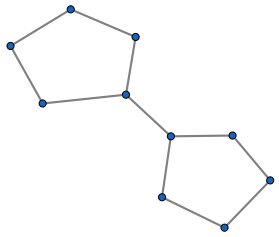

Abbildung 3.1: Graph_3 nach Anwendung von Grad_1-Regel und Kronenregel.

Die in Tabelle 3.4 zusammengefassten Ergebnisse zeigen, dass die Nemhauser-Trotter-Regel in Kombination mit anderen Regeln nicht die gleiche Reduktion erzeugt, wie Grad_1 und Kronenregel. Des Weiteren lassen sich eine Reihe von Beobachtungen anstellen.
Da durch die Grad_1-Regel einfache 1-gradige Knoten, beziehungsweise deren Nachbarn und damit der Kopf einer einfachen Krone, reduziert wird, stellt sie eine vereinfachte Form der Kronenregel dar. Daher sollten intuitiv nach der erschöpfenden

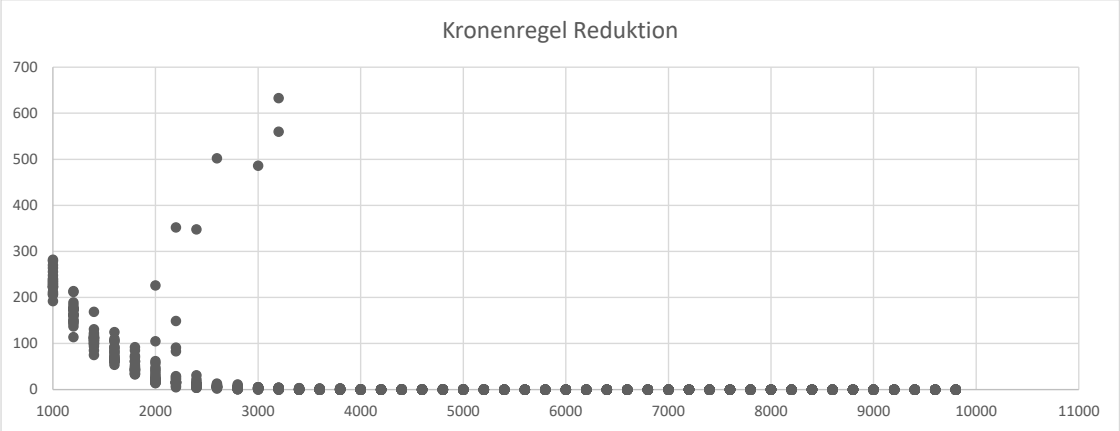

Abbildung 3.2: Anwendung der Kronenregel.

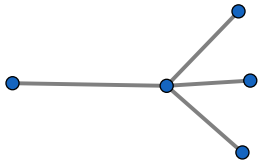

Abbildung 3.3: Einfache Krone

Anwendung der Kronenregel kaum Probleminstanzen, bei denen die Grad_1-Regel erfolgreich ist, übrig bleiben. Wie die Tabelle allerdings zeigt, ist die Kombination beider Regeln im Vergleich zu den anderen Zweierkombination diejenige, die am meißten Knoten aus den Problemgraphen entfernt. Zu dem besseren Ergebnis der Reduktion kommmt hier noch die stark erhöhte durchschnittliche Anwendung pro Graph. Ein Grund hierfür könnte die Auswahl des ersten Matchings in der Kronenregel sein, welches maßgeblich für die darauffolgende Reduktion ist. Eine ëinfache"Krone, wie sie in Abbildung 3.1 zu sehen ist, wird mit großer Wahrscheinlichkeit von der Kronenregel, so wie sie hier verwendet wird, ignoriert. Im Abschnitt 4.1 wird genauer erläutert, warum das der Fall ist: Bei der Anwendung auf das Testset wird die größte Reduktion mit der Kronenregel dann erreicht, wenn bei der Erstellung des Matchings M_1 zunächt Kanten betrachtet werden, bei denen beide Knoten höhergradig, genauer gesagt einen höheren Grad, als der durchschnittliche Knoten im akutellen Graphen haben. Wäre die abgebildete Krone Teil eines größeren Graphen, kann es sein, dass der Kopf dieser Krone nicht in das Matching M_1 aufgenommen und so bei der weiteren Reduktion nicht berücksichtigt wird. Die Grad_1-Regel reduziert demnach jene einfachen Kronen, die von der Kronenregel ignoriert, beziehungsweise nicht erkannt werden. Ändert man nun die Reihenfolge, in der die beiden Regeln angewandt werden, zeigen sich nur minimale Änderungen in der durchschnittlichen Reduktion und der durchschnittlichen Anwendung pro Graph. Das könnte die Vermutung stützen, dass die Grad_1-Regel und die Kronenregel überwiegend verschiedene Arten von Kronen abdecken, da annähernd gleiche Ergebnisse unabhängig von der Reihenfolge der Regeln erzielt werden. Die Unterschiede in den Ergebnissen zeichnen sich dadurch aus, dass die als Zweites verwendete Regel im Schnitt weniger Anwendungen hat, als wenn sie als Erstes verwendet wird. Für die Kronenregel bedeutet das eine Dif-

Tabelle 3.3: Besondere Graphen für die Kronenregel

Graph	**Reduktionsregeln**	**Anwendungen$_1$**	**Anwendungen$_2$**	**Reduktion**
Graph$_1$	Kronenregel	11	-	560
	Nemhauser-Trotter	1	-	4
	Grad$_1$	1	-	18
	Grad$_1$-Kronenregel	1	1	22
	Kronenregel - Grad$_1$	14	11	946
Graph$_2$	Kronenregel	13	-	486
	Nemhauser-Trotter	1	-	6
	Grad$_1$	2	-	40
	Grad$_1$-Kronenregel	12	13	1000
	Kronenregel - Grad$_1$	18	12	858
Graph$_3$	Kronenregel	15	-	633
	Nemhauser-Trotter	1	-	4
	Grad$_1$	2	-	46
	Grad$_1$-Kronenregel	18	11	990
	Kronenregel - Grad$_1$	15	9	971

EINFÜGEN

ferenz von 0.42 durchschnittlichen Durchläufen, für die Grad$_1$-Regel nur 0.07. Diese 0.07 Grad$_1$-Regel-Durchläufe mehr pro Graph verschlechtern allerdings die Gesamtreduktion. Diesen Effekt kann auch bei dem Extrembespiel von Graph$_1$ in Tabelle 3.3 beobachten, wo der gleiche Effekt eintritt. Es scheint also im Großteil der Fälle für eine maximale Reduktion von Vorteil zu sein, wenn die Kronenregel vor der Grad$_1$-Regel angewandt wird, was die die anderen Beispielen in Tabelle 3.4 wiederum untermalen.

Betrachtet man die Grad$_1$-Regel in Kombination mit der Nemhauser-Trotter-Regel sorgt die Reihenfolge zwar für einen Unterschied in der durchschnittlichen Anwendung, allerdings ist das Ergebnis der Reduktion identisch. Ein Teil der von der Grad$_1$-Regel abgedeckten Reduktion wird auch von der Nemhauser-Trotter-Regel entfernt und dem entsprechend auch anders herum. Die Differenz bei verschiedener Reihenfolge beträgt bei der Grad$_1$-Regel 0.2 und bei der NT-Regel 0.26 Iterationen pro Graph, was vermuten lässt, dass die Grad$_1$-Regel Bereiche des Graphen, wo beide Regeln greifen effizienter, mit weniger Durchläufen entfernt. Vergleicht man diese Kombination mit einer Grad$_1$-Kronenregel-Reduktion, fällt auch auf, dass die Form des Graphen, die die Nemhauser-Trotter-Regel durch das Entfernen von Knoten und Kanten erzeugt, die Anwendungsmöglichkeit der Grad$_1$-Regel einschränkt. In die andere Richtung scheint diese Einschränkung auch zu gelten. Die gesamte Reduktion pro Graph erhöht sich nicht sonderlich, vergleicht man die Werte, die die Grad$_1$-Regel alleine erzeugt (Tabelle 3.2).

Kronenregel und Nemhauser-Trotter-Regel sorgen in Kombination dafür, dass die jeweilige Anwendung pro Graph leicht erhöht wird. Auch zeigt sich ein deutlicher Anstieg der Reduktion pro Graph im Vergleich zur Einzelanwendung, was darauf hindeutet, dass die beiden Regeln wiederum verschiedene Bereiche des Graphen entfernen. Bei den Dreierkombinationen zeigt sich dieser Trend ebenfalls: Wird hier die Kronenregel unmittelbar vor der Nemhauser-Trotter-Regel angewandt, finden deutlich mehr Iterationen (der NT-Regel) pro Graph statt, als es bei vorheriger Verwendung der Grad_1-Regel der Fall ist. Zwei Interessante Ergebnisse liefern die Kombinationen Grad_1 - Nemhauser-Trotter - Kronenregel und Grad_1 - Kronenregel - Nemhauser-Trotter. Die Position, beziehungsweise die Reihenfolge der Regeln beeinflusst die Anwendungshäufigkeit von Grad_1 und Nemhauser-Trotter-Regel deutlich, während Gesamtreduktion und Anwendungen der Kronenregel annähernd gleich bleiben. Dies könnte wiederum ein Indikator dafür sein, dass NT und Grad_1-Regel ähnliche Teile des Graphen reduzieren und die Kronenregel den Graphen scheinbar für die jeweilige Reduktion günstig verändert. Werden Nemhauser-Trotter-Regel, Kronenregel und Grad_1-Regel in dieser Reihenfolge angewandt, kann im Schnitt die größte Reduktion pro Graph erreicht werden. In Abbildung 3.4 lassen sich wiederum einige Ausnahmen erkennen: Zwei Graphen stechen besonders aus dem Durchschnitt heraus, zu sehen in Tabelle 3.5. Zum Einen ein Graph mit 2000 Kanten und einer Reduktion von lediglich 195 Knoten (Graph_4), während andere Graphen dieser Größe annähernd komplett gelöst wurden. Zum Anderen ein Graph mit 7200 Kanten, bei dem 762 Knoten entfernt werden konnten (Graph_5). Keiner der Graphen ist - weder vor noch nach der Reduktion - regulär oder bipartit. Die Kronenregel greift bei Graph_5 nur, wenn die Grad_1-Regel den einen eingradigen und dessen Nachbarknoten entfernt. Bis auf die Grad_1-Regel kann keine Reduktionsregel auch nur einen Knoten des Graphen entfernen, wenn sie alleine angewandt werden. Eine weitere interessante Beobachtung für Graph_5 ist, dass die Konfiguration für die Kronenregel sehr wichtig für eine hohe Reduktion ist. Die Knoten des Graphen haben vor der Reduktion einen durchschnittlichen Grad von 14 (abgerundet). Werden beim finden des Matchings M_1 zunächst Kanten, bei denen jeder Knoten den Grad ≥ 14 hat bevorzugt, kann findet keine Reduktion statt, auch nicht in Kombination mit den anderen Regeln. Wählt man als Einschränkung stattdessen Grad > 14 werden 702 Knoten reduziert. Das Beste Ergebnis wird bei einer Konfiguration erreicht, bei der der Grad beider Knoten größer, als der durchschnittliche Grad aller Knoten ist. Auf dieses Phänomen wird in Kapitel 4.1 weiter eingegangen. Bei Graph_4 zeigt sich, wie bei Graph_5, dass die Nemhauser-Trotter-Regel keinen Einfluss auf die Gesamtreduktion hat. Kronenregel und Grad_1-Regel entfernen in Kombination die Größte Menge an Knoten.

3.4 Interpretation der Ergebnisse

- Gegenüberstellung der Reduktionsergebnisse
- Graphen werden dichter $\Rightarrow k$ erhöht sich $\Rightarrow$ Keine Reduktion mehr erfolgreich
-

3.5 Zusammenfassung

Am Ende sollten ggf. die wichtigsten Ergebnisse nochmal in *einem* kurzen Absatz zusammengefasst werden.

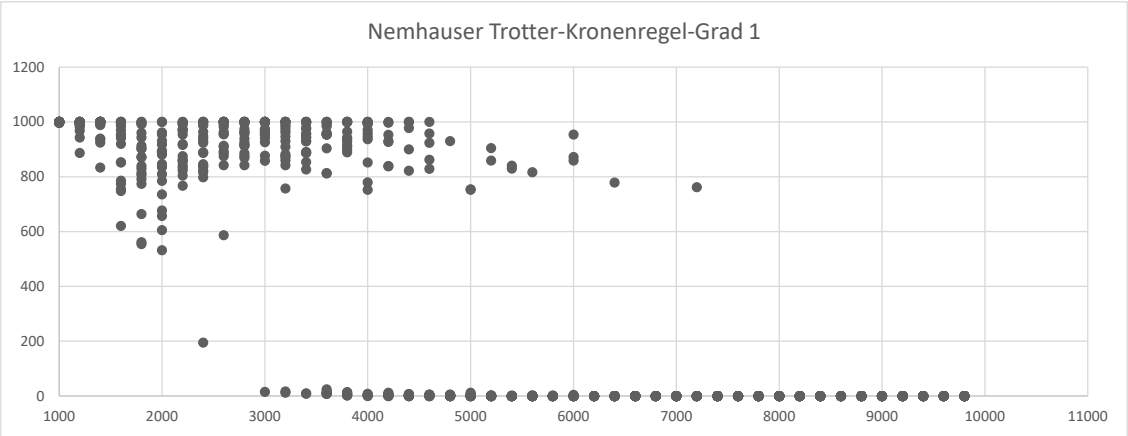

Abbildung 3.4: Anwendung von Nemhauser-Trotter-Regel, Kronenregel und Grad_1-Regel.

Tabelle 3.4: Anwendung kombinierter Reduktionsregeln

Kombination	**Anwendungen$_1$**	**Anwendungen$_2$**	**Anwendungen$_3$**	**Reduktion**
K - G_1	3.63	4.3	-	331.8
G_1 - K	4.37	3.22	-	331.17
K - NT	0.8	0.38	-	68.28
NT - K	0.45	0.56	-	68.6
G_1 - NT	1.33	0.017	-	99.87
NT - G_1	0.28	1.13	-	99.87
K - G_1 - NT	3.61	4.29	0.11	334.67
K - NT - G_1	3.6	0.87	3.39	334.83
G_1 - NT - K	4.36	0.12	3.2	334.17
G_1 - K - NT	3.61	3.2	0.65	334.16
NT - K - G_1	0.39	3.44	4.03	335.2
NT - G_1 - K	0.91	3.42	3.2	334.16

EINFÜGEN

Tabelle 3.5: Besondere Graphen für die Dreierkombinationen von Regeln

Graph	**Reduktionsregeln**	**Anwend.$_1$**	**Anwend.$_2$**	**Anwend.$_3$**	**Reduktion**
Graph_4	NT - K - G_1	1	5	6	195
	G_1 - NT - K	6	0	5	195
	Nemhauser-Trotter	1	-	-	11
	Kronenregel	1	-	-	11
	Grad_1	2	-	-	91
	Kronenregel - Grad_1	6	5	-	195
Graph_5	NT - K - G_1	1	9	2	762
	Nemhauser-Trotter	0	-	-	0
	Kronenregel	0	-	-	0
	Grad_1	1	-	-	2
	Kronenregel - Grad_1	9	3	-	762

EINFÜGEN

Tabelle 3.6: Statistiken über Graph_5

Status	**Grad der Knoten**	**Knoten**	**Grad der Knoten**	**Knoten**
Vor der Reduktion	0	0	15	106
	1	1	16	99
	2	1	17	84
	3	0	18	66
	4	1	19	39
	5	3	20	27
	6	12	21	16
	7	13	22	18
	8	17	23	6
	9	37	24	3
	10	65	25	4
	11	59	26	5
	12	101	27	3
	13	99	28	1
	14	114		
Nach der Reduktion	0	0	6	18
	1	0	7	11
	2	55	8	8
	3	59	9	0
	4	46	10	0
	5	41	11	1

EINFÜGEN

4. Implementierung

Die Regeln wurden in der Programmiersprache *C++* unter Verwendung der Bibliothek *LEDA* [2] implementiert. Sie wurden bei der Anwendung jeweils solange wiederholt, bis sich am Graphen keine Änderung mehr ergab.

4.1 Kronenregel

Beim Austesten der Kronenregel hat sich gezeigt, dass die Auswahl des Matchings M_1 im in Kapitel 2.3 dargestellten Algorithmus das Ergebnis der Reduktion in großem Maße beeinflusst. Wenn also beim Finden von M_1 zunächst Kanten mit Knoten höheren Grades betrachtet werden, können bessere Ergebnisse in der darauf folgenden Reduktion erzielt werden. Daraufhin wurde untersucht, wie die hohergradigen Knoten, beziehungsweise die dazugehörigen Kanten ausgewählt werden müssen. Zum einen (Tabelle 4.1) wurden Kanten betrachtet, bei denen das Auswahlkriterium auf mindestens einen der Knoten zutrifft, auf der anderen Seite Kanten, bei denen beide Knoten die Bedingung erfüllen (Tabelle 4.2). *Größte Anzahl* und *Durchschnittliche Anzahl* ergeben sich jeweils aus der Menge an Knoten eines bestimmtes Grades. Bei ersterem werden Knoten, deren Grad im aktuellen Graphen am häufigsten vorkommt bevorzugt. Bei letzterem dementsprechend Knoten, deren Grad im aktuellen Graphen dem Durchschnitt entspricht. Diese Werte werden bei jeder Iteration des Algorithmus neu berechnet und passen sich dadurch während der Laufzeit an den Graphen an.
Generell wird eine bessere (größere) Reduktion mit der Kronenregel erreicht, wenn beim Machting M_1 zunächst Kanten betrachtet werden, bei denen die Einschränkung auf beide Knoten zutrifft. Die Reduktionsmenge bei Knoten mit $Grad > 2$ erzeugt im Vergleich mit anderen statischen Werten das beste Ergebnis. Dies könnte damit zusammenhängen, dass bei den Graphen, bei denen diese Regel sehr effektiv ist, der durchschnittliche Grad der Knoten 2 ist. Vermutlich erzielt die Bevorzugung des durchschnittlichen Grades das beste Ergebnis, da sich dieser Wert mit jedem Durchlauf verändert.

Wie verhält sich der durchschnittliche Grade der Knoten im Laufe der Iteration?

4.2 Nemhauser-Trotter-Regel

Tabelle 4.1: Mindestens ein Knoten mit Einschränkung

Grad der Knoten	**Anwendungen**	**Reduktion**
keine Einschränkung	0.29	13.04
>1	0.29	13.04
>2	0.29	13.22
>3	0.27	12.92
>4	0.3	13.71
>5	0.31	13.38
Größte Anzahl	0.32	13.44
Durchschnittliche Anzahl	0.29	12.98

Grad der Knoten bezieht sich auf die Bedingung für die bevorzugte Auswahl der Kanten für M_1, bzw. dessen Knoten. *Anwendung* und *Reduktion* stellen jeweils den Durchschnittswert beim gesamten Testset dar.

Tabelle 4.2: Beide Knoten mit Einschränkung

Grad der Knoten	**Anwendungen**	**Reduktion**
keine Einschränkung	0.29	13.04
>1	0.36	15.34
>2	0.41	16.96
>3	0.39	16.52
>4	0.4	15.78
>5	0.4	15.72
Größte Anzahl	0.29	13.06
Durchschnittliche Anzahl	0.46	19.77

Grad der Knoten bezieht sich auf die Bedingung für die bevorzugte Auswahl der Kanten für M_1, bzw. dessen Knoten. *Anwendung* und *Reduktion* stellen jeweils den Durchschnittswert beim gesamten Testset dar.

5. Diskussion und Ausblick

- Form von randomgraphen untersuchen
  - Welche Form hinterlassen die Reduktionsregeln
  - Welche Form wäre für die Nemhauser Trotter am besten
- Warum ist die Nemhauser-Trotter-Regel in der Praxis so schlecht?
- Wie ergibt sich in bei den Reduktionen (Abbildung 3.4) der Große Unterschied zwischen den Graphen im Bereich der Kantenmenge zwischen 3000 und 4600?
- Wieso ist das Matching M1 so wichtig für die Kronenregel?

Literaturverzeichnis

[1] R. Niedermeier, *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.

[2] K. Mehlhorn and S. Näher, *LEDA: A Platform for Combinatorial and Geometric Computing.* Cambridge University Press, 2006.

[3] E. C. Sewell, S. H. Jacobson, and H. Kaul, "Reductions for the stable set problem," *Algorithmic Operations Research Vol.6 40–55*, 2011.

[4] F. N. Abu-Khzam, M. A. Langston, P. Shanbhag, and C. T. Symons, "Scalable parallel algorithms for fpt problems," *Algorithmica, Volume 45, Number 3, 269-284*, 2006.

[5] F. Abu-Khzam, R. Collins, M. Fellows, M. Langston, and W. S. C. Symons, "Kernelization algorithms for the vertex cover problem: Theory and experiments," *Proceedings of the 6th Workshop on Algorithm Engineering and Experiments (ALENEX), ACM/SIAM, Proc. Applied Mathematics 115*, 2004.

[6] M. Cygan, *Parameterized algorithms.* Cham : Springer, 2015.

[7] F. Gurski, *Exakte Algorithmen für schwere Graphenprobleme.* Heidelberg : Springer, 2010.

[8] R. G. Downey and M. R. Fellows, *Fundamentals of parameterized complexity.* London : Springer, 2013.

[9] ——, *Parametrized Complexity.* New York: Springer, 1997.

[10] M. R. Garey and D. S. Johnson, *Computers and intractability : a guide to the theory of NP-completeness.* New York: Freeman, 2007.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich diese Bacheloarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die aus fremden Quellen direkt oder indirekt übernommenen Gedanken als solche kenntlich gemacht habe. Die Arbeit habe ich bisher keinem anderen Prüfungsamt in gleicher oder vergleichbarer Form vorgelegt. Sie wurde bisher auch nicht veröffentlicht.

Trier, den 2. März 2018