

Neue Strategien zur Lösung von Isomorphieproblemen

Dem Fachbereich Mathematik der
Universität Bayreuth
zur Erlangung des akademischen Grades eines
Dr. rer. nat.

eingereichte Dissertation

von
Matthias Koch
aus
Würzburg

1. Gutachter:
2. Gutachter:

Tag der Einreichung:

Tag des Kolloquiums:

Summary

Deciding whether two objects with a given structure are identical is a common problem in mathematics, just as in other areas such as chemistry or electronic design automation. For instance, the question whether two graphs can be made equal by renumbering their nodes is a classical isomorphism problem. In electronic design automation, effectively telling whether two given logic circuits are implementing the same logic is a common task.

Some of these problems are easy to solve, while others seem to have no efficient algorithmical solution. The subgraph isomorphism problem, for example, is known to be NP-complete and thus difficult to solve. The same applies to the graph isomorphism problem, which might be NP-complete, although this has not been proven.

Homomorphisms of group actions have turned out to be one of the most powerful concepts to solve those isomorphism problems. The Homomorphism Principle by Reinhard Laue established the mathematical basis of many of the most efficient algorithms. However, this approach seemed futile in situations where no homomorphism in the desired direction exists.

This changed, when Bernd Schmalz discovered a solution to this problem using homomorphisms in the opposite of the usual direction. Schmalz named this technique „Leiterspiel“ after the children’s game „Snakes and Ladders“ and developed a very successful breadth first search algorithm applying this technique.

Solving an impressive diversity of problems, Schmalz’ Snakes and Ladders algorithm showed its surprising flexibility. But in its original form, this algorithm still had some serious limitations. In the present thesis the Snakes and Ladders algorithm has been given a new mathematical foundation to overcome those limitations. On the basis of strong ladders, the algorithm has been considerably improved in the following ways:

- With the original Snakes and Ladders algorithm it was not possible to compute the orbit representative for a single coset. The only possible way to use this algorithm was to solve all isomorphism problems in the given domain simultaneously. Given that there is only one result of interest, this method clearly was inefficient. With each of the three new algorithms presented in the current thesis it is possible to calculate the canonical representative for a single coset.
- When solving a given problem with the original Snakes and Ladders algorithm, all intermediate results must be stored until the end of the calculation. But for the majority of isomorphism problems, the quantity of intermediate results increases rapidly with problem size. Therefore the original Snakes and Ladders can only be used for problems where all intermediate results can be stored in a fast working memory. Until now, there appeared to be no alternative to storing all intermediate results, since Snakes and Ladders accesses them in no predictable order. In this thesis, the requirements on the memory size could be restricted by calculating only the intermediate results needed for the retrieval of the canonical representative. Therefore, each of the three new algorithms can also be used for more complex isomorphism problems, each one limited only by the available calculating time.
- In its original form the Snakes and Ladders algorithm is not applicable for calculation on a distributed system. This is due to the fact that every processing unit needs access to a central memory area, where all intermediate results are stored. As there is no central memory area in a distributed system, all intermediate results need to be transferred among the involved processing units. Depending on the latency of the communication channel, this can have severe consequences on the efficiency of the algorithm. Two out of the three new algorithms presented in this thesis have near to no need for storing intermediate results. Those two algorithms can be used for calculations on a distributed system.

For every Snakes and Ladders algorithm, a series of homomorphisms, the so-called ladder, is needed. The mathematical foundation for the new algorithms are the so-called strong ladders, which require the homomorphisms to satisfy certain additional conditions. This thesis establishes the mathematical basis of these ladders clearly and formulates proof for each conclusion.

The three novel algorithms, called Depthfirst StroLL, Breadthfirst StroLL and Leiterspiel Light, were implemented in the C++ programming language. Along with Schmalz' original Snakes and Ladders, they were tested on their comparative performance and memory requirements.

The algorithms of the present thesis have been deliberately designed to minimize their memory requirements. Two out of the three new algorithms recalculate intermediate results instead of storing them in the memory. Those two algorithms use a variant of the branch-and-bound method to identify and cut unused branches. This way, the time used for the recalculation of intermediate results could be compensated. Thus, these algorithms have very modest working memory requirements and still show the same efficiency as the original Snakes and Ladders algorithm.

In order to use the algorithms presented in this work, specific ladders tailored to the problem must be found. A construction strategy presented in this thesis helps the user to build the required ladders.

Using the well-known Split Lemma many isomorphism problems can be mapped to double cosets. This is why the new algorithms presented in this thesis are formulated on the basis of double cosets. This way, a great diversity of problems as different as the graph isomorphism problem or the calculation of the intersection of two groups can be solved using the same algorithm.

Finally, the current thesis also shows how to map the subgraph isomorphism problem to double cosets. This so-transformed problem can then be solved using any of the three new algorithms.

Zusammenfassung

In vielen Bereichen der Mathematik, in der Chemie und beim automatisierten Entwurf elektronischer Systeme muss für zwei verschiedene Strukturbeschreibungen festgestellt werden, ob die beschriebenen Objekte identisch sind. Ein klassisches Isomorphieproblem ist zum Beispiel die Fragestellung, ob zwei verschiedene Graphen durch eine Umbenennung ihrer Knotenmengen identisch werden können. Beim automatisierten Entwurf elektronischer Systeme taucht ein ähnliches Problem auf, wenn für zwei unterschiedliche elektronische Schaltpläne einer logischen Schaltung festgestellt werden soll, ob die beiden Schaltungen sich logisch identisch verhalten.

Einige dieser Problemstellungen sind leicht zu lösen, während für andere wiederum kein effizienter Lösungsalgorithmus bekannt ist. Beim Teilgraphenisomorphieproblem zum Beispiel konnte nachgewiesen werden, dass es in der Klasse der NP-vollständigen Probleme enthalten ist und aufgrund dessen nur schwer zu lösen ist. Ähnliches gilt für das Graphenisomorphieproblem, bei dem jedoch bisher nicht festgestellt werden konnte, ob es in der Klasse der NP-vollständigen Probleme liegt.

Bei vielen Isomorphieproblemen hat sich gezeigt, dass Homomorphismen von Gruppenoperationen ein mächtiges Werkzeug zur Lösung dieser Probleme sein können. Das von Reinhard Laue gefundene Homomorphieprinzip ist die mathematische Grundlage von vielen der effizientesten Algorithmen zur Lösung dieser Isomorphieprobleme. Lange Zeit schien es jedoch unmöglich zu sein, das Homomorphieprinzip auch in den Situationen, in denen kein Homomorphismus in der gewünschten Richtung existiert, einsetzen zu können. Dies änderte sich, als Bernd Schmalz eine Technik entdeckte, bei der ein Homomorphismus auch in der umgekehrten zu der sonst üblichen Richtung verwendet werden kann. Er benannte seine „Leiterspiel“-Technik nach einem Spiel für Kinder und entwickelte einen sehr erfolgreichen Algorithmus in Breitensuche, der diese Technik verwendet.

Dieser Leiterspiel-Algorithmus ist äußerst vielseitig und kann zur Lösung einer beeindruckend großen Vielfalt von Isomorphieproblemen genutzt werden.

Gleichzeitig ist der Algorithmus gewissen Schranken unterworfen, die dazu führen, dass dieser Leiterspiel-Algorithmus in der Praxis nicht für alle Isomorphieprobleme geeignet ist. Im Rahmen dieser Arbeit konnte auf Basis der starken Leitern eine neue mathematische Grundlage gelegt werden, die es ermöglicht, diese Schranken aufzuheben. Der ursprüngliche Leiterspiel-Algorithmus von Schmalz konnte in folgenden Punkten entscheidend verbessert werden:

- Mit dem ursprünglichen Leiterspiel-Algorithmus war es nicht möglich, den kanonischen Repräsentanten zu einer einzelnen Nebenklasse zu berechnen, sondern es mussten immer alle Isomorphieprobleme der gegebenen Problemstellung gleichzeitig gelöst werden. Dieser Ansatz ist verständlicherweise nicht effizient, wenn nur die Lösung eines einzelnen Isomorphieproblems gefragt ist. Mit den drei neuen Algorithmen dieser Arbeit ist es hingegen möglich, auch zu einer einzelnen Nebenklasse deren kanonischen Bahnrepräsentanten zu berechnen.
- Beim ursprünglichen Leiterspiel-Algorithmus mussten alle Zwischenergebnisse, die während der Bearbeitung einer gegebenen Problemstellung berechnet wurden, im Speicher gehalten werden. Da die Anzahl dieser Zwischenergebnisse jedoch sehr schnell anwächst, ist der ursprüngliche Leiterspiel Algorithmus in der Praxis nur für diejenigen Problemstellungen geeignet, bei denen die Zwischenergebnisse im Hauptspeicher Platz finden. Bis zur Veröffentlichung dieser Arbeit war es nicht möglich vorherzusagen, welche Zwischenergebnisse zur Berechnung eines kanonischen Bahnrepräsentanten benötigt werden. Mit den drei neuen Algorithmen dieser Arbeit ist es hingegen möglich, gezielt nur diejenigen Zwischenergebnisse zu berechnen und zu speichern, die auch tatsächlich zur Lösung dieses Isomorphieproblems benötigt werden. Dadurch konnte der Speicherbedarf drastisch reduziert werden. Daher können die drei neuen Leiterspiel-Algorithmen auch für umfangreiche Isomorphieprobleme angewendet werden und sind praktisch nur noch durch die zur Verfügung stehende Rechenzeit begrenzt.

- Der Leiterspiel-Algorithmus war in seiner ursprünglichen Form nur sehr begrenzt für eine verteilte und echt nebenläufige Berechnung geeignet. Denn jeder Prozess benötigt den Zugriff auf einen zentralen Speicherbereich, in dem die Zwischenergebnisse gespeichert werden. Da bei verteilten Systemen kein zentraler Speicherbereich vorgesehen ist, müssen die benötigten Zwischenergebnisse unter den beteiligten Prozessen ausgetauscht werden. Abhängig von der Latenzzeit der Kommunikationswege kann dies sehr negative Auswirkungen auf die Effizienz haben. Zwei der drei neuen Algorithmen kommen mit der Speicherung von sehr wenigen Zwischenergebnissen aus. Diese beiden Algorithmen sind ohne Weiteres auch für eine verteilte Berechnung geeignet.

Für alle Leiterspiel-Algorithmen wird eine Menge Homomorphismen benötigt, die durch eine sogenannte Leiter beschrieben werden. Die Basis dieser neuen Algorithmen sind die sogenannten starken Leitern, das sind Leitern, bei denen die verwendeten Homomorphismen weitere Bedingungen erfüllen müssen. In dieser Arbeit wird die mathematische Grundlage dieser Leitern klar herausgearbeitet und alle Schlussfolgerungen werden mit Beweisen belegt.

Die drei neuen Algorithmen dieser Arbeit, der Breadthfirst StroLL Algorithmus, der Depthfirst StroLL Algorithmus und der Leiterspiel Light Algorithmus wurden in der Programmiersprache C++ implementiert. Um die Effizienz der drei neuen Algorithmen unter Beweis zu stellen, wurden diese mit dem ursprünglichen Leiterspiel-Algorithmus von Schmalz in Bezug auf ihre Speicheranforderungen und ihre Laufzeit verglichen.

Die Algorithmen dieser Arbeit wurden so konzipiert, dass ihre Speicheranforderungen minimal sind. Bei zwei der drei Algorithmen führt dies dazu, dass Zwischenergebnisse mehrfach berechnet werden müssen. Bei diesen beiden Algorithmen konnte auf Basis der starken Leitern jedoch eine Variante der Branch-and-Bound Methode angewendet werden, die es ermöglicht, überflüssige Bahnelemente zu identifizieren und von der Untersuchung auszuschließen. Auf diese Weise konnte die Effizienz dieser beiden Algorithmen so gesteigert werden, dass diese trotz der Einschränkungen hinsichtlich ihres Speicherbedarfs eine vergleichbare Effizienz aufweisen, wie der ursprüngliche Leiterspiel-

Algorithmus.

Um die in dieser Arbeit vorgestellten Algorithmen anwenden zu können, müssen speziell auf das entsprechende Problem zugeschnittene starke Leitern bereitgestellt werden. Daher wurde in dieser Arbeit ein Weg beschrieben, wie die entsprechenden Leitern berechnet werden können.

Mit Hilfe des seit Langem bekannten Split Lemmas lassen sich sehr viele Isomorphieprobleme in Nebenklassenisomorphieprobleme umformulieren. Der in dieser Arbeit gewählte Ansatz, die Leiterspiel-Algorithmen auf der Basis von Doppelnebenklassen zu formulieren, erlaubt es, die drei Algorithmen zur Lösung einer Vielzahl von weiteren Problemen einsetzen zu können. Auf diese Weise können so unterschiedliche Probleme, wie das Graphenisomorphieproblem und die Berechnung der Schnittmenge zweier Gruppen, mit demselben Algorithmus gelöst werden.

Im Rahmen dieser Arbeit wurde auch gezeigt, wie sich das Teilgraphenisomorphieproblem auf ein Nebenklassenisomorphieproblem abbilden lässt. Auch dieses Nebenklassenisomorphieproblem kann anschließend mit jedem der drei neuen Algorithmen gelöst werden.

Inhaltsverzeichnis

Abbildungsverzeichnis	xiii
1 Für den eiligen Leser	1
2 Motivation	3
3 Grundlagen	9
3.1 Sätze und Definitionen	9
3.2 Homomorphieprinzip	15
3.3 Ordnungstreue Erzeugung	17
3.4 Doppelnebenklassen	20
3.5 Split Lemma	23
4 Das Leiterspiel nach Schmalz	29
4.1 Voraussetzungen	30
4.2 Fusionierende Abbildung	31
4.3 Splitting Orbits	32
4.3.1 Voraussetzungen	33
4.3.2 Vorgehen beim Splitting Orbits	33
4.4 Fusing Orbits	34
4.4.1 Voraussetzungen	34
4.4.2 Bestimmung der Bahnrepräsentanten	34
4.4.3 Bestimmung der Stabilisatoren	34
5 Der Kanonisierungsalgorithmus am Beispiel eines Würfels	37
5.1 Drehungen eines Würfels	37
5.2 Fragestellung	38
5.3 Kanonische Färbung und Totalordnung	40

5.4	Stabilisatoren	41
5.5	Homomorphismen	41
5.6	Konstruktionspfade	42
5.7	Vorbereitungen	43
5.8	Erweiterung des Stabilisators	44
5.9	Menge der Pfade	46
5.10	Schnittmengen von Stabilisatoren	46
5.11	Fusionierende Drehung	48
5.12	Tiefensuche	52
5.13	Abschneiden von Ästen	55
5.14	Ergebnisse	57
6	Mathematische Grundlagen der Algorithmen	59
6.1	Grundbegriffe	60
6.1.1	Erweiterungsfunktion	60
6.1.2	Ordnungsrelationen	61
6.1.3	Kanonizitätsprädikat	63
6.2	Konstruktionspfade	65
6.3	Ordnungstreue Abbildungen	67
6.4	Blöcke aus Konstruktionspfaden	69
6.5	Starke Untergruppenleiter	71
6.6	Urbilder von G -Homomorphismen	80
7	Kanonizitätstests für Doppelnebenklassen	83
7.1	Teilgraphenisomorphieproblem und Doppelnebenklassen	86
7.2	Würfelfärbungen und Rechtsnebenklassen	89
7.3	Breadthfirst StroLL Algorithmus	91
7.3.1	Überblick	92
7.3.2	Bestimmung der zu untersuchenden Nebenklassen	92
7.3.3	Ein Isomorphismus von Gruppenoperationen	93
7.3.4	Eine weitere Untergruppenleiter	94
7.3.5	Inklusions-Homomorphismus	94
7.3.6	Fusionierende Abbildung	95
7.3.7	Splitting Orbits	95
7.3.8	Fusing Orbits	98

7.3.9	Varianten des Algorithmus	99
7.4	Depthfirst StroLL Algorithmus	101
7.4.1	Überblick	101
7.4.2	Vorbereitungen	103
7.4.3	Splitting Orbits	106
7.4.4	Fusing Orbits	116
7.4.5	Pseudocode	127
7.5	Leiterspiel Light Algorithmus	134
7.5.1	Überblick	134
7.5.2	Vorbereitung	136
7.5.3	Splitting Orbits	140
7.5.4	Fusing Orbits	141
7.5.5	Berechnung des Stabilisators von $A_k q$	143
7.6	Unterprogramme des Kanonizitätstests	145
7.6.1	Identitäten und inverse Homomorphismen	145
7.6.2	ReduceStab	147
7.6.3	Canonizer	151
7.6.4	ExtendGroup	153
7.6.5	FindOrbitRep	154
7.6.6	FindSmallestPath	155
8	Analyse des Laufzeitverhaltens	159
8.1	Kurzbeschreibung der Algorithmen	160
8.1.1	Leiterspiel-Algorithmus von Schmalz	160
8.1.2	Leiterspiel Light	160
8.1.3	Depthfirst StroLL	161
8.1.4	Breadthfirst StroLL	161
8.2	Vergleich der Eigenschaften	161
8.2.1	Der Beispielgraph	162
8.2.2	Voraussetzungen	163
8.2.3	Ergebnisse	164
8.3	Vergleich der Leistungsfähigkeit	167
8.3.1	Voraussetzungen	168
8.3.2	Ergebnisse	169
8.3.3	Ergebnis des Vergleichs	171

9 Ausblick	173
Literaturverzeichnis	175
Eidesstattliche Erklärung	180

Abbildungsverzeichnis

3.1	Zwei schlichte Graphen auf acht Knoten	24
5.1	Die 24 Drehungen eines Würfels	38
5.2	Würfel mit weiß markierten Ecken	39
5.3	Würfel mit Färbungstupel	41
5.4	Beispiel Fusing-Homomorphismus	42
5.5	Beispiel Splitting-Homomorphismus	42
5.6	Der kleinste Pfad zum Würfel in Abbildung 5.2	44
5.7	Würfel des Pfades ρ und deren Stabilisatoren	45
5.8	Eine Untergruppe des Stabilisators	46
5.9	Alle Pfade zum Würfel in Abbildung 5.2	47
5.10	Beispiel eines Pfades	49
5.11	Würfel zur Verlängerung eines Pfades	49
5.12	Würfel mit fusionierender Drehung	50
5.13	Wende fusionierende Drehung an	50
5.14	Stabilisator des Bahnrepräsentanten	50
5.15	Bahn unter der Operation des Stabilisators	51
5.16	Würfel mit fusionierender Drehung	51
5.17	Darstellung fusionierender Drehungen	53
5.18	Neuer erweiterter Stabilisator	54
5.19	Abschneiden von Pfaden	54
5.20	Würfel mit Stabilisator	55
5.21	Abschneiden von zwei Ästen	56
5.22	Abschneiden von drei Ästen	56
5.23	Anteil der im Algorithmus betrachteten Würfelfärbungen	58
7.1	Zwei schlichte Graphen auf acht und auf vier Knoten	86

7.2	Vereinigungsgraph \mathcal{C} auf 12 Knoten	87
7.3	Kommutatives Diagramm	90
7.4	Homomorphismen zwischen den Leitern	102
7.5	Bahnen von H	124
7.6	Bijektion zwischen Urbildmengen	142
8.1	Graph aus 10 Knoten und 25 Kanten	162
8.2	Anzahl aller durchlaufenen Nebenklassen	164
8.3	Bahnlängen aller Nebenklassen A_{ip} unter der Gruppe B	165
8.4	Bedarf an Arbeitsspeicher der drei neuen Algorithmen	166
8.5	Bedarf an Rechenzeit der drei neuen Algorithmen	166
8.6	Vergleich der Laufzeit	169
8.7	Vergleich der Laufzeit pro Graph	170
8.8	Bedarf an Arbeitsspeicher	170
8.9	Durchschnittlicher Speicherbedarf pro Nebenklasse	171
8.10	Leiterspiel nach Schmalz, Ergebnisse tabellarisch	172
8.11	Leiterspiel Light, Ergebnisse tabellarisch	172

Kapitel 1

Für den eiligen Leser

Im Folgenden wird ein kurzer Überblick über den Inhalt aller Kapitel dieser Arbeit gegeben.

In Kapitel 2 wird der Leser anhand von Beispielen an das Thema dieser Dissertation herangeführt. Dieses Kapitel kann von allen Lesern, die bereits motiviert sind, übersprungen werden.

In Kapitel 3 wird die mathematische und algorithmische Basis dieser Arbeit gelegt. Neben grundlegenden mathematischen Sätzen und Definitionen werden das Homomorphieprinzip und die Ordnungstreue Erzeugung eingeführt. Mit den Algorithmen dieser Arbeit lassen sich Repräsentanten von Doppelnebenklassen berechnen. Der Zusammenhang zwischen diesen Doppelnebenklassen und anderen Isomorphieproblemen, wie zum Beispiel dem Graphenisomorphieproblem, wird in Kapitel 3.5 erläutert. Dieses Kapitel ist absolut notwendig für das Verständnis dieser Arbeit.

In Kapitel 4 wird das Leiterspiel nach Schmalz beschrieben. Da alle Algorithmen dieser Arbeit auf dem ursprünglichen Leiterspiel von Schmalz aufbauen, ist dieses Kapitel sehr hilfreich, aber nicht notwendig für das weitere Verständnis.

In Kapitel 5 wird einer der Algorithmen dieser Arbeit anhand eines einfachen Beispiels vorgeführt. Hier werden dem Leser die Ideen, die hinter den

drei neuen Leiterspiel-Algorithmen dieser Arbeit stehen, näher gebracht. Anhand von verschieden gefärbten Würfeln werden dem Leser die Konzepte und der Ablauf von einem der neuen Algorithmen vorgestellt. Die mathematischen Konzepte werden dabei anschaulich skizziert, aber nicht genauer ausgeführt.

Kapitel 6 stellt den mathematischen Kern dieser Arbeit dar. Hier werden neue Begriffe wie starke Untergruppenleitern und Konstruktionspfade definiert. Weiterhin werden die Ordnungen auf den Nebenklassenmengen und das im Folgenden verwendete Kanonizitätsprädikat festgelegt. Aufbauend auf diesen Definitionen wird eine Vielzahl von Sätzen bewiesen, die sowohl zur Konstruktion starker Leitern als auch für die später beschriebenen Algorithmen notwendig sind. Die mathematischen Sätze und die Definition der starken Leitern werden in Kapitel 6.5 beschrieben. Diese sind schon allein unter dem gruppentheoretischen Blickwinkel sehr interessant, da diese auch als Beweisgrundlage für weitere Sätze der Gruppentheorie dienen können.

In Kapitel 7 werden drei verschiedene, bisher unbekannte Algorithmen beschrieben. Diese dienen dazu, die Kanonizität einer gegebenen Nebenklasse zu untersuchen und den Stabilisator dieser Nebenklasse zu berechnen. Weiterhin wird in Kapitel 7.2 auch gezeigt, wie diese Algorithmen zur Lösung von weiteren Isomorphieproblemen, wie dem Graphenisomorphieproblem verwendet werden können. In Kapitel 7.1 wird beschrieben, wie das Teilgraphenisomorphieproblem auf ein Doppelnebenklassenproblem abgebildet werden kann. Auch das Teilgraphenisomorphieproblem kann mit jedem dieser drei neuen Algorithmen gelöst werden. Diese Algorithmen sind zusammen mit den starken Leitern die zentralen Ergebnisse dieser Arbeit.

In Kapitel 8 wird anhand einer Beispielrechnung die Effizienz der drei Algorithmen aus Kapitel 7 mit der Effizienz des ursprünglichen Leiterspiel-Algorithmus von Schmalz verglichen. Dabei werden nicht nur die Laufzeit, sondern auch der Speicherbedarf der Algorithmen untersucht.

In Kapitel 9 wird angesprochen, welche neuen Fragen diese Arbeit aufgeworfen hat und in welchen Richtungen noch weiterer Forschungsbedarf besteht.

Kapitel 2

Motivation

In diesem Kapitel soll das Thema der vorliegenden Arbeit vorgestellt und das Interesse des Lesers geweckt werden. Daher wird an manchen Stellen auf eine ausführlichere Darstellung der angesprochenen Themen verzichtet.

Stellen Sie sich vor: Sie sind Chemiker und haben gerade ein sehr interessantes Molekül gefunden. Möglicherweise hat das Molekül überraschende chemische Eigenschaften, lässt sich zur Herstellung von Wasserstoff verwenden oder es verhindert das Wachstum von Tumoren. Dann möchten Sie natürlich wissen, ob dieses Molekül bereits von anderen Chemikern untersucht wurde, ob weitere Eigenschaften des Moleküls bekannt sind und ob bereits ein Patent auf diesen Stoff angemeldet wurde. Viele Moleküle haben sogenannte Trivialnamen, das sind Bezeichnungen, die von der Struktur des Moleküls unabhängig sind, die sich aber unter Chemikern durchgesetzt haben, wie zum Beispiel der Name Glaubersalz für Natriumsulfat. Wenn das Molekül bisher noch nicht bekannt war, so besitzt es natürlich auch noch keinen Trivialnamen. Daher benötigen Sie für das Molekül eine international anerkannte Bezeichnung, mit Hilfe derer sie in Datenbanken und in Publikationen nach ihrem Molekül suchen können. Um eine eindeutige Bezeichnung für ein kleines Molekül zu finden, genügt es, ein paar einfache Regeln anzuwenden, die jeder Chemiker während des Studiums erlernt hat. Für größere und kompliziertere Moleküle existieren international anerkannte Regelwerke, die eine eindeutige Bezeichnung jedes Moleküls ermöglichen. Natürlich gibt es auch Software, die in der Lage ist, zu jedem Molekül die international anerkannte Bezeichnung zu finden. Für den Mathe-

matiker fällt die Aufgabe, einem Molekül einen eindeutigen Namen zu geben in den Bereich der Isomorphieprobleme. Auch die Aufgabe, zu zwei verschiedenen Strukturbeschreibungen festzustellen, ob es sich dabei um dasselbe Molekül handelt, ist ein Isomorphieproblem.

In der Biochemie und Medizin werden auch zunehmend Computer eingesetzt, um Moleküle zu finden, die auf Grund ihrer besonderen Eigenschaften zum Beispiel als Medikamente eingesetzt werden können. Häufig wird ein Stoff gesucht, dessen Struktur ganz speziell auf ein Protein abgestimmt ist, so dass sich dieser Stoff an das Protein anheften kann [BM97, CFM96, And03, VH94]. Der gesuchte Stoff muss an einer oder mehreren Stellen wie ein genaues Gegenstück zu dem entsprechenden Protein passen. Jedes Protein hat mehrere Andockstellen, an die sich der Stoff anheften kann. Zu jeder Andockstelle des Proteins kann das passende Gegenstück berechnet werden. Anschließend kann eine Datenbank daraufhin untersucht werden, ob bereits einen Stoff darin gespeichert wurde, der das zuvor berechnete Gegenstück als Teilstruktur enthält. Die Fragestellung, ob ein gegebener Stoff das gesuchte Gegenstück enthält, das genau zur Andockstelle passt, ist auch wieder ein Isomorphieproblem.

Isomorphieprobleme tauchen jedoch nicht nur in der Chemie auf, sondern kommen in ganz unterschiedlichen Zusammenhängen vor. Es existieren viele Objekte wie zum Beispiel logische Schaltungen in Field Programmable Gate Arrays oder abstrakte Objekte wie zum Beispiel Graphen, für die es schwierig ist, eine eindeutige Darstellung zu finden. Um festzustellen, ob zwei verschiedene Beschreibungen dasselbe Objekt bezeichnen, muss in vielen Fällen ein Isomorphieproblem gelöst werden. Gerade in der Mathematik existieren viele Objekte, bei denen nur unter großem Aufwand festgestellt werden kann, ob zwei verschiedene Beschreibungen dasselbe Objekt bezeichnen. Insbesondere bei der Konstruktion von diskreten Objekten, wie Codes, t -Designs, Graphen, Arks und vielen geometrischen Objekten, müssen meist Isomorphieprobleme gelöst werden [Kur06, KK09, KKK11, Sch92].

Bei der Lösung von Isomorphieproblemen muss vor allen Dingen die Effizienz des verwendeten Algorithmus gewährleistet werden. Denn viele Isomor-

phieprobleme gehören zur Klasse der NP-vollständigen Probleme, die berühmt und berüchtigt sind, weil sie so schwer zu lösen sind. Schwer zu lösen sind diese Probleme nicht deshalb, weil kein Lösungsweg bekannt wäre, sondern weil der Lösungsweg derart umfangreiche Berechnungen erfordert, dass viele Probleme auch auf einem Rechencluster nicht in akzeptabler Zeit durchgeführt werden können. Daher gibt es viele mathematische Fragestellungen, die allein deshalb nicht gelöst sind, weil ein Computer zur Lösung des Problems Wochen, Monate oder sogar Jahre rechnen müsste.

Die meisten professionellen Algorithmen, mit denen sich Isomorphieprobleme lösen lassen, zerlegen ein gegebenes Problem in immer kleinere Teilprobleme, die einzeln leichter zu lösen sind. Die Lösung jedes dieser Teilprobleme kann dann zur Lösung des nächst größeren Problems verwendet werden.

Reinhard Laue war der erste, der diese Strategie auf der Basis von Homomorphismen von Gruppenoperationen formulierte [Lau82]. Sein Homomorphieprinzip kann vermutlich als die Grundlage aller erfolgreicher Algorithmen zur Lösung von Isomorphieproblemen angesehen werden. Denn schon jede Invariante, die zur Klassifikation von Strukturen verwendet wird, kann als ein Homomorphismus von Gruppenoperationen von der Menge der Strukturen in die Menge der Invarianten betrachtet werden. Und jeder Homomorphismus von Gruppenoperationen, bei dem die Bahnen in der Bildmenge alle einelementig sind, kann als Abbildung einer Objektmenge auf eine Menge von Invarianten betrachtet werden. Dieses Homomorphieprinzip wird in Kapitel 3.2 beschrieben.

Ein erstaunliches Beispiel für die Effizienz des Homomorphieprinzips ist der von M. Meringer [Mer99] und T. Grüner [GLM97] entwickelte Graphengenerator *Gradpart*. In einem Graphen wird die Anzahl der Kanten, die an einem Knoten zusammentreffen, als Knotengrad dieses Knotens bezeichnet. Sind die Knotengrade aller Knoten vorgegeben, so können mit dem Programm *Gradpart* alle Graphen konstruiert werden, deren Knoten die entsprechenden Knotengrade besitzen. Genauer gesagt werden die Graphen implizit durch eine Art Konstruktionsanleitung beschrieben, so dass nicht jeder Graph einzeln konstruiert werden muss. Der Graphengenerator *Gradpart* ist so effizient, dass dieser zum Auffinden und Konstruieren eines Graphen im Schnitt weniger als

einen CPU-Takt benötigt.

Eines der bekanntesten Programme zur Lösung von Isomorphieproblemen ist das von B. D. McKay geschriebene Programm *nauty* [McK81]. *Nauty* ist gleichzeitig auch eines der schnellsten Programme zur Lösung von Graphenisomorphieproblemen. Leider ist *nauty* ausschließlich zur Lösung des Graphenisomorphieproblems anwendbar. Daher sehen sich viele Leute gezwungen, die Objekte, für die sie gerne einen Isomorphietest durchführen würden, in Graphen umzuwandeln [Roy98]. Um die Objekte auf Isomorphie testen zu können, muss daher zu jedem einzelnen Objekttyp eine Strategie entworfen werden, wie die zu untersuchenden Objekte so auf Graphen abgebildet werden können, dass deren Struktur erhalten bleibt.

Ein auf Doppelnebenklassen basierender Isomorphietest ist hingegen viel universeller einsetzbar, da das seit Langem bekannte Split Lemma, das in Kapitel 3.5 beschrieben ist, eine sehr vielseitige Methode bereitstellt, um verschiedenste Isomorphieprobleme auf Doppelnebenklassenprobleme abzubilden. Mit Hilfe des Split Lemmas kann beispielsweise auch das Graphenisomorphieproblem auf ein Doppelnebenklassenproblem abgebildet werden. In Satz 6.5.2 wird die Konstruktion einer starken Untergruppenleiter beschrieben, die bei allen Algorithmen dieser Arbeit benötigt wird. Diese starke Untergruppenleiter ermöglicht es, zu den meisten Isomorphieproblemen auf Doppelnebenklassen einen Isomorphietest durchzuführen.

Weiterhin können Doppelnebenklassen zur Lösung vieler gruppentheoretischer Probleme eingesetzt werden, die mit den Algorithmen aus der vorliegenden Arbeit auf einheitliche Weise gelöst werden können. Ein Beispiel für ein schwieriges Problem, das mit Hilfe von Doppelnebenklassen gelöst werden kann, ist die Berechnung der Schnittmenge zweier Untergruppen einer Gruppe.

B. McKay, der bereits zuvor angesprochene Urheber des Programmes *nauty*, gibt in [McK98] einen Überblick über die bekannten Ansätze zur Lösung von Isomorphieproblemen. Dabei wirft er die Frage nach den Zusammenhängen zwischen den verschiedenen Algorithmen zur Lösung von Isomorphieproble-

men auf und grenzt verschiedene Techniken voneinander ab, indem er auf die Unterschiede im Speicherbedarf und die Möglichkeit zur Parallelisierung eingeht.

In der vorliegenden Arbeit wird einer dieser Ansätze, der Leiterspiel-Algorithmus von B. Schmalz [Sch90], mit der Ordnungstreuenerzeugung von R. C. Read [Rea78] und I. Faradžev [Far78] verbunden. Dies ermöglicht es, den Speicherbedarf des Leiterspiel-Algorithmus auf ein Minimum zu reduzieren und bahnt damit den Weg für einen parallelisierbaren Leiterspiel-Algorithmus. Damit geht die vorliegende Arbeit einen Schritt weiter in Richtung einer vereinheitlichten, auf dem Homomorphieprinzip basierenden Beschreibung aller Techniken zur Lösung von Isomorphieproblemen. Als nächsten Schritt sehe ich eine dynamischere Verwendung der Untergruppenleitern an. Statt eine fest vorgegebene Leiter zu verwenden, könnte der nächste „Leiterschritt“ von der Struktur der verwendeten Untergruppen abhängig gemacht werden und die Schrittgröße dem Problem angepasst werden. Um die Verwendung von dynamischen Leitern möglich zu machen, besteht aber noch weiterer Forschungsbedarf.

Kapitel 3

Grundlagen

In diesem Kapitel sollen dem Leser die Grundlagen, auf denen die vorliegende Arbeit aufbaut, näher gebracht werden. Die Sätze und Definitionen dieses Kapitels sind allgemein bekannt und werden daher ohne Beweis und ohne Angabe des Urhebers aufgeführt [Lau93, Hal59].

3.1 Sätze und Definitionen

Definition 1. *Gruppe*

Eine Gruppe sei eine Menge M zusammen mit einer Abbildung $\nu : M \times M \longrightarrow M$, die Verknüpfung genannt wird und für die die folgenden Anforderungen erfüllt sind:

- *Die Abbildung ν ist assoziativ, es gilt daher*
$$\forall a, b, c \in M : \nu(\nu(a, b), c) = \nu(a, \nu(b, c))$$
- *Es existiert ein sogenanntes neutrales Element $e \in M$ mit der Eigenschaft, dass für alle $m \in M$ gilt: $\nu(m, e) = \nu(e, m) = m$. Das neutrale Element wird auch mit 1 oder 1_M bezeichnet, wobei der Index identisch mit der Bezeichnung der Gruppe ist.*
- *Zu jedem $m \in M$ existiert ein sogenanntes inverses Element $m' \in M$ mit der Eigenschaft, dass $\nu(m, m') = \nu(m', m) = e$ ist, wobei e ein neutrales Element bezeichnet. Das zu einem $m \in M$ inverse Element wird auch mit m^{-1} bezeichnet.*

Ist aus dem Zusammenhang klar, welche Gruppenverknüpfung verwendet werden soll oder ist die Gruppenverknüpfung noch nicht genauer spezifiziert, so wird die Gruppe (G, ν) auch häufig nur als Gruppe G angegeben. Für zwei beliebige Elemente $m_1, m_2 \in M$ wird das Bildelement $\nu(m_1, m_2)$ von m_1 und m_2 unter der Abbildung ν auch mit $m_1 \circ m_2$ oder einfach nur mit $m_1 m_2$ bezeichnet. Eine Teilmenge einer Gruppe, die mit der, auf diese Teilmenge eingeschränkten, Gruppenverknüpfung selbst eine Gruppe ist, wird Untergruppe genannt. Um auszudrücken, dass eine Teilmenge U von G eine Untergruppe von G ist, wird auch die Schreibweise $U \leq G$ bzw. für echte Teilmengen U von G auch die Schreibweise $U < G$ verwendet.

Definition 2. Symmetrische Gruppe

Es bezeichne Ω ein n -elementige Menge. Die Menge aller bijektiven Abbildungen von Ω auf sich selbst zusammen mit der Komposition als Gruppenverknüpfung ist eine Gruppe, die Symmetrische Gruppe genannt wird. Die Symmetrische Gruppe auf einer nicht näher spezifizierten n -elementigen Menge wird als S_n bezeichnet.

Definition 3. Nebenklasse

Es sei G eine Gruppe und U eine Untergruppe von G . Für jedes beliebige Element $g \in G$ wird die Menge $\{ug \mid u \in U\}$ mit Ug bezeichnet und Rechtsnebenklasse von U in G genannt. Genauso wird die Menge $\{gu \mid u \in U\}$ mit gU bezeichnet und Linksnebenklasse von U in G genannt. Zwei Rechtsnebenklassen derselben Untergruppe sind entweder gleich oder disjunkt. Gleiches gilt auch für Linksnebenklassen. Die Menge aller Rechtsnebenklassen von U in G wird mit $U \backslash G$ bezeichnet und die Menge aller Linksnebenklassen von U in G wird mit G/U bezeichnet.

Satz 3.1.1. Lagrange

Es sei G eine Gruppe und H eine Untergruppe von G und U eine Untergruppe von H . Für alle Gruppen U und V bezeichne $(V : U)$ den Gruppenindex, das ist die Anzahl der Nebenklassen von U in V . Dann gilt:

$$(G : U) = (G : H) \cdot (H : U)$$

Definition 4. Gruppenhomomorphismus

Es seien (G, \circ) und (H, \bullet) zwei Gruppen und $\varphi : G \longrightarrow H$ eine Abbildung, die

folgende Anforderung erfüllt:

$$\forall g_1, g_2 \in G : \varphi(g_1 \circ g_2) = \varphi(g_1) \bullet \varphi(g_2)$$

Dann wird die Abbildung φ Gruppenhomomorphismus genannt. Ist die Abbildung zusätzlich injektiv, so wird sie auch Gruppenmonomorphismus genannt, ist sie surjektiv, so wird sie auch Gruppenepimorphismus genannt und ist sie bijektiv, so wird sie Gruppenisomorphismus genannt.

Definition 5. Gruppenoperation

Es sei G eine Gruppe, Ω eine Menge und eine Abbildung $\nu : M \times G \longrightarrow M$, die den folgenden beiden Anforderungen genügt:

$$\begin{aligned} \forall g_1, g_2 \in G \forall \omega \in \Omega : \quad & \nu(\omega, g_1 g_2) = \nu(\nu(\omega, g_1), g_2) \\ \forall \omega \in \Omega : \quad & \nu(\omega, 1_G) = \omega \end{aligned}$$

Dann wird das Tupel (G, Ω, ν) eine Gruppenoperation genannt. Existiert zur Gruppe G und der Menge M eine Gruppenoperation (G, M, ν) , so wird auch davon gesprochen, dass die Gruppe G auf der Menge M operiert. Ist aus dem Zusammenhang klar, welche Gruppenoperation gemeint ist, so wird statt $\nu(\omega, g_1 g_2)$ auch die Schreibweise $\omega^{g_1 g_2}$ verwendet.

Definition 6. Stabilisator

Es sei G eine Gruppe, M eine Menge und (G, M, ν) eine Gruppenoperation. Zu jedem Element $\omega \in \Omega$ wird die Menge $\{g \in G \mid \omega^g = \omega\}$ der Stabilisator von ω in G genannt. Diese Menge wird auch mit G_ω bezeichnet.

Definition 7. Homomorphismus von Gruppenoperationen

Es seien G und H zwei Gruppen, Ω und Δ zwei Mengen. Die Gruppe G operiere auf der Menge Ω und die Gruppe H operiere auf der Menge Δ . Es sei $\psi : G \longrightarrow H$ ein Gruppenhomomorphismus und $\varphi : \Omega \longrightarrow \Delta$ eine weitere Abbildung. Für die beiden Abbildungen φ und ψ gelte:

$$\forall g \in G \forall \omega \in \Omega : \quad \varphi(\omega^g) = \varphi(\omega)^{\psi(g)}$$

Dann wird das Abbildungspaar (φ, ψ) ein Homomorphismus von Gruppenoperationen genannt. Sind beide Abbildungen injektiv, so wird (φ, ψ) auch Monomorphismus von Gruppenoperationen genannt und sind beide Abbildungen

surjektiv, so wird (φ, ψ) auch ein *Epimorphismus* von Gruppenoperationen genannt. Sind beide Abbildungen bijektiv, so wird das Abbildungspaar (φ, ψ) ein *Isomorphismus* von Gruppenoperationen genannt.

Definition 8. *G-Homomorphismus*

Es sei G eine Gruppe und Ω und Δ zwei Mengen. Es sei $\psi : G \rightarrow G$ die identische Abbildung und $\varphi : \Omega \rightarrow \Delta$ eine Abbildung. Ist das Paar (φ, ψ) ein Homomorphismus von Gruppenoperationen, so wird das Paar (φ, ψ) auch ein *G-Homomorphismus* genannt. Ist die Abbildung φ zusätzlich bijektiv, so wird das Abbildungspaar (φ, ψ) auch ein *G-Isomorphismus* genannt.

Definition 9. *Bahn*

Es sei Ω eine Menge und G eine Gruppe, die auf der Menge Ω operiert. Für jedes Element $\omega \in \Omega$ wird die Menge $\{\omega^g \mid g \in G\}$ mit ω^G bezeichnet. Die Menge ω^G wird die *Bahn* von ω unter der Operation der Gruppe G genannt. Die Operation der Gruppe G definiert auch eine Äquivalenzrelation auf der Menge Ω :

$$\forall \omega, \omega' \in \Omega : \omega \sim \omega' \iff \omega' \in \omega^G$$

Definition 10. *Block*

Es sei G eine Gruppe, die auf einer Menge Ω operiert. Es sei Δ eine Teilmenge von Ω und für alle $g \in G$ bezeichne Δ^g die Menge $\{\delta^g \mid \delta \in \Delta\}$. Die Menge Δ wird *Block* genannt, wenn folgende Aussage gilt:

$$\forall g \in G : \Delta^g \cap \Delta \neq \emptyset \implies \Delta^g = \Delta$$

Definition 11. *Untergruppenleiter*

Es sei G eine Gruppe und es sei (A_1, \dots, A_n) ein n -Tupel, bei dem jede der n Komponenten eine Untergruppe von G ist. Es soll weiterhin gelten:

$$\begin{aligned} A_1 &= G && \text{und} \\ \forall 1 \leq i < n : & A_i \leq A_{i+1} \quad \vee \quad A_i \geq A_{i+1}. \end{aligned}$$

Dann wird (A_1, \dots, A_n) eine *Untergruppenleiter* zwischen G und A_n oder *Untergruppenleiter* von G nach A_n genannt.

Im weiteren Text werden Untergruppenleitern auch kurz als *Leitern* bezeichnet. Für die in der vorliegenden Arbeit beschriebenen Algorithmen spielen Untergruppenleitern eine zentrale Rolle.

Definition 12. *Partition*

Es sei $\{\lambda_1, \dots, \lambda_m\}$ eine Menge, die aus paarweise disjunkten, nicht leeren Teilmengen der Menge $\{1, \dots, n\}$ besteht. Gilt weiterhin, dass $\bigcup_{i=1}^m \lambda_i = \{1, \dots, n\}$ ist, dann wird $\{\lambda_1, \dots, \lambda_m\}$ auch eine Partition der Menge $\{1, \dots, n\}$ genannt.

Definition 13. *Younggruppe*

Es sei $n > 1$ und $S_n = \langle (1, 2), (1, \dots, n) \rangle$ die Symmetrische Gruppe auf der Menge $\{1, \dots, n\}$ und $\{\lambda_1, \dots, \lambda_m\}$ eine Partition der Menge $\{1, \dots, n\}$. Dann wird die Untergruppe der S_n , die aus dem Stabilisator der Partition $\{\lambda_1, \dots, \lambda_m\}$ besteht, als Younggruppe zur Partition $\{\lambda_1, \dots, \lambda_m\}$ bezeichnet. Die Bezeichnung $S_{(\lambda_1, \dots, \lambda_m)}$ steht im weiteren Text immer für die Younggruppe zur Partition $\{\lambda_1, \dots, \lambda_m\}$.

Definition 14. *Schlichter Graph*

Ein geordnetes Paar (V, E) aus zwei Mengen V und E wird als schlichter oder einfacher Graph bezeichnet, wenn V eine endliche Menge und E eine Teilmenge aller zweielementigen Teilmengen von V ist.

Die Menge E eines einfachen Graphen (V, E) wird als Kantenmenge und jedes Element dieser Menge als Kante bezeichnet. Die Menge V eines einfachen Graphen (V, E) wird dabei als Knotenmenge und jedes Element dieser Menge als Knoten bezeichnet. Zwei Knoten v_1, v_2 eines Graphen (V, E) werden genau dann als benachbart bezeichnet, wenn $\{v_1, v_2\}$ in der Menge der Kanten enthalten ist.

Lemma 1. *Fundamentallemma*

Es sei G eine Gruppe, die auf einer Menge Ω operiert und $\omega \in \Omega$ ein fest gewähltes Element. Dann ist die Abbildung $\varphi_\omega : \omega^G \longrightarrow G_\omega \backslash G$, die jedes Element ω^g aus der Bahn von ω auf die Rechtsnebenklasse $\varphi_\omega(\omega^g) = G_\omega g$ abbildet, ein G -Isomorphismus.

Satz 3.1.2. Es sei G eine Gruppe, die auf einer Menge Ω operiert. Es bezeichne \mathcal{L}_G die Menge $\{U \mid U \leq G\}$ aller Untergruppen von G . Dann operiert die Gruppe G durch Konjugation auf der Menge \mathcal{L}_G . Die Abbildung $\varphi : \Omega \longrightarrow \mathcal{L}_G$, die jedes Element $\omega \in \Omega$ auf seinen Stabilisator G_ω abbildet, ist ein G -Homomorphismus.

Satz 3.1.3. *Es sei G eine Gruppe, die transitiv auf einer Menge Ω operiert. Weiterhin sei Δ ein Block. Dann operiert der Stabilisator G_Δ des Blockes Δ transitiv auf den Elementen des Blockes.*

3.2 Homomorphieprinzip

Das Homomorphieprinzip von R. Laue [Lau82] ist ein sehr mächtiges Werkzeug zur Konstruktion von diskreten Strukturen. Der Graphengenerator GRADPART von T. Grüner [GLM97] und M. Meringer [Mer99], der Graphen zu einer gegebenen Gradpartition erzeugt, verdeutlicht die Stärke einer auf dem Homomorphieprinzip beruhenden Konstruktionsstrategie.

Bei diesem Generator werden die konstruierten Graphen nicht explizit beschrieben, sondern wie bei einem Baukasten aus einzelnen Teilgraphen zusammengesetzt. Kann jeder Graph aus einer Menge \mathcal{A} mit jedem Graphen einer Menge \mathcal{B} auf eindeutige Weise kombiniert werden, so lassen sich durch Kombination der beiden Mengen genau $|\mathcal{A}| \cdot |\mathcal{B}|$ Graphen konstruieren. Die Teilgraphen, die auf diese Weise miteinander kombiniert werden können, lassen sich unter Verwendung des Homomorphieprinzips so schnell berechnen, dass die Konstruktion der gesuchten Graphen im Durchschnitt pro Graph weniger als einen CPU-Takt benötigt.

Im Gegensatz zu anderen Verfahren, für die solche Geschwindigkeiten nicht ungewöhnlich sind, wurde jedoch nicht nur die Anzahl der gesuchten Graphen bestimmt, sondern es wurden auch genaue Konstruktionsanleitungen zu allen diesen Graphen generiert.

Satz 3.2.1. Homomorphieprinzip

Es seien G und H zwei Gruppen und Ω und Δ zwei Mengen. Die Gruppe G operiere auf der Menge Ω und H operiere auf der Menge Δ . Es sei Φ eine surjektive Abbildung $\Phi : \Omega \rightarrow \Delta$ und es sei $\Psi : G \rightarrow H$ ein Gruppenepimorphismus und für die beiden Abbildungen Φ und Ψ gelte weiterhin:

$$\Phi(\omega^g) = \Phi(\omega)^{\Psi(g)} \quad \forall g \in G \forall \omega \in \Omega$$

Dann folgt:

1. $\forall \delta \in \Delta \forall h \in H$: Die Urbildmengen $\Phi^{-1}(\delta)$ und $\Phi^{-1}(\delta^h)$ enthalten Elemente aus denselben Bahnen von G auf Ω .
2. Ist $\omega^g = \omega'$ und ist $\Phi(\omega) = \Phi(\omega')$, so ist $\Psi(g) \in H_{\Phi(\omega)}$.

3. Sind Ω und G endlich und ist $\{\omega_1, \dots, \omega_r\}$ ein Repräsentantensystem aller Bahnen von G auf Ω , so gilt:

$$\frac{1}{|G|} = \frac{1}{|\Omega|} \sum_{i=1}^r \frac{1}{|G_{\omega_i}|}$$

Beweis.

1. $\forall \delta \in \Delta \forall h \in H \forall g \in \Psi^{-1}(h) \forall \omega \in \Omega :$
 $\omega \in \Phi^{-1}(\delta) \implies \Phi(\omega) = \delta \implies \Phi(\omega)^h = \delta^h \implies \Phi(\omega^g) = \delta^h \implies$
 $\omega^g \in \Phi^{-1}(\delta^h)$
2. $\forall \delta \in \Delta \forall \omega_1, \omega_2 \in \Phi^{-1}(\delta) \forall g \in G :$
 $\omega_1^g = \omega_2 \implies \Phi(\omega_1^g) = \Phi(\omega_2) \implies \Phi(\omega_1)^{\Psi(g)} = \Phi(\omega_2) \implies$
 $\delta^{\Psi(g)} = \delta \implies \Psi(g) \in H_\delta$
3. Zu jeder Bahn ω_i^G existiert nach Lemma 1 ein G -Isomorphismus $\varphi : \omega_i^G \rightarrow G_{\omega_i} \backslash G$. Nach Satz 3.1.1 ist die Länge der Bahn ω_i^G daher gleich $|\omega_i^G| = (G : G_{\omega_i}) = \frac{|G|}{|G_{\omega_i}|}$. Da $\{\omega_1, \dots, \omega_r\}$ ein Repräsentantensystem aller Bahnen der Gruppe G auf Ω ist und alle diese Bahnen paarweise disjunkt sind gilt:

$$|\Omega| = \sum_{i=1}^r |\omega_i^G| = \sum_{i=1}^r \frac{|G|}{|G_{\omega_i}|}$$

□

3.3 Ordnungstreue Erzeugung

Die Ordnungstreue Erzeugung ist eine Technik, die es ermöglicht, auf effiziente Weise ein minimales Repräsentantensystem zu einer Menge von Äquivalenzklassen zu konstruieren. Diese Technik wurde 1978 von I. A. Faradžev [Far78] und unabhängig davon auch von R. C. Read [Rea78] gefunden. Die Darstellung der Ordnungstreuen Erzeugung nach Faradžev ist zwar allgemeiner, an dieser Stelle soll aber die Ordnungstreue Erzeugung nach Read beschrieben werden, da diese leichter verständlich ist.

Es seien eine natürliche Zahl n und die Mengen $\Omega_1, \dots, \Omega_n$ gegeben. Auf jeder der Mengen Ω_1 bis Ω_n sei eine Äquivalenzrelation gegeben. Für alle $i \leq n$ und Elemente $\omega_1, \omega_2 \in \Omega_i$ wird die Schreibweise $\omega_1 \sim \omega_2$ verwendet, um zu beschreiben, dass ω_1 und ω_2 zueinander äquivalent sind. Für alle $i \leq n$ und jede Menge Ω_i wird eine Teilmenge von Ω_i als Repräsentantensystem bezeichnet, wenn diese Teilmenge aus jeder der Äquivalenzklassen genau ein Element enthält. Die Elemente eines Repräsentantensystems werden Repräsentanten genannt. Die Ordnungstreue Erzeugung ermöglicht es, zu jeder der Mengen Ω_1 bis Ω_n ein Repräsentantensystem zu konstruieren.

Für alle $i \leq n$ sei eine Abbildung $\chi_i : \Omega_i \rightarrow \{0, 1\}$ gegeben, die genau einem Element aus jeder Äquivalenzklasse den Wert eins und allen anderen Elementen derselben Äquivalenzklasse den Wert null zuweist. Das Repräsentantensystem, bestehend aus allen Elementen der Menge Ω_i , deren Bild unter der Abbildung χ_i gleich eins ist, wird mit \mathcal{L}_i bezeichnet.

Um die Ordnungstreue Erzeugung anwenden zu können, wird für alle $i < n$ eine Methode benötigt, mit Hilfe derer sich aus den Elementen der Menge Ω_i Elemente aus der Menge Ω_{i+1} konstruieren lassen. Dabei soll es durchaus erlaubt sein, aus einem einzigen Element der Menge Ω_i mehrere Elemente der Menge Ω_{i+1} zu erzeugen. Dann kann mit Hilfe einer Abbildung beschrieben werden, welche Elemente aus der Menge Ω_{i+1} aus jedem einzelnen Element der Menge Ω_i konstruiert werden können. Diese Abbildungen werden im Folgenden als Erweiterungsfunktionen bezeichnet. Es bezeichne $\mathcal{P}(\Omega_{i+1})$ die Potenzmenge

der Menge Ω_{i+1} , dann kann die Erweiterungsfunktion als $\Upsilon_i : \Omega_i \longrightarrow \mathcal{P}(\Omega_{i+1})$ definiert werden.

Bedingung 1. *Für alle $i < n$ und jedes Element $\omega_{i+1} \in \mathcal{L}_{i+1}$ existiert ein $\omega_i \in \mathcal{L}_i$, so dass ω_{i+1} in der Menge $\Upsilon_i(\omega_i)$ enthalten ist.*

Zu jeder der Mengen Ω_1 bis Ω_n sei eine totale Ordnungsrelation gegeben. Bei der Ordnungstreuen Erzeugung nach Read werden weniger restriktive Anforderungen an diese Ordnungsrelationen gestellt. In der vorliegenden Arbeit wird die Ordnungstreue Erzeugung jedoch nur in Zusammenhang mit einer Totalordnung verwendet und soll daher auch so beschrieben werden.

Für jedes $i \leq n$ und Elemente $\omega_1, \omega_2 \in \Omega_i$ wird die Schreibweise $\omega_1 \preceq \omega_2$ angewendet, um auszudrücken, dass das Tupel (ω_1, ω_2) in der Ordnungsrelation enthalten ist. Ebenso wird für je zwei Elemente $\omega_1, \omega_2 \in \Omega_i$ die Schreibweise $\omega_1 \prec \omega_2$ angewendet, um auszudrücken, dass das Tupel (ω_1, ω_2) in der Ordnungsrelation enthalten ist und gleichzeitig $\omega_1 \neq \omega_2$ gilt. Für Elemente $\omega_1, \omega_2 \in \Omega_i$ wird im Folgenden ω_1 als kleiner beziehungsweise größer als das Element ω_2 bezeichnet, wenn $\omega_1 \prec \omega_2$ beziehungsweise $\omega_1 \succ \omega_2$ gilt.

Für alle $i < n$ sei eine Abbildung $f_i : \mathcal{L}_{i+1} \longrightarrow \mathcal{L}_i$ wie folgt definiert. Es bezeichne ω_{i+1} ein beliebiges Element aus der Menge \mathcal{L}_{i+1} . Dann soll ω_{i+1} durch die Abbildung f_i auf das kleinste Element ω_i aus der Menge \mathcal{L}_i abgebildet werden, für das gilt, dass ω_{i+1} in der Menge $\Upsilon_i(\omega_i)$ enthalten ist.

Bedingung 2. *Für alle $i < n$ ist die Abbildung f_i monoton steigend, in dem Sinne, dass für alle $\omega_1, \omega_2 \in \mathcal{L}_{i+1}$ mit $\omega_1 \preceq \omega_2$ gilt, dass $f_i(\omega_1) \preceq f_i(\omega_2)$ ist.*

Algorithmus Ordnungstreues Erzeugen

Input: \mathcal{L}_i $\backslash \backslash$ Repräsentantensystem zur Menge Ω_i
 Υ_i $\backslash \backslash$ Erweiterungsfunktion
 \preceq $\backslash \backslash$ Ordnungsrelation auf der Menge Ω_{i+1}
 χ_{i+1} $\backslash \backslash$ Kanonizitätsprädikat auf der Menge Ω_{i+1}
Output: \mathcal{L}_{i+1} $\backslash \backslash$ Repräsentantensystem zur Menge Ω_{i+1}

```

1    $\mathcal{L}_{i+1} \leftarrow \emptyset$ 
2   foreach (  $x \in \mathcal{L}_i$  )                                 $\backslash \backslash$  durchlaufe in aufsteigender Reihenfolge
3       foreach (  $y \in \Upsilon_i(x)$  )                                 $\backslash \backslash$  durchlaufe in aufsteigender Reihenfolge
4           if (  $\emptyset == \mathcal{L}_{i+1}$  or  $y \succ \max(\mathcal{L}_{i+1})$  )
5               if (  $true == \chi_{i+1}(y)$  )
6                    $\mathcal{L}_{i+1} \leftarrow \mathcal{L}_{i+1} \cup \{y\}$ 
7               endif
8           endif
9       end
10  end

```

Im Pseudocode des Algorithmus zur Ordnungstreuen Erzeugung sollen in Zeile 2 die Elemente der Menge \mathcal{L}_i in aufsteigender Reihenfolge entsprechend der gegebenen Ordnungsrelation durchlaufen werden. Genauso sollen auch in Zeile 3 für alle Elemente $x \in \mathcal{L}_i$ die Elemente der Menge $\Upsilon_i(x)$ in aufsteigender Reihenfolge durchlaufen werden. Sind diese beiden Voraussetzungen erfüllt und sind die Bedingungen 1 und 2 der Ordnungstreuen Erzeugung erfüllt, so berechnet der Algorithmus für alle $i < n$ aus dem Repräsentantensystem \mathcal{L}_i das Repräsentantensystem \mathcal{L}_{i+1} . Die Korrektheit dieses Algorithmus wird in [Rea78] nachgewiesen.

3.4 Doppelnebenklassen

Es bezeichne G eine Gruppe und U eine Untergruppe von G . Zu jedem $g \in G$ wird die Menge $Ug = \{ug \mid u \in U\}$ als Rechtsnebenklasse der Gruppe U in der Gruppe G bezeichnet. Zwei Rechtsnebenklassen Ug_1 und Ug_2 sind entweder identisch oder disjunkt:

$$Ug_1 \cap Ug_2 \neq \emptyset \iff \exists u \in U : ug_1 = g_2 \iff \exists u \in U : Ug_1 = Ugug_1 = Ug_2$$

Die Menge aller Rechtsnebenklassen von U in G wird mit $U \backslash G$ bezeichnet. Im weiteren Textverlauf werden Rechtsnebenklassen auch kurz als Nebenklassen bezeichnet.

Es sei B eine Untergruppe von G und $\nu : U \backslash G \times B \rightarrow U \backslash G$ eine Abbildung mit $\nu(Ug, b) = Ugb$ für alle $g \in G$ und $b \in B$. Diese Abbildung ist eine Gruppenoperation, denn für alle $g \in G$ gilt $\nu(Ug, 1_B) = Ug$ und es gilt weiterhin:

$$\forall g \in G \forall b_1, b_2 \in B : \quad \nu(\nu(Ug, b_1), b_2) = Ugb_1b_2 = \nu(Ug, b_1b_2)$$

Es sei V eine weitere Untergruppe von G , mit $U \leq V$. Die Gruppe B operiert natürlich genauso durch Rechtsmultiplikation auf der Menge $V \backslash G$. Es bezeichne ν' die Gruppenoperation auf der Menge $V \backslash G$.

Es sei $\varphi : U \backslash G \rightarrow V \backslash G$ eine Abbildung mit $\varphi(Ug) = Vg$ für alle $g \in G$. Dann ist die Abbildung φ ein B -Homomorphismus, denn für alle $g \in G$ und $b \in B$ gilt:

$$\varphi(\nu(Ug, b)) = \varphi(Ugb) = Vgb = \nu'(Vg, b) = \nu'(\varphi(Ug), b)$$

Auf diese Weise ergibt sich eine Fülle von B -Homomorphismen zwischen den Nebenklassenmengen von verschiedenen Untergruppen von G .

Es bezeichne (A_1, \dots, A_n) eine Untergruppenleiter mit $A_1 = G$ und $A_i \leq G$ für alle $i \leq n$. Dann existiert zu dieser Leiter eine Menge von $n - 1$ verschiedenen B -Homomorphismen:

Für alle $i < n$ mit $A_i \leq A_{i+1}$ ist die Abbildung $\varphi_i : A_i \backslash G \rightarrow A_{i+1} \backslash G$ mit $\varphi_i(A_i g) = A_{i+1} g$ ein B -Homomorphismus.

Genauso ist auch für alle $i < n$, mit $A_{i+1} \leq A_i$ die Abbildung $\varphi_i : A_{i+1} \backslash G \rightarrow A_i \backslash G$ mit $\varphi_i(A_{i+1} g) = A_i g$ ein B -Homomorphismus.

Diese Menge von B -Homomorphismen kann dazu verwendet werden für alle $i \leq n$ die Bahnen und Stabilisatoren aller Nebenklassen aus der Menge $A_i \backslash G$ unter der Operation der Gruppe B zu bestimmen.

Bezeichnet $\nu : U \backslash G \times B \longrightarrow U \backslash G$ die Operation einer Gruppe B auf der Nebenklassenmenge $U \backslash G$ dann wird im weiteren Text die Operation eines Elementes $b \in B$ auf einer Nebenklasse $Ug \in U \backslash G$ nicht mehr als $\nu(Ug, b)$ sondern kurz als Ugb geschrieben. Operiert die Gruppe B auf der Menge $U \backslash G$, so wird die Bahn $\{Ugb \mid b \in B\}$ der Nebenklasse Ug im weiteren Text mit Ug^B bezeichnet.

Die Vereinigungsmenge $\bigcup_{b \in B} Ugb = \{ugb \mid u \in U, b \in B\}$ aller Nebenklassen einer Bahn wird Doppelnebenklasse genannt. Die Doppelnebenklasse $\bigcup_{b \in B} Ugb$ wird im weiteren Text mit UgB bezeichnet. Die Menge aller Doppelnebenklassen von U und B in der Gruppe G wird mit $U \backslash G / B = \{UgB \mid g \in G\}$ bezeichnet.

Die Betrachtung von Doppelnebenklassen ist von besonderem Interesse, da sich viele Probleme aus der Gruppentheorie mit Hilfe von Doppelnebenklassen lösen lassen. Dazu zählen schwierige Isomorphieprobleme, wie dies in den folgenden Kapiteln erläutert wird, aber auch die Berechnung der Schnittmenge zweier, nur durch ihre Erzeuger gegebener Untergruppen von G .

Lemma 2. *Es seien A und B zwei Untergruppen von G . Die Gruppe B operiere durch Rechtsmultiplikation auf der Menge $A \backslash G$.*

Der Stabilisator B_{Ag} einer beliebigen Nebenklasse $Ag \in A \backslash G$ unter der Operation der Gruppe B ist identisch mit der Schnittmenge $g^{-1}Ag \cap B$.

Beweis. Die Schnittmenge $g^{-1}Ag \cap B$ ist eine Teilmenge des Stabilisators B_{Ag} , denn für alle $c \in g^{-1}Ag \cap B$ gilt $gcg^{-1} \in A$ und daraus folgt:

$$Ag^c = Agc = Agcg^{-1}g = Ag$$

Umgekehrt muss für jedes Element c aus dem Stabilisator B_{Ag} gelten, dass $Agc = Ag$ ist. Daraus folgt $g^{-1}Agc = g^{-1}Ag$ und daher gilt auch $c \in g^{-1}Ag$.

□

Aus Lemma 2 folgt auch, dass der Stabilisator $B_A = \{b \in B \mid Ab = A\}$ der Nebenklasse A identisch ist zur Schnittmenge der beiden Gruppen A und B .

Mit Hilfe der in den folgenden Kapiteln vorgestellten Algorithmen kann daher auch die Schnittmenge $B \cap A$ berechnet werden. Der Zusammenhang zwischen dem Stabilisator einer Nebenklasse und der Schnittmenge zweier Gruppen wird in den folgenden Kapiteln eine wichtige Rolle spielen.

3.5 Split Lemma

Das Split Lemma beschreibt eine universell einsetzbare Abbildung, anhand derer Isomorphieprobleme in unterschiedlichsten Objektmengen in Isomorphieprobleme von Nebenklassen umgewandelt werden können. Der Urheber des Split Lemmas ist unbekannt.

Unter Anwendung des Split Lemmas können unterschiedlichste Isomorphieprobleme, wie zum Beispiel das Graphenisomorphieproblem, in ein Isomorphieproblem auf Nebenklassen umgewandelt werden, das mit den in den folgenden Kapiteln vorgestellten Algorithmen gelöst werden kann. Auf diese Weise kann derselbe Algorithmus zur Lösung von sehr unterschiedlichen Isomorphieproblemen genutzt werden.

Satz 3.5.1. Split Lemma

Eine Gruppe G operiere transitiv auf einer Menge Ω . Es sei B eine Untergruppe von G und ω ein fest gewähltes Element aus Ω . Der Stabilisator von ω in G wird mit G_ω bezeichnet. Die Menge aller Bahnen von B auf Ω wird mit $\Omega \setminus B$ bezeichnet.

Dann existiert eine bijektive Abbildung $\rho : \Omega \setminus B \longrightarrow G_\omega \setminus G/B$, die für alle $g \in G$ die Bahn ω^{gB} auf die Doppelnebenklasse $G_\omega gB$ abbildet.

Beweis. Die Gruppe B operiert durch Rechtsmultiplikation auf der Menge $G_\omega \setminus G$, der Menge der Rechtsnebenklassen von G_ω in G . Nach Lemma 1 ist die Abbildung $\psi : \omega^G \longrightarrow G_\omega \setminus G$, $\psi : \omega^g \longmapsto G_\omega g$ für jedes fest gewählte $\omega \in \Omega$ ein G -Isomorphismus. Da G transitiv auf der Menge Ω operiert, gilt $\omega^G = \Omega$. Da B eine Untergruppe von G ist, operiert B sowohl auf der Menge Ω als auch auf der Menge $G_\omega \setminus G$. Da ψ ein G -Isomorphismus ist, werden nach dem Homomorphieprinzip 3.2.1 zwei Elemente aus der Menge $\Omega \setminus B$ genau dann auf Elemente aus derselben Bahn abgebildet, wenn diese zwei Elemente der Menge $\Omega \setminus B$ bereits in derselben Bahn lagen.

□

Beispiel

In Abbildung 3.1 sind zwei schlichte Graphen dargestellt. Es stellt sich die Frage, ob es möglich ist, die Nummerierung der Ecken des Graphen \mathcal{A} so zu vertauschen, dass dieser identisch zum Graphen \mathcal{B} wird.

Viele Isomorphieprobleme, wie das Graphenisomorphieproblem aus diesem Beispiel, können mit Hilfe des Split Lemmas auf Isomorphieprobleme von Doppelnebenklassen abgebildet und dann gelöst werden. Auf diese Weise kann derselbe Algorithmus zur Lösung von ganz unterschiedlichen Isomorphieproblemen eingesetzt werden.

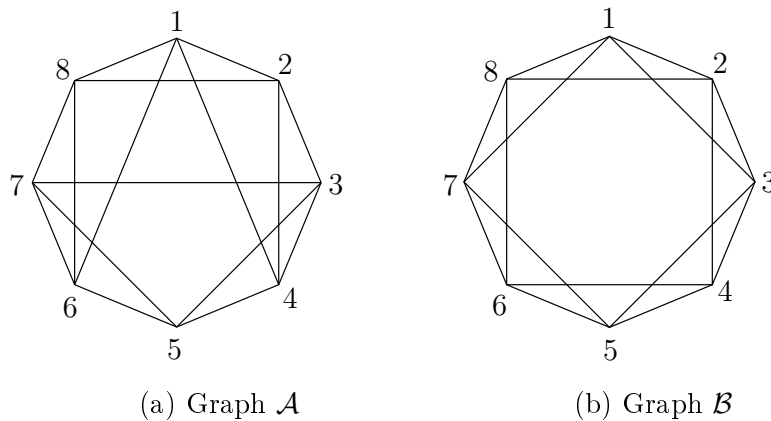


Abbildung 3.1: Zwei schlichte Graphen auf acht Knoten

Die Menge aller schlichten Graphen, die aus acht Knoten und sechzehn Kanten bestehen, soll als Ω bezeichnet werden. Die beiden zu untersuchenden Graphen \mathcal{A} und \mathcal{B} sind also in der Menge Ω enthalten. Alle Graphen dieser Menge haben dieselbe Knotenmenge und zwar die Menge $\{1, \dots, 8\}$. Jede Kante eines dieser Graphen wird durch eine zweielementige Teilmenge der Knotenmenge beschrieben.

Zuerst einmal soll exakt beschrieben werden, wie genau die Nummerierung der Ecken eines Graphen geändert werden kann. Die Gruppe S_8 , welche Symmetrische Gruppe genannt wird, bezeichnet die Gruppe aller möglichen Permutationen von acht Punkten. Diese Gruppe operiert auf der Knotenmenge $\{1, \dots, 8\}$ und damit auch auf der Menge aller zweielementigen Teilmengen

der Knotenmenge. Die Menge $\{2, 5\}$ wird beispielsweise durch die Permutation $g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 1 & 5 & 7 & 6 & 4 & 8 \end{pmatrix}$ auf die Menge $\{2, 5\}^g = \{2^g, 5^g\} = \{3, 7\}$ abgebildet. Die Menge aller zweielementigen Teilmengen der Menge $\{1, \dots, 8\}$ wird mit K bezeichnet. Die S_8 operiert auch auf der Menge aller Teilmengen von K , indem jede Menge $\Delta = \{\delta_1, \dots, \delta_m\}$ durch jedes Element $g \in S_8$ auf die Menge $\Delta^g = \{\delta_1^g, \dots, \delta_m^g\}$ abgebildet wird.

Zu jeder 16-elementigen Teilmenge von K existiert genau ein Graph in der Menge Ω , dessen Kantenmenge aus diesen 16 Kanten besteht. Da sich die Graphen der Menge Ω nur durch ihre Kanten unterscheiden, kann aus der Operation der S_8 auf der Menge der Teilmengen von K eine Operation der S_8 auf der Menge Ω abgeleitet werden. Die Menge Ω ist abgeschlossen unter dieser Operation.

Die Menge $\{\{1, 2\}, \{1, 4\}, \{1, 6\}, \{1, 8\}, \{2, 3\}, \{2, 4\}, \{2, 8\}, \{3, 4\}, \{3, 5\}, \{3, 7\}, \{4, 5\}, \{5, 6\}, \{5, 7\}, \{6, 7\}, \{6, 8\}, \{7, 8\}\}$ ist die Kantenmenge des Graphen \mathcal{A} . Dieser Graph wird zum Beispiel durch die Permutation $g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 1 & 5 & 7 & 6 & 4 & 8 \end{pmatrix}$ auf den Graphen mit der Kantenmenge $\{\{2, 3\}, \{2, 5\}, \{2, 6\}, \{2, 8\}, \{3, 1\}, \{3, 5\}, \{3, 8\}, \{1, 5\}, \{1, 7\}, \{1, 4\}, \{5, 7\}, \{7, 6\}, \{7, 4\}, \{6, 4\}, \{6, 8\}, \{4, 8\}\}$ abgebildet.

Diese Operation der Gruppe S_8 auf der Menge Ω wurde in der ursprünglichen Fragestellung als Umnummerierung bezeichnet. Daher kann die ursprüngliche Fragestellung wie folgt umformuliert werden: Liegen die beiden Graphen \mathcal{A} und \mathcal{B} in derselben Bahn unter der Operation der S_8 ?

Um das ursprüngliche Graphenisomorphieproblem mit Hilfe des Split Lemmas auf ein Doppelnebenklassenproblem abbilden zu können, wird eine Gruppe benötigt, die transitiv auf der Menge Ω operiert. Die zuvor beschriebene Operation der S_8 auf der Menge Ω ist jedoch nicht transitiv. Die Menge K besteht aus den $\binom{8}{2} = 28$ verschiedenen zweielementigen Teilmengen der Menge $\{1, \dots, 8\}$.

Die Elemente der Menge K werden mit folgenden Bezeichnungen versehen:

$$\begin{aligned}
 i_{12} &= \{1, 2\} & i_{13} &= \{1, 3\} & i_{14} &= \{1, 4\} & i_{15} &= \{1, 5\} & i_{16} &= \{1, 6\} \\
 i_{17} &= \{1, 7\} & i_{18} &= \{1, 8\} & i_{23} &= \{2, 3\} & i_{24} &= \{2, 4\} & i_{25} &= \{2, 5\} \\
 i_{26} &= \{2, 6\} & i_{27} &= \{2, 7\} & i_{28} &= \{2, 8\} & i_{34} &= \{3, 4\} & i_{35} &= \{3, 5\} \\
 i_{36} &= \{3, 6\} & i_{37} &= \{3, 7\} & i_{38} &= \{3, 8\} & i_{45} &= \{4, 5\} & i_{46} &= \{4, 6\} \\
 i_{47} &= \{4, 7\} & i_{48} &= \{4, 8\} & i_{56} &= \{5, 6\} & i_{57} &= \{5, 7\} & i_{58} &= \{5, 8\} \\
 i_{67} &= \{6, 7\} & i_{68} &= \{6, 8\} & i_{78} &= \{7, 8\}
 \end{aligned}$$

Die S_{28} , die Symmetrische Gruppe auf den 28 Elementen der Menge K operiert auf natürliche Weise auf den 28 Elementen der Menge K . Diese Gruppe kann auch als die Menge aller bijektiven Abbildungen der Menge K auf sich selbst aufgefasst werden. Die S_{28} operiert auch auf der Menge aller Teilmengen von K , indem jede Menge $\Delta = \{\delta_1, \dots, \delta_m\} \subseteq K$ durch jedes Element $g \in S_{28}$ auf die Menge $\Delta^g = \{\delta_1^g, \dots, \delta_m^g\}$ abgebildet wird. Auf diese Weise operiert die S_{28} auch auf der Menge Ω und diese Operation ist sogar transitiv. In Anlehnung an die Bezeichnungen des Split Lemmas wird die Gruppe S_{28} im weiteren Text mit G bezeichnet.

Wie zuvor beschrieben, operiert auch die S_8 auf der Menge K . Es bezeichne $\nu : K \times S_8 \rightarrow K$ die entsprechende Abbildung zur Gruppenoperation der S_8 auf der Menge K . Wird die Gruppenoperation ν auf die Operation eines einzelnen Gruppenelementes g eingeschränkt, so ergibt sich eine bijektive Abbildung von der Menge K auf sich selbst:

$$\nu_g : K \rightarrow K, \quad \nu_g(k) = \nu(k, g) \quad \forall k \in K$$

Da die S_{28} auch als Gruppe aller bijektiven Abbildungen der Menge K auf sich selbst aufgefasst werden kann, entspricht die Abbildung ν_g dann wiederum einem Element der Gruppe G .

Dieser Zusammenhang kann wiederum durch eine Abbildung $\varphi : S_8 \rightarrow S_{28}$ mit $\varphi(g) = \nu_g \in S_{28}$ beschrieben werden, für die Folgendes gilt:

$$\forall \{i, j\} \in K \quad \forall h \in S_8 : \quad \{i, j\}^h = \{\nu(i, h), \nu(j, h)\} = \{i, j\}^{\varphi(h)}$$

Da die S_8 durch die beiden Permutationen $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 \end{pmatrix}$ und $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 1 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}$ erzeugt wird, kann der Gruppenmonomorphismus φ eindeutig durch die Bilder dieser beiden Permutationen beschrieben werden:

$$\varphi\left(\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 \end{pmatrix}\right) =$$

$$\begin{pmatrix} i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} & i_{45} & i_{46} & i_{47} & i_{48} & i_{56} & i_{57} & i_{58} & i_{67} & i_{68} & i_{78} \\ i_{12} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} & i_{45} & i_{46} & i_{47} & i_{48} & i_{56} & i_{57} & i_{58} & i_{67} & i_{68} & i_{78} \end{pmatrix}$$

$$\varphi\left(\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 \end{pmatrix}\right) =$$

$$\begin{pmatrix} i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} & i_{45} & i_{46} & i_{47} & i_{48} & i_{56} & i_{57} & i_{58} & i_{67} & i_{68} & i_{78} \\ i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} & i_{12} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} & i_{13} & i_{45} & i_{46} & i_{47} & i_{48} & i_{14} & i_{56} & i_{57} & i_{58} & i_{15} & i_{67} & i_{68} & i_{16} & i_{78} & i_{17} & i_{18} \end{pmatrix}$$

Die Bildmenge der S_8 unter dem Gruppenmonomorphismus φ ist eine Untergruppe von G , die im Folgenden mit B bezeichnet wird. Diese Gruppe B wird von den beiden Gruppenelementen $\varphi\left(\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 \end{pmatrix}\right)$ und $\varphi\left(\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 1 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}\right)$ erzeugt. Um das Split Lemma anwenden zu können muss jetzt nur noch ein Element ω aus der Menge Ω ausgewählt und dessen Stabilisator unter der Operation der Gruppe G berechnet werden. Der Graph aus der Menge Ω mit der Kantenmenge

$$\{i_{12}, i_{13}, i_{14}, i_{15}, i_{16}, i_{17}, i_{18}, i_{23}, i_{24}, i_{25}, i_{26}, i_{27}, i_{28}, i_{34}, i_{35}, i_{36}\}$$

soll mit ω bezeichnet werden. Der Stabilisator G_ω von ω in G ist identisch mit dem mengenweisen Stabilisator der Kantenmenge von ω in G .

Aufgrund der Transitivität der Operation der Gruppe G auf der Menge Ω kann der Graph \mathcal{A} als ω^{g_1} geschrieben werden, wobei $g_1 =$

$$\begin{pmatrix} i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} & i_{45} & i_{46} & i_{47} & i_{48} & i_{56} & i_{57} & i_{58} & i_{67} & i_{68} & i_{78} \\ i_{12} & i_{14} & i_{16} & i_{18} & i_{23} & i_{24} & i_{28} & i_{34} & i_{35} & i_{37} & i_{45} & i_{56} & i_{57} & i_{67} & i_{68} & i_{78} & i_{13} & i_{15} & i_{17} & i_{25} & i_{26} & i_{27} & i_{36} & i_{38} & i_{46} & i_{47} & i_{48} & i_{58} \end{pmatrix}$$

ist. Ähnliches gilt für den Graphen \mathcal{B} , der identisch mit dem Graphen ω^{g_2} ist, wobei $g_2 =$

$$\begin{pmatrix} i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} & i_{45} & i_{46} & i_{47} & i_{48} & i_{56} & i_{57} & i_{58} & i_{67} & i_{68} & i_{78} \\ i_{12} & i_{13} & i_{17} & i_{18} & i_{23} & i_{24} & i_{28} & i_{34} & i_{35} & i_{45} & i_{46} & i_{56} & i_{57} & i_{67} & i_{68} & i_{78} & i_{14} & i_{15} & i_{16} & i_{25} & i_{26} & i_{27} & i_{36} & i_{37} & i_{38} & i_{47} & i_{48} & i_{58} \end{pmatrix}$$

ist.

Nach dem Split Lemma wird die Bahn des Graphen \mathcal{A} durch die in Satz 3.5.1 definierte Abbildung ρ auf die Doppelnebenklasse $G_\omega g_1 B$ abgebildet. Die Bahn des Graphen \mathcal{B} wird auf die Doppelnebenklasse $G_\omega g_2 B$ abgebildet. Die beiden Doppelnebenklassen $G_\omega g_1 B$ und $G_\omega g_2 B$ sind genau dann identisch, wenn ein $b \in B$ existiert, so dass $\omega^{g_1 b} = \omega^{g_2}$ ist. Dieses Element b kann mit Hilfe der Algorithmen, die in den folgenden Kapiteln beschrieben werden, gefunden werden.

Auf diese Weise kann die Permutation $b =$

$$\begin{pmatrix} i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} & i_{45} & i_{46} & i_{47} & i_{48} & i_{56} & i_{57} & i_{58} & i_{67} & i_{68} & i_{78} \\ i_{12} & i_{14} & i_{13} & i_{15} & i_{17} & i_{16} & i_{18} & i_{24} & i_{23} & i_{25} & i_{27} & i_{26} & i_{28} & i_{34} & i_{45} & i_{47} & i_{46} & i_{48} & i_{35} & i_{37} & i_{36} & i_{38} & i_{57} & i_{56} & i_{58} & i_{67} & i_{78} & i_{68} \end{pmatrix}$$

aus der Gruppe B gefunden werden, für die $G_\omega g_1 b = G_\omega g_2$ gilt. Das Urbild dieser Permutation B unter dem Gruppenhomomorphismus φ ist die Permutation $b' = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 4 & 3 & 5 & 7 & 6 & 8 \end{pmatrix}$.

Die ursprüngliche Fragestellung kann daher mit „ja“ beantwortet werden, denn der Graph \mathcal{A} wird unter der Operation des Gruppenelementes b' auf den Graphen \mathcal{B} abgebildet.

Kapitel 4

Das Leiterspiel nach Schmalz

Das Leiterspiel ist eine von Bernd Schmalz [Sch90] entwickelte Technik, die es ermöglicht, viele diskrete Strukturen und insbesondere auch Doppelnebenklassen auf äußerst effiziente Weise zu konstruieren. Seine Stärke verleiht dem Leiterspiel das von Reinhard Laue [Lau82] entwickelte Homomorphieprinzip 3.2.1, das Homomorphismen von Gruppenoperationen dazu benutzt, schwierige Problemstellungen auf einfachere zurückzuführen. Seine beeindruckende Effizienz hat dieser Ansatz längst unter Beweis gestellt.

In den sogenannten Splittingschritten des Leiterspiels werden, auf Basis des Homomorphieprinzips, die Lösungen eines einfacheren Problems zu Lösungen von schwierigeren Problemen verfeinert (splitting orbits). Eine Besonderheit des Leiterspiels im Unterschied zu früheren Ansätzen ist die Verwendung von sogenannten Fusingschritten, in denen Bahnen vereinigt (fusing orbits) werden, indem Homomorphismen in der umgekehrten zu der sonst üblichen Richtung verwendet werden. Diese scheinen das zu lösende Problem auf den ersten Blick zu vergrößern, eröffnen jedoch neue Möglichkeiten sich der Lösung anzunähern. Spätestens wenn von der betrachteten Menge aus keine weiteren Splitting-Homomorphismen mehr existieren, die eine direkte Annäherung an die Lösung ermöglichen würden, wird diese umgekehrte Verwendung von Homomorphismen besonders wertvoll.

Ein Beispiel für eine Situation, in der dieses Problem auftritt, ist die folgende. Es seien A und B zwei Untergruppen zu einer gegebenen Gruppe G . Zu diesen Gruppen sollen die Doppelnebenklassen $A \backslash G / B$ schrittweise be-

rechnet werden. Ist die Gruppe A eine maximale Untergruppe in G , so existiert zu keiner echten Untergruppe $C < G$ ein G -Homomorphismus der Form $\varphi : A \backslash G \longrightarrow C \backslash G$, mit $\varphi(Ag) = Cg$. Daher existiert zu keiner nicht trivialen Menge von Doppelnebenklassen ein geeigneter G -Homomorphismus, mit Hilfe dessen die gesuchte Menge von Doppelnebenklassen in einem Splittingschritt konstruiert werden könnte. Diese Situation wird beim Leiterspiel gelöst, indem statt eines Splittingschrittes ein Fusingschritt zur weiteren Konstruktion der gesuchten Doppelnebenklassen eingesetzt wird.

In diesem Kapitel wird der Leiterspiel-Algorithmus von Schmalz vorgestellt, wobei die Beschreibung angelehnt ist an die Darstellung in [Lau93]. Dieser Algorithmus ist dazu geeignet, ein minimales Repräsentantensystem aller Bahnen einer Gruppe B auf einer Menge Ω_n zu berechnen.

Ist die Menge Ω_n zum Beispiel eine Menge von Rechtsnebenklassen, auf der die Gruppe B durch Rechtsmultiplikation operiert, so wird mit dem Leiterspiel-Algorithmus ein minimales Repräsentantensystem aller Doppelnebenklassen konstruiert. Die Vertreter dieses minimalen Repräsentantensystems werden in Abgrenzung zu den übrigen Bahnelementen auch als kanonische Bahnrepräsentanten bezeichnet. Weiterhin werden zu allen Bahnrepräsentanten die zugehörigen Stabilisatoren in der Gruppe B berechnet. Der Algorithmus konstruiert auch eine Abbildung $\eta_n : \Omega_n \longrightarrow B$ mit der Eigenschaft, dass für alle $\omega \in \Omega_n$ das Element $\omega^{\eta_n(\omega)}$ in der Menge der kanonischen Bahnrepräsentanten enthalten ist. Diese Abbildung wird fusionierende Abbildung genannt.

4.1 Voraussetzungen

Es sei B eine Gruppe und $(\Omega_1, \dots, \Omega_n)$ ein n -Tupel von Mengen mit der Eigenschaft, dass die Gruppe B auf jeder der Mengen $\Omega_1, \dots, \Omega_n$ operiert. Für je zwei aufeinander folgenden Mengen Ω_i und Ω_{i+1} des Tupels $(\Omega_1, \dots, \Omega_n)$ soll zusätzlich gelten, dass entweder ein B -Homomorphismus $\varphi_i : \Omega_{i+1} \longrightarrow \Omega_i$ oder ein surjektiver B -Homomorphismus $\varphi_i : \Omega_i \longrightarrow \Omega_{i+1}$ existiert. Mit Hilfe dieser B -Homomorphismen werden iterativ aus den Repräsentanten T_i der Bahnen von B auf der Menge Ω_i die Repräsentanten T_{i+1} der Bahnen von B auf der Menge Ω_{i+1} berechnet. Dies geschieht unter Verwendung der Stabilisatoren

der Repräsentanten aus der Menge T_i und einer sogenannten fusionierenden Abbildung $\eta_i : \Omega_i \longrightarrow B$. Weiterhin wird zu jedem Repräsentanten $t \in T_{i+1}$ auch der Stabilisator B_t in der Gruppe B und die fusionierende Abbildung $\eta_{i+1} : \Omega_{i+1} \longrightarrow B$ berechnet. Die Menge der Stabilisatoren und die fusionierende Abbildung bilden wiederum die Grundlage für den darauf folgenden Iterationsschritt.

Wenn zur Konstruktion des Repräsentantensystems T_{i+1} ein B -Homomorphismus $\varphi_i : \Omega_i \longrightarrow \Omega_{i+1}$ verwendet wird, so wird dieser Konstruktionsschritt als Fusingschritt bezeichnet. Wenn zur Konstruktion des Repräsentantensystems T_{i+1} ein B -Homomorphismus $\varphi_i : \Omega_{i+1} \longrightarrow \Omega_i$ verwendet wird, so wird der entsprechende Konstruktionsschritt als Splittingschritt bezeichnet.

Zum Starten der Iteration wird das Repräsentantensystem T_1 zusammen mit der fusionierenden Abbildung η_1 und den Stabilisatoren aller Elemente der Menge T_1 als bekannt vorausgesetzt.

4.2 Fusionierende Abbildung

Die fusionierenden Abbildungen ermöglichen es, für alle $i \leq n$ und zu jedem Element $\omega \in \Omega_i$ den kanonischen Repräsentanten der Bahn dieses Elementes zu bestimmen. In jedem Fusing- oder Splittingschritt von Ω_{i-1} nach Ω_i soll neben der Menge T_i und der Menge der Stabilisatoren auch eine fusionierende Abbildung $\eta_i : \Omega_i \longrightarrow B$ berechnet werden. Diese fusionierende Abbildung η_i ordnet jedem Element $\omega \in \Omega_i$ ein fusionierendes Element aus der Menge B zu. Jedes dieser fusionierenden Elemente $\eta_i(\omega)$ hat die besondere Eigenschaft, dass $\omega^{\eta_i(\omega)}$ in der Menge T_i liegt.

Da der Definitionsbereich der fusionierenden Abbildungen mitunter sehr groß ist, würde bei direkter Speicherung aller Paare von Urbild- und Bildelementen sehr viel Speicher benötigt werden. Daher soll die fusionierende Abbildung η_i immer aus einer Menge von Abbildungen f_1, \dots, f_i berechnet werden, die einen geringen Speicherbedarf haben. Diese Abbildungen f_1, \dots, f_i werden wie folgt definiert:

Die Abbildung $f_1 : \Omega_1 \longrightarrow B$ sei identisch zu der nach Voraussetzung gege-

benen fusionierenden Abbildung η_1 . Ist der Konstruktionsschritt, in dem die Menge T_i konstruiert wird, ein Fusingschritt, so ist die Abbildung f_i auf der Menge $\{\varphi_{i-1}(t) \mid t \in T_{i-1}\}$ definiert, wobei φ_{i-1} den B -Homomorphismus von Ω_{i-1} nach Ω_i bezeichnet. Ist der Konstruktionsschritt, in dem die Menge T_i konstruiert wird, ein Splittingschritt, so ist die Abbildung f_i auf der Menge $\{\omega \in \Omega_i \mid \varphi_{i-1}(\omega) \in T_{i-1}\}$ definiert, wobei φ_{i-1} den B -Homomorphismus von Ω_i nach Ω_{i-1} bezeichnet. Jede der Abbildungen f_i entspricht einer auf den entsprechenden Definitionsbereich eingeschränkten fusionierenden Abbildung η_i . Die Abbildungen f_1, \dots, f_i können dann schrittweise zusammen mit den kanonischen Repräsentanten der Mengen T_1, \dots, T_i berechnet werden.

Sind für ein $1 < i \leq k$ die Abbildungen f_1 bis f_i bekannt und soll zu einem gegebenen Element $\omega \in \Omega_i$ das fusionierende Element berechnet werden, so kann dieses wie folgt rekursiv bestimmt werden:

Ist der Konstruktionsschritt, in dem die Menge T_i bestimmt wurde, ein Splittingschritt, so wird zuerst das Element $\delta = \varphi_{i-1}(\omega)$ berechnet. Ist der Konstruktionsschritt, in dem die Menge T_i bestimmt wurde, hingegen ein Fusingsschritt, so wird zuerst eines der Elemente $\delta \in \Omega_{i-1}$ mit $\varphi_{i-1}(\delta) = \omega$ bestimmt. Anschließend wird das Element $b_1 = \eta_{i-1}(\delta) \in B$ berechnet. Das Element ω^{b_1} liegt im Definitionsbereich der Funktion f_i , denn δ^{b_1} ist in der Menge T_{i-1} enthalten. Daher kann das Element $b_2 = f_i(\omega^{b_1}) \in B$ berechnet werden. Dann ist das Element $b_1 b_2$ ein fusionierendes Element zum Element ω , denn $b_1 b_2$ liegt in der Gruppe B und $\omega^{b_1 b_2}$ in der Menge T_i . Auf diese Weise kann zu allen $\omega \in \Omega_i$ die fusionierende Abbildung η_i bestimmt werden.

4.3 Splitting Orbits

Existiert für ein $i < n$ ein B -Homomorphismus $\varphi_i : \Omega_{i+1} \longrightarrow \Omega_i$, so kann die Menge T_{i+1} in einem Splittingschritt aus der Menge T_i berechnet werden. Weiterhin kann eine fusionierende Abbildung $\eta_{i+1} : \Omega \longrightarrow B$ bestimmt und zu jedem Element $t \in T_{i+1}$ der Stabilisator B_t berechnet werden.

4.3.1 Voraussetzungen

Für die Durchführung eines Splittingschritts werden die Menge T_i , die fusionierende Abbildung η_i und zu jedem $t \in T_i$ der Stabilisator B_t benötigt. Zudem werden zwei Teilalgorithmen verwendet, die zu einem gegebenen Element $\omega \in \Omega_{i+1}$ und einer gegebenen Gruppe $U \leq B$, deren Bahn ω^U kurz ist, Folgendes berechnen:

Der Algorithmus $FindOrbitRep(\omega, U)$ bestimmt einen kanonischen Repräsentanten t zur Bahn ω^U und gibt ein $u \in U$ zurück, für das $\delta^u = t$ ist. Die Auswahl des Bahnrepräsentanten t muss unabhängig davon sein, welches Bahnelement aus der jeweiligen Menge ω^U an den Algorithmus übergeben wird. Es gilt also:

$$\forall \omega \in \Omega_{i+1} \forall \delta \in \omega^U : \quad \omega^{FindOrbitRep(\omega, U)} = \delta^{FindOrbitRep(\delta, U)}$$

Der Algorithmus $ReduceStab(\omega, U)$ berechnet den Stabilisator von ω in U .

4.3.2 Vorgehen beim Splitting Orbits

1. Setze Σ gleich T_i , der Menge der kanonischen Bahnrepräsentanten in Ω_i , und $T_{i+1} = \emptyset$.
2. Wähle ein beliebiges Element s aus der Menge Σ aus und bestimme die Menge $\Pi = \{\omega \in \Omega_{i+1} \mid \varphi_i(\omega) = s\}$ und den Stabilisator B_s .
3. Wähle ein beliebiges Element ω aus der Menge Π . Berechne das Element $b = FindOrbitRep(\omega, B_s)$ und setze $f_{i+1}(\omega) = b$.
4. Ist $\omega^b = \omega$, so wird ω zum Repräsentantensystem T_{i+1} hinzugefügt und mit Hilfe des Algorithmus $ReduceStab(\omega, B_s)$ der Stabilisator B_ω berechnet.
5. Entferne ω aus der Menge Π . Enthält die Menge Π weitere Elemente, fahre fort mit Schritt 4.
6. Entferne s aus der Menge Σ . Enthält die Menge Σ weitere Elemente, fahre fort mit Schritt 2.

4.4 Fusing Orbits

Existiert für ein $i < n$ ein B -Homomorphismus $\varphi_i : \Omega_i \rightarrow \Omega_{i+1}$, so kann die Menge T_{i+1} in einem Fusingschritt aus der Menge T_i berechnet werden. Weiterhin kann eine fusionierende Abbildung $\eta_{i+1} : \Omega \rightarrow B$ bestimmt und zu jedem Element $t \in T_{i+1}$ der Stabilisator B_t berechnet werden.

4.4.1 Voraussetzungen

Für die Durchführung eines Fusingschritts werden die Menge T_i , eine fusionierende Abbildung η_i und zu jedem $t \in T_i$ der Stabilisator B_t benötigt. Zudem wird ein Teilalgorithmus $ExtendGroup(b, U)$ verwendet, der zu einem gegebenen Element $b \in B$ und einer gegebenen Gruppe $U \leq B$ die kleinste Untergruppe von B berechnet, die sowohl die Untergruppe U als auch das Element b enthält.

4.4.2 Bestimmung der Bahnrepräsentanten

1. Setze Π gleich der Menge T_i und $T_{i+1} = \emptyset$.
2. Wähle ein beliebiges Element t aus der Menge Π aus und bestimme die Menge $\Sigma = \{\omega^b \in \Omega_i \mid \varphi_i(\omega) = \varphi_i(t), b = \eta_i(\omega)\}$. Damit entspricht die Menge Σ der Menge aller Bahnrepräsentanten $s \in T_i$ mit $\varphi_i(s) \in \varphi_i(t)^{B_t}$.
3. Füge das Element $\varphi_i(t)$ zur Menge T_{i+1} hinzu.
4. Entferne alle Elemente, die in der Menge Σ enthalten sind, aus der Menge Π . Enthält die Menge Π weitere Elemente, fahre fort mit Schritt 2.

4.4.3 Bestimmung der fusionierenden Abbildung und der Stabilisatoren

Die fusionierende Abbildung η_{i+1} kann, wie in Kapitel 4.2 beschrieben, aus der in Kapitel 4.4.1 als bekannt vorausgesetzten fusionierenden Abbildung η_i und einer Funktion f_{i+1} berechnet werden. Die Funktion f_{i+1} wird wie folgt bestimmt:

1. Setze Π gleich der Menge aller Bahnrepräsentanten T_{i+1} .
2. Wähle ein Element t aus der Menge Π aus und setze Σ gleich der Menge $\{s \in \Omega_i \mid \varphi_i(s) = t\}$. Damit entspricht die Menge Σ der Menge aller Urbilder von t . Setze $U = \langle 1_B \rangle$.
3. Wähle ein Element s aus der Menge Σ aus und berechne das fusionierende Element $\eta_i(s)$, das mit b bezeichnet wird.
4. Das Element s^b ist in der Menge T_i enthalten. Setze $f_{i+1}(\varphi_i(s^b)) = f_{i+1}(t^b) = b^{-1}$.
5. Wenn $\varphi_i(s^b) = t$ ist, so kann die Gruppe U , die eine Untergruppe des Stabilisators B_t ist, erweitert werden: Wenn $|U| < |B_{s^b}|$ ist, setze $U = B_{s^b}$, andernfalls ersetze U durch das Ergebnis des Algorithmus *ExtendGroup*(b, U).
6. Entferne s aus der Menge Σ . Enthält die Menge Σ weitere Elemente, fahre fort mit Schritt 3.
7. Die Gruppe U ist jetzt identisch mit dem gesuchten Stabilisator B_t . Entferne t aus der Menge Π . Enthält die Menge Π weitere Elemente, fahre fort mit Schritt 2.

Kapitel 5

Der Kanonisierungsalgorithmus am Beispiel eines Würfels

In diesem Kapitel sollen der Ablauf und die Ideen hinter den in den folgenden Kapiteln beschriebenen Kanonizitätstests vorgestellt werden. Der Schwerpunkt dieses Kapitels liegt auf einer intuitiven und anschaulichen Beschreibung des Kanonizitätstests. Daher wurde an vielen Stellen auf eine exakte Beschreibung der Vorgehensweise verzichtet. Eine genaue und mathematisch fundierte Darstellung des Algorithmus folgt im Kapitel 7.

Der Zusammenhang zwischen den Würfelfärbungen, die in diesem Kapitel zur Darstellung des Kanonizitätstests verwendet werden und den Nebenklassen, die in Kapitel 6 und 7 zur Beschreibung der Algorithmen verwendet werden, ist in Kapitel 7.2 erläutert.

5.1 Drehungen eines Würfels

Es gibt genau 24 Möglichkeiten, einen Würfel im Raum zu drehen, so dass jede Ecke nach der Drehung wieder an einer Position zu Liegen kommt, an der sich auch vor der Drehung eine Ecke befunden hat. Die 24 Positionen, in denen sich ein Würfel nach einer Drehung befinden kann, werden in Abbildung 5.1 veranschaulicht. Die „identische Drehung“, bei der der Würfel unverändert bleibt, wurde in die Menge der Drehungen mit aufgenommen. Denn das vorliegende Beispiel soll die mathematische Beschreibung des Algorithmus vorbereiten, in der die Menge der Drehungen als Gruppe betrachtet wird. Dabei nimmt die

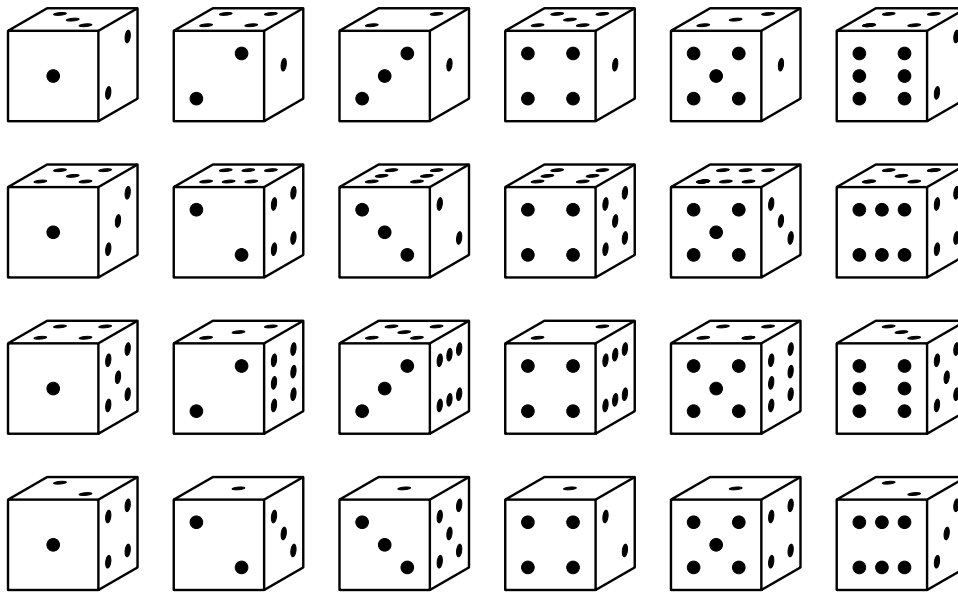


Abbildung 5.1: Die 24 Drehungen eines Würfels

identische Drehung den Platz des neutralen Elementes der Gruppe ein. Die in Abbildung 5.1 dargestellten Würfelaugen dienen nur der Veranschaulichung. Wenn im weiteren Text von Würfeln die Rede ist, so sind immer Würfel ohne Augen gemeint. Mit Drehungen sind im Folgenden immer Drehungen des Würfels im dreidimensionalen Raum gemeint, bei denen Ecken wieder auf Ecken abgebildet werden.

5.2 Fragestellung

In Abbildung 5.2 ist ein Würfel dargestellt, bei dem vier der acht Ecken mit weißen Kreisen markiert wurden. Um die Positionen der markierten Ecken angeben zu können, wurde jede Eckposition mit einer Zahl markiert. Diese Positionsangaben bleiben bei den Drehungen unverändert, das heißt, vor wie nach jeder Drehung wird die vordere, obere, linke Ecke immer als Ecke Nummer eins bezeichnet.

Werden verschiedene Drehungen auf den Würfel in Abbildung 5.2 angewendet, so stellt sich die Frage, bei welchen Drehungen die weißen Ecken wieder

an einer Position landen, an der sich bereits vor der Drehung eine weiße Ecke befunden hat.

Weiterhin stellt sich die Frage, wie viele Möglichkeiten es gibt, vier Ecken eines Würfels weiß zu markieren, wenn nur diejenigen Würfelfärbungen als verschieden betrachtet werden, die durch keine Drehung aufeinander abgebildet werden können.

Zu einer gegebenen Würfelfärbung kann die Menge aller Würfelfärbungen berechnet werden, die durch Drehungen aufeinander abgebildet werden. Diese Menge wird auch die Bahn dieser Würfelfärbung unter der Operation der Gruppe aller Würfeldrehungen genannt.

Wenn aus einer beliebigen Bahn eine Würfelfärbung ausgewählt wurde, die repräsentativ für die ganze Bahn stehen soll, so wird diese ausgewählte Färbung auch kanonischer Repräsentant genannt. In Kapitel 5.3 wird eine Methode vorgestellt, mit der festgelegt werden kann, welche Färbung aus jeder Bahn als kanonisch bezeichnet werden soll.

Wurden bei der Auswahl der kanonischen Repräsentanten gewisse Voraussetzungen erfüllt, so ist der beschriebene Algorithmus dazu geeignet, folgende Fragestellungen zu beantworten:

1. Bei welchen Drehungen landet jede der weiß gefärbten Ecken des Würfels in Abbildung 5.2 wieder auf einer Ecke, die bereits vor der Drehung weiß gefärbt war? (Berechnung des Stabilisators)

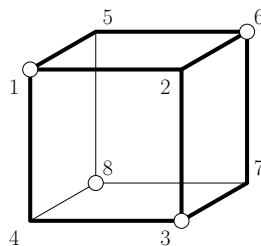


Abbildung 5.2: Würfel mit weiß markierten Ecken

2. Ist der Würfel in Abbildung 5.2 der kanonische Repräsentant seiner Bahn? (Kanonizitätstest)

3. Welche Bahnen von Würfelfärbungen gibt es, bei denen vier Ecken des Würfels weiß gefärbt wurden? (Vollständige Konstruktion bis auf Isomorphie)

Im folgenden Beispiel soll vorgeführt werden, wie sich die Fragen 1 und 2 mit Hilfe des beschriebenen Algorithmus beantworten lassen.

5.3 Kanonische Färbung und Totalordnung

Im Folgenden wird eine Totalordnung auf der Menge der Würfelfärbungen festgelegt. Dies ermöglicht es, Würfelfärbungen anhand dieser Ordnungsrelation miteinander zu vergleichen. Um die Beschreibungen nicht unnötig kompliziert zu machen, wird die Sprechweise „Färbung \mathcal{A} ist größer/kleiner/gleich Färbung \mathcal{B} “ verwendet, wenn zwei Färbungen anhand dieser Ordnungsrelation miteinander verglichen werden.

Die in Abbildungen 5.2 und 5.3 angegebenen Zahlen weisen jeder Position, an der sich eine Ecke des Würfels befindet, eine Zahl zu. Diese Positionsangaben ermöglichen es, eine Würfelfärbung durch Angabe der markierten Würfecken zu charakterisieren. Um eine eindeutige Beschreibung jeder Würfelfärbung zu ermöglichen, wird eine einheitliche Beschreibung durch sogenannte Färbungstupel eingeführt.

Im vorliegenden Beispiel werden nur schwarze, weiße und ungefärbte Ecken verwendet. Die Farbe Schwarz wird mit dem Buchstaben s und die Farbe Weiß mit dem Buchstaben w abgekürzt. Im Färbungstupel sollen zuerst alle Ecken, die mit weißer Farbe markiert wurden und erst anschließend die Ecken, die mit schwarzer Farbe markiert wurden, aufgezählt werden. Um die Farbe dieser Ecke zu verdeutlichen, soll jeweils die Position der Ecke, gefolgt vom Buchstaben w , beziehungsweise s , geschrieben werden. Dabei sollen die Positionen aller gleichfarbig markierten Ecken immer der Reihenfolge nach aufgezählt werden.

In Abbildung 5.2 sind die Ecken an den Positionen 1, 3, 6 und 8 weiß gefärbt, daher entspricht die Färbung dem Tupel $(1, w, 3, w, 6, w, 8, w)$.

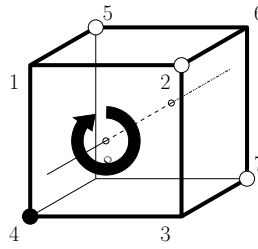


Abbildung 5.3: Würfel mit dem Färbungstupel $(2, w, 5, w, 7, w, 4, s)$

Bei dem Würfel in Abbildung 5.3 hingegen sind die Ecken an den Positionen 2, 5 und 7 weiß und die Ecke an Position 4 schwarz gefärbt, daher entspricht die Färbung dem Tupel $(2, w, 5, w, 7, w, 4, s)$.

Anhand dieses Färbungstupels kann nun eine Totalordnung auf der Menge der Würfelfärbungen angegeben werden. Für alle untersuchten Würfelfärbungen wird die Ordnung so festgelegt, dass sie der lexikographischen Ordnung der jeweiligen Färbungstupel entspricht.

Aus jeder Bahn soll genau diejenige Färbung als kanonisch gelten, deren Färbungstupel kleiner ist als das aller anderen Färbungen derselben Bahn.

5.4 Stabilisatoren

Zu jedem beliebig markierten Würfel bildet die Menge aller Drehungen, bei denen das Färbungstupel vor der Drehung dasselbe ist wie nach der Drehung, eine Gruppe. Diese Gruppe wird der Stabilisator dieses markierten Würfels genannt.

In diesem Beispiel wird unter anderem gezeigt, wie der Stabilisator der Würfelfärbung aus Abbildung 5.2 berechnet werden kann.

5.5 Homomorphismen

In diesem Beispiel werden zwei verschiedene Abbildungen verwendet. Die erste Abbildung hat eine Definitionsmenge, die aus allen Würfeln besteht, bei denen genau eine Ecke schwarz und zwischen null und sieben Ecken weiß markiert

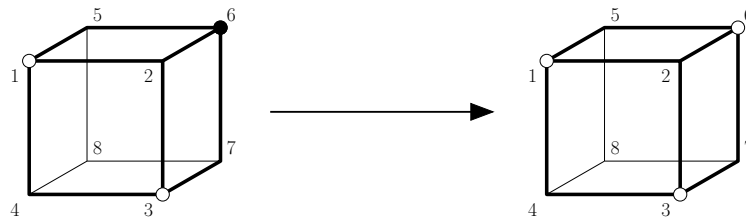


Abbildung 5.4: Beispiel einer Abbildung unter dem Fusing-Homomorphismus

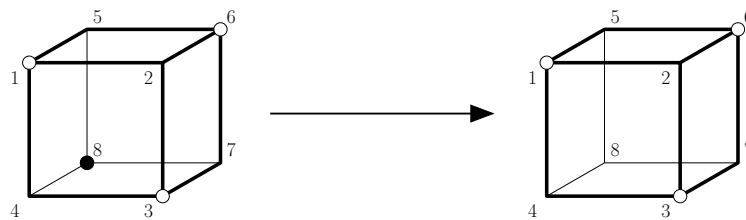


Abbildung 5.5: Beispiel einer Abbildung unter dem Splitting-Homomorphismus

sind. Jede dieser Würfelfärbungen wird auf einen Würfel abgebildet, der an denselben Ecken markiert wurde wie der Würfel im Urbild, dessen Markierungen aber alle weiß sind. Diese Abbildung wird im Folgenden auch Fusing-Homomorphismus genannt und ist in Abbildung 5.4 dargestellt.

Die zweite Abbildung hat dieselbe Definitionsmenge wie die erste. Diese Definitionsmenge besteht aus allen Würfeln, bei denen genau eine Ecke schwarz und zwischen null und sieben Ecken weiß markiert sind. Jeder Würfel aus dieser Menge wird durch die zweite Abbildung auf einen Würfel abgebildet, bei dem genau diejenigen Ecken, die im Urbild weiß gefärbt waren, auch wieder weiß sind, dessen übrige Ecken aber alle unmarkiert bleiben. Diese Abbildung wird im Folgenden auch Splitting-Homomorphismus genannt und ist in Abbildung 5.5 dargestellt.

5.6 Konstruktionspfade

Ein Konstruktionspfad ist eine Folge von Würfelfärbungen, die beim völlig unmarkierten Würfel beginnt und die folgenden Bedingungen erfüllt: Für je

zwei aufeinander folgende Würfelfärbungen des Tupels, von denen die erste eine schwarz markierte Ecke enthält, muss gelten, dass die zweite gleich dem Bild der ersten unter dem Fusing-Homomorphismus ist. Für je zwei aufeinander folgende Würfelfärbungen des Tupels, von denen die erste keine schwarz markierte Ecke enthält, muss gelten, dass die erste gleich dem Bild der zweiten unter dem Splitting-Homomorphismus ist. In Abbildung 5.6 ist ein Beispiel für einen Konstruktionspfad zu sehen.

Unter Verwendung der Totalordnung auf der Menge der Würfelfärbungen kann jetzt eine Totalordnung auf der Menge der Konstruktionspfade definiert werden. Im Folgenden soll genau derjenige von zwei gegebenen Konstruktionspfaden als kleiner gelten, dessen Tupel lexikographisch kleiner ist als das des anderen Pfades.

Konstruktionspfade werden im folgenden Text auch kurz als Pfade bezeichnet.

5.7 Vorbereitungen

Als Vorbereitung für den Algorithmus ist es erforderlich, den kleinsten Pfad zu dem in Abbildung 5.2 gegebenen Würfel zu finden. Dieser kleinste Pfad ist in Abbildung 5.6 dargestellt und wird im Folgenden mit ρ bezeichnet. Dieser kleinste Pfad kann unter Verwendung des in Kapitel 7.6.6 beschriebenen Algorithmus leicht gefunden werden, daher soll an dieser Stelle nicht weiter darauf eingegangen werden.

Als weiterer Vorbereitungsschritt muss für den Pfad ρ geprüft werden, ob jede Komponente, außer der letzten, eine kanonische Würfelfärbung ist. Ist einer der Würfel des Pfades nicht kanonisch, so sind auch die darauf folgenden Würfel des Pfades ρ nicht kanonisch. Dies kann in ähnlicher Weise wie die Beweisführung von Lemma 6 auf Seite 68 nachgewiesen werden, auf den Nachweis soll jedoch an dieser Stelle verzichtet werden.

Wenn sich in diesem Vorbereitungsschritt herausstellt, dass eine der Komponenten des Pfades ρ nicht kanonisch ist, so muss der in diesem Beispiel

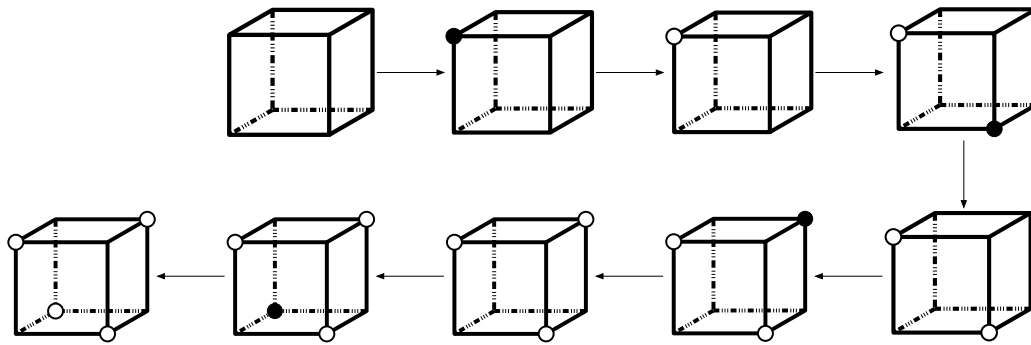


Abbildung 5.6: Der kleinste Pfad zum Würfel in Abbildung 5.2

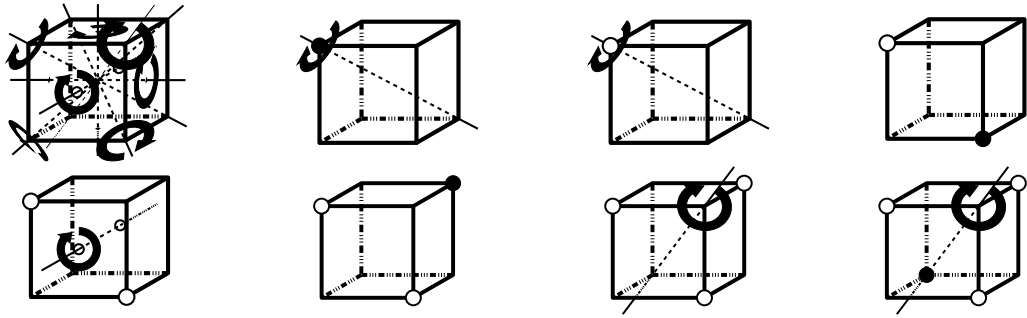
betrachtete Algorithmus nicht mehr durchgeführt werden, da das Ergebnis bereits fest steht. Die Überprüfung, ob jede, außer der letzten, Komponente des Pfades ρ kanonisch ist, kann durch einen rekursiven Aufruf des hier beschriebenen Algorithmus erfolgen. Daher soll auch auf diesen Schritt nicht weiter eingegangen werden. Die Überprüfung der letzten Komponente des Pfades ρ wird im Folgenden ausführlich durchgeführt.

In Zusammenhang mit der Überprüfung, ob die Komponenten des Pfades ρ kanonisch sind, können die zugehörigen Stabilisatoren aller dieser Komponenten berechnet werden. Die Stabilisatoren aller, außer der letzten, Würfelfärbung des Pfades ρ werden im weiteren Verlauf des Algorithmus noch benötigt und sind in Abbildung 5.7 dargestellt. Die Berechnung des Stabilisators der letzten Komponente des Pfades ρ wird im Folgenden ausführlich durchgeführt.

5.8 Erweiterung des Stabilisators

Der Splitting-Homomorphismus besitzt eine sehr interessante Eigenschaft, wie sich anhand der Stabilisatoren zweier Würfel im Bild und im Urbild des Homomorphismus zeigen lässt. Der Würfel im Urbild und der Würfel im Bild des Splitting-Homomorphismus unterscheiden sich nur in der Färbung einer einzigen Ecke. Diese Ecke ist im Urbild schwarz und im Bild ungefärbt.

Jede Drehung, die im Stabilisator des Urbildes liegt, muss auch im Stabilisator des Bildes liegen. Denn jede Drehung, die das Urbild auf sich selbst abbildet,

Abbildung 5.7: Würfel des Pfades ρ und deren Stabilisatoren

muss immer auch alle weißen Ecken aufeinander abbilden.

Der Fusing-Homomorphismus verhält sich genauso in Bezug auf die Stabilisatoren in Bild und Urbild. Eine Würfelfärbung im Definitionsbereich und die entsprechende Würfelfärbung im Bild des Fusing-Homomorphismus unterscheiden sich auch wieder nur in einer einzigen Ecke. Die Ecke, die beim Würfel im Urbild schwarz gefärbt ist, ist bei der Würfelfärbung im Bild weiß. Jede Drehung, die die schwarz und die weiß gefärbten Ecken des Würfels im Urbild auf sich selbst abbildet, muss auch die weiß gefärbten Ecken des Würfels im Bild aufeinander abbilden. Daher muss jede Drehung aus dem Stabilisator einer gegebenen Würfelfärbung auch im Stabilisator der entsprechenden Würfelfärbung im Bild enthalten sein.

Diese Erkenntnis kann dazu benutzt werden, eine Untergruppe des Stabilisators der Würfelfärbung aus Abbildung 5.2 zu bestimmen. Die Abbildung zwischen dem vorletzten Würfel des Pfades ρ und dem letzten Würfel des Pfades ist ein Fusing-Homomorphismus. Daher muss jede Drehung aus dem Stabilisator der vorletzten Würfelfärbung des Pfades ρ auch im gesuchten Stabilisator der letzten Würfelfärbung enthalten sein.

Die drei Drehungen um die, in Abbildung 5.8 skizzierte Achse, werden aus dem Stabilisator der vorletzten Würfelfärbung in den Stabilisator der letzten Würfelfärbung des Pfades ρ übernommen.

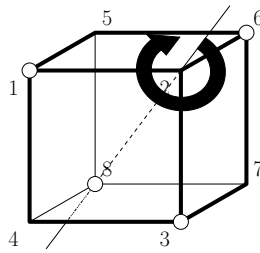


Abbildung 5.8: Eine Untergruppe des Stabilisators

5.9 Menge der Pfade

In Abbildung 5.9 sind alle Pfade dargestellt, die vom Würfel ohne gefärbte Ecken zu dem zu untersuchenden Würfel aus Abbildung 5.2 führen. Diese Pfade sollen im weiteren Verlauf des Algorithmus in Tiefensuche durchlaufen werden.

Nur ein Bruchteil dieser Pfade wird jedoch dazu beitragen, neue Stabilisatoren des Würfels in Abbildung 5.2 zu finden. Um zu überprüfen, ob der Würfel in Abbildung 5.2 kanonisch ist, wird auch nur eine kleine Teilmenge der dargestellten Pfade benötigt. Um zu vermeiden, dass alle diese Pfade trotzdem betrachtet werden müssen, werden Kriterien benötigt, die es ermöglichen, Pfade von der Untersuchung auszuschließen, die das Ergebnis nicht beeinflussen.

5.10 Schnittmengen von Stabilisatoren

Beginnend mit dem Pfad, der nur aus einem ungefärbten Würfel besteht, sollen die Pfade in Tiefensuche durchlaufen werden, indem der bestehende Pfad schrittweise um weitere Würfelfärbungen verlängert wird. Zu jeder Würfelfärbung, die an einen bestehenden Pfad angehängt werden soll, wird weiterhin eine besondere Untergruppe des Stabilisators dieses Würfels berechnet. Diese Untergruppe ermöglicht es, Pfadmengen von der Untersuchung auszuschließen und so die zu untersuchende Menge von Pfaden zu verkleinern.

Die gesuchte Gruppe ist eine Untergruppe des Stabilisators der letzten Würfelfärbung des betrachteten Pfades. Diese besteht genau aus der Menge aller Drehungen, die zusätzlich auch im aktuell bekannten Stabilisator des Würfels

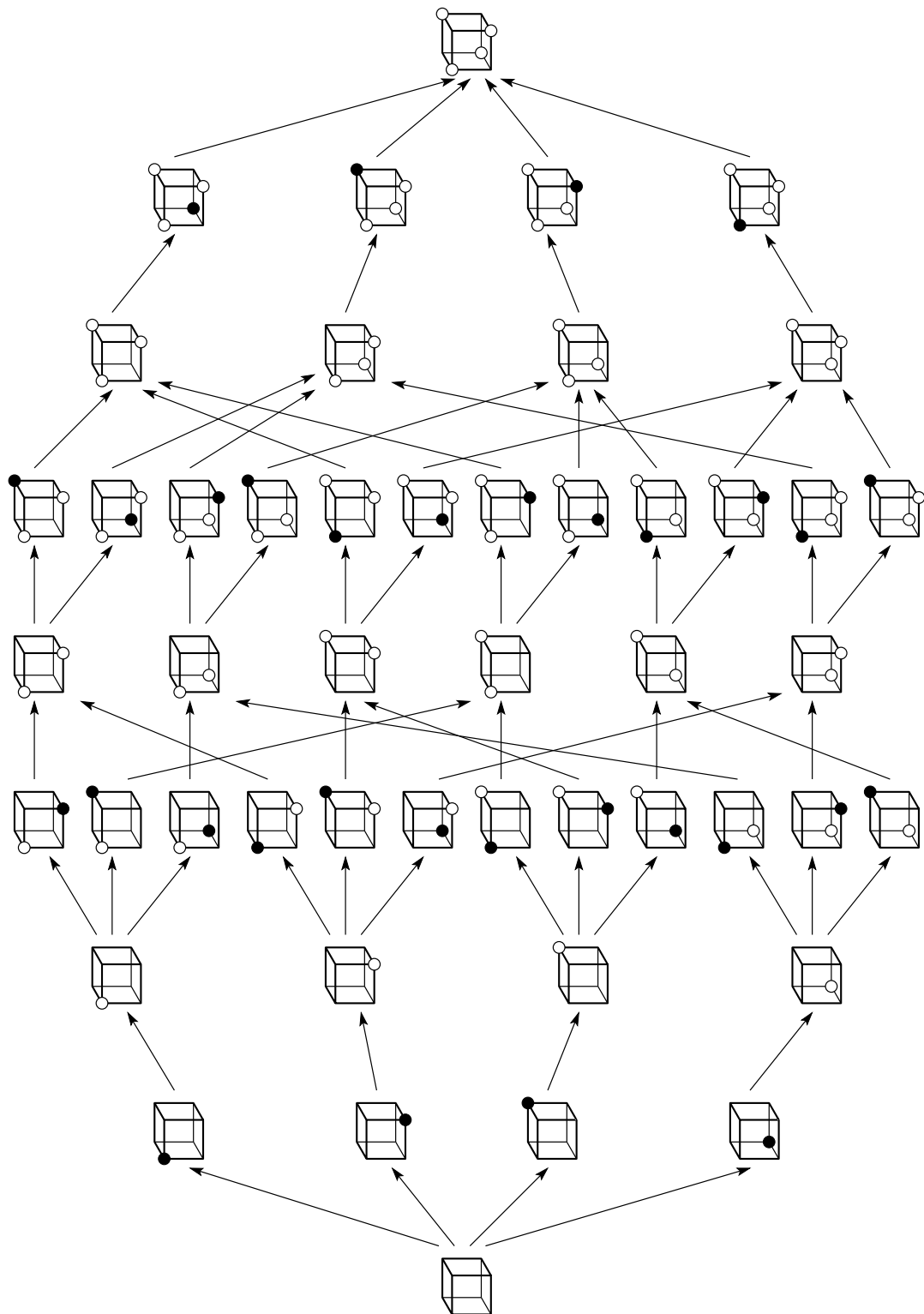


Abbildung 5.9: Alle Pfade zum Würfel in Abbildung 5.2

in Abbildung 5.2 enthalten sind.

Diese Menge ist identisch mit dem Stabilisator des betrachteten Würfels in der Gruppe aller Drehungen des bisher bekannten Stabilisators des Würfels in Abbildung 5.2. Daher kann zur Berechnung dieser Untergruppe ein rekursiver Aufruf des hier in diesem Beispiel vorgestellten Algorithmus verwendet werden. Im Folgenden soll zu Gunsten der Übersicht nicht mehr weiter auf die Berechnung dieser Untergruppen eingegangen werden.

5.11 Fusionierende Drehung

Um das Abschneiden von Pfaden zu ermöglichen und den gesuchten Stabilisator zu berechnen, muss zu jeder Würfelfärbung, die an einen bestehenden Pfad angehängt werden soll, eine sogenannte fusionierende Drehung bestimmt werden. Als fusionierende Drehungen werden diejenigen Drehungen bezeichnet, die eine gegebene Würfelfärbung auf den kanonischen Repräsentanten dieser Bahn abbilden.

Bei der Bestimmung einer fusionierenden Drehung können zwei Fälle unterschieden werden. Enthält die Würfelfärbung, um die ein gegebener Pfad verlängert werden soll, keine schwarze Ecke, so braucht die fusionierende Drehung nicht explizit berechnet werden. Denn die fusionierende Drehung der letzten Würfelfärbung des noch nicht verlängerten Pfades ist bereits eine der gesuchten fusionierenden Drehungen zu der Würfelfärbung, die an den Pfad angehängt werden soll. Dies ergibt sich als direkte Folge aus den Eigenschaften des Fusing-Homomorphismus.

Andernfalls soll anhand eines Beispiels erläutert werden, wie eine fusionierende Drehung gefunden werden kann:

Der in Abbildung 5.10 gegebene Pfad soll um den in Abbildung 5.11 dargestellten Würfel verlängert werden. Die fusionierende Abbildung bildet die untersuchte Würfelfärbung immer auf eine Würfelfärbung ab, die dieselbe Anzahl von schwarz und weiß gefärbten Ecken besitzt. Denn jede Drehung eines

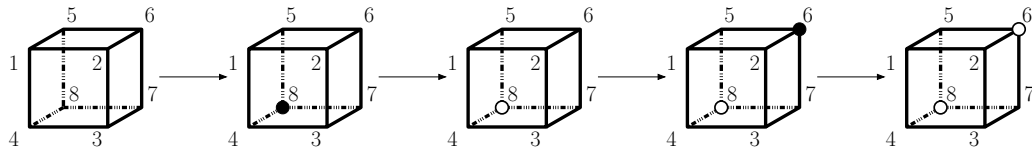


Abbildung 5.10: Beispiel eines Pfades

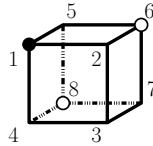


Abbildung 5.11: Würfel zur Verlängerung des Pfades aus Abbildung 5.10

Würfel, dessen Ecken farblich markiert wurden, lässt die Anzahl der schwarz und weiß markierten Ecken unverändert.

Zu jeder Würfel­färbung des untersuchten Pfades existiert genau eine Würfel­färbung des Pfades ρ , die dieselbe Anzahl von schwarz und weiß gefärbten Ecken besitzt. Es kann davon ausgegangen werden, dass jede Würfel­färbung des noch nicht verlängerten Pfades in der Bahn einer Würfel­färbung des Pfades ρ liegt. Die Pfade, bei denen dies nicht der Fall ist, werden gesondert betrachtet und die Bestimmung der fusionierenden Abbildung ist in diesen Fällen nicht notwendig.

Da in Kapitel 5.7 vorausgesetzt wurde, dass jede, außer der letzten, Würfel­färbung des Pfades ρ kanonisch sein muss, kann davon ausgegangen werden, dass eine Drehung, die die untersuchte Würfel­färbung auf eine Würfel­färbung des Pfades ρ abbildet, eine der gesuchten fusionierenden Abbildungen ist.

Unter Anwendung der fusionierenden Drehung der letzten Würfel­färbung des noch nicht verlängerten Pfades kann diese letzte Würfel­färbung auf den entsprechend markierten Würfel des Pfades ρ abgebildet werden. Diese fusionierende Drehung kann als bekannt vorausgesetzt werden und ist im Falle des Pfades aus Abbildung 5.10 eine Drehung um 180 Grad um die in Abbildung 5.12 dargestellte Achse. In Kapitel 5.7 wurde der Stabilisator dieser Würfel­färbung des Pfades ρ bereits bestimmt. Der Stabilisator dieses Würfels aus dem Pfad ρ ist in Abbildung 5.14 skizziert.

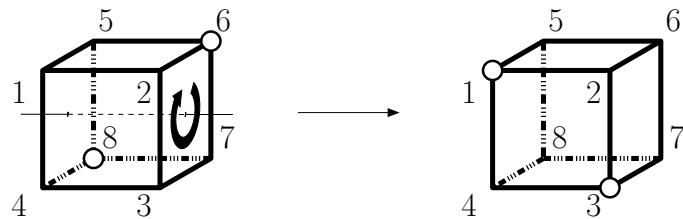


Abbildung 5.12: Würfel mit fusionierender Drehung

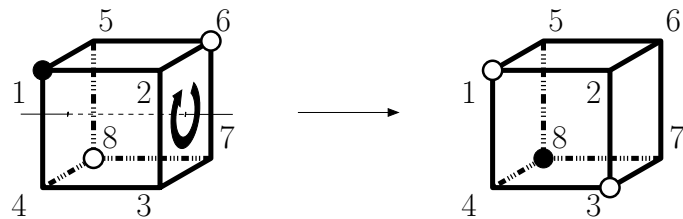


Abbildung 5.13: Wende fusionierende Drehung aus Abbildung 5.12 an

Anschließend soll die fusionierende Drehung aus Abbildung 5.12 auf den Würfel in Abbildung 5.11 angewendet werden, wie dies in Abbildung 5.13 dargestellt ist.

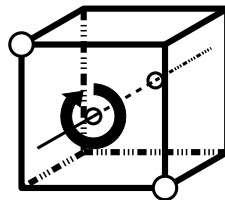


Abbildung 5.14: Stabilisator des Bahnrepräsentanten

Jede der Drehung des Stabilisators aus Abbildung 5.14 soll anschließend auf den in Abbildung 5.13 auf der rechten Seite dargestellten Würfel angewendet werden. Dies führt zu der Menge von Würfelfärbungen, die Bahn genannt wird. Die Bahn unter der Operation des Stabilisators aus Abbildung 5.14 ist in Abbildung 5.15 dargestellt.

Aus dieser Menge von Würfelfärbungen wird diejenige, deren Färbungstupel am kleinsten ist, ausgewählt. Diese Würfelfärbung ist bereits die gesuchte kanonische Würfelfärbung zur Würfelfärbung aus Abbildung 5.11. Jede Drehung,

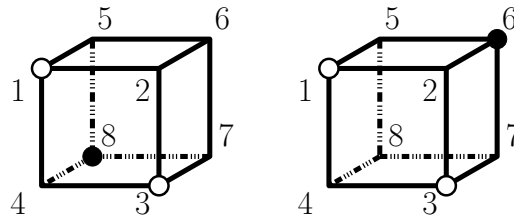


Abbildung 5.15: Bahn unter der Operation des Stabilisators aus Abbildung 5.14

die den ursprünglichen Würfel aus Abbildung 5.11 auf diesen ausgewählten Würfel abbildet, ist eine der gesuchten fusionierenden Drehungen. Eine dieser fusionierenden Drehungen ist in Abbildung 5.16 skizziert.

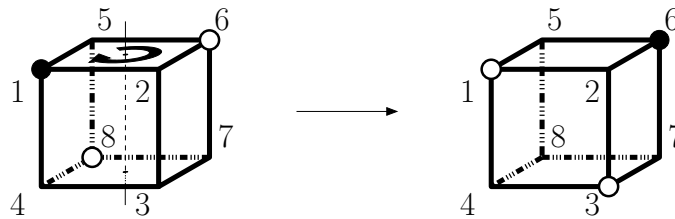


Abbildung 5.16: Würfel mit fusionierender Drehung

Wird, wie in Abbildung 5.16, die fusionierende Drehung eines Würfels auf diesen Würfel selbst angewandt und ist die Würfelfärbung, die sich daraus ergibt, kleiner als die Würfelfärbung des entsprechenden Würfels des Pfades ρ , so ist der Würfel am Ende des Pfades ρ nicht kanonisch.

Dies kann auf ähnliche Weise wie in Lemma 6 bewiesen werden. An dieser Stelle soll jedoch darauf verzichtet werden. Ist das Färbungstupel des Würfels, der sich aus der Anwendung der fusionierenden Drehung eines Würfels auf diesen Würfel selbst ergibt, größer als das Färbungstupel des entsprechenden Würfels des Pfades ρ , so braucht keiner der Pfade, die diesen Würfel enthalten, untersucht werden. Denn diese Pfade sind weder für die Überprüfung der Kanonizität noch für die Berechnung des Stabilisators des Würfels in Abbildung 5.2 von Bedeutung.

5.12 Tiefensuche

Die in Abbildung 5.9 dargestellten Pfade werden jetzt in Tiefensuche durchlaufen. Beginnend mit dem Pfad ρ , der nicht weiter zur Berechnung des Stabilisators und der Überprüfung der Kanonizität beiträgt, wird der nächstmögliche Pfad untersucht. Dies ist der Pfad, der als rechte Würfelreihe in Abbildung 5.17 dargestellt ist. Die linke Würfelreihe in dieser Abbildung stellt den Pfad ρ dar.

Bei dem in Abbildung 5.17 dargestellten Pfad bildet die fusionierende Drehung des vorletzten Würfels des rechten Pfades diesen auf den vorletzten Würfel des linken Pfades ab. Da der Fusing-Homomorphismus den jeweils vorletzten Würfel beider Pfade auf den gemeinsamen letzten Würfel abbildet, liegt die fusionierende Drehung im gesuchten Stabilisator des letzten Würfels des Pfades ρ .

Der bisher bekannte Stabilisator des letzten Würfels des Pfades kann daher so erweitert werden, dass dieser zusätzlich alle Drehungen enthält, die sich als Kombinationen aus Drehungen des bisherigen Stabilisators und der fusionierenden Drehung des vorletzten Würfels des Pfades ergeben. Wie diese Gruppe am geschicktesten berechnet werden kann, soll an dieser Stelle jedoch nicht erklärt werden. Die Beschreibung einer geeigneten Methode ist in Kapitel 7.6.4 zu finden.

Der neue, um die fusionierende Drehung erweiterte Stabilisator des untersuchten Würfels, ist anhand der beiden Würfel in 5.18 dargestellt. Die Drehungen um die vier in 5.18 auf der linken Seite dargestellten Achsen sind Drehungen um 120 Grad, während die Drehungen um die auf der rechten Seite dargestellten Achsen Drehungen um 180 Grad sind. Der neue Stabilisator besteht aus insgesamt 12 Drehungen und ist damit nach wie vor kleiner als der Stabilisator des unmarkierten Würfels, der aus 24 Drehungen besteht.

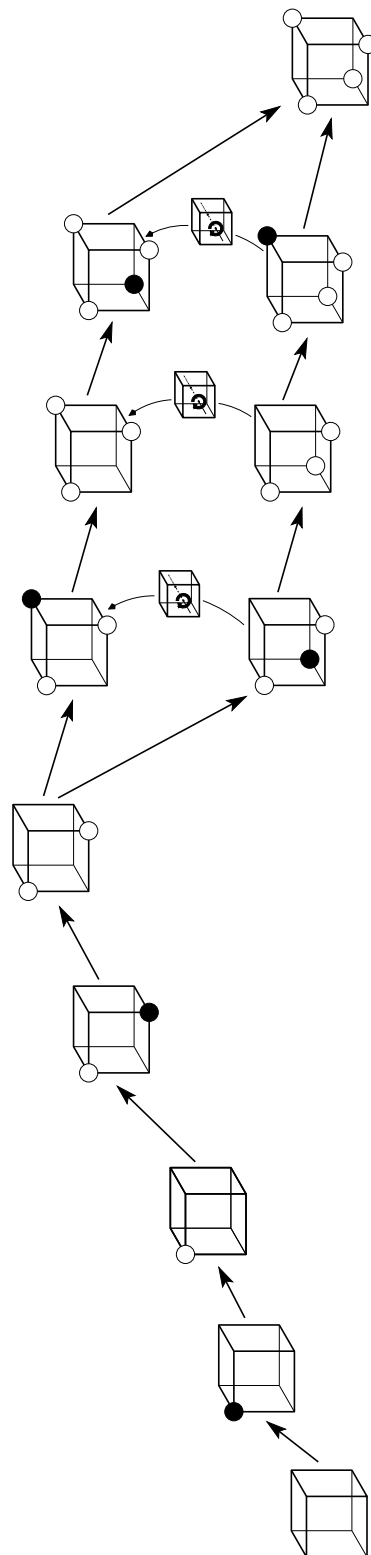


Abbildung 5.17: Pfad mit Darstellung der fusionierenden Drehungen

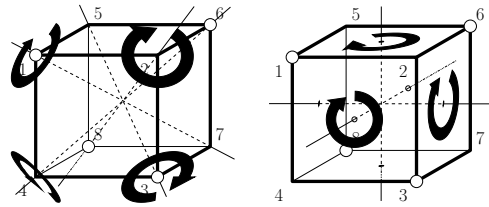


Abbildung 5.18: Neuer erweiterter Stabilisator

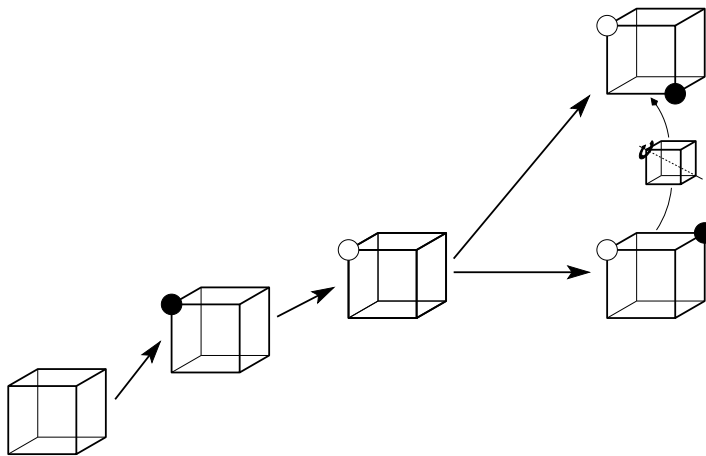


Abbildung 5.19: Abschneiden von Pfaden

5.13 Abschneiden von Ästen

Der nächste Pfad, der bei der Tiefensuche betrachtet wird, enthält den in Abbildung 5.19 auf der rechten Seite unten dargestellten Würfel. Die fusionierende Drehung ist in Form eines kleinen Würfels angedeutet. Bei dieser fusionierenden Drehung ist bemerkenswert, dass diese sowohl im neuen, erweiterten Stabilisator des letzten Würfels des Pfades ρ enthalten ist als auch in dem in Abbildung 5.20 dargestellten Stabilisator des vorangehenden Würfels des Pfades. Das heißt, die zum vorangehenden Würfel berechnete Schnittmenge von Stabilisatoren, die in Kapitel 5.10 beschrieben wurde, enthält die fusionierende Drehung des darauf folgenden Würfels des Pfades.

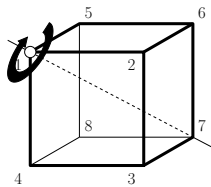


Abbildung 5.20: Würfel mit Stabilisator

In solchen Fällen kann die weitere Untersuchung aller Pfade, die den in Abbildung 5.19 auf der rechten Seite unten dargestellten Würfel enthalten, abgebrochen werden. Keiner dieser Pfade kann zur Erweiterung des Stabilisators oder zur Überprüfung der Kanonizität beitragen. Denn jeder Pfad, der durch diesen Würfel verläuft, besteht ausschließlich aus Würfelfärbungen, deren kanonische Bahnrepräsentanten bereits untersucht wurden.

Dies gilt ganz ähnlich für den nächstmöglichen Pfad, der nicht durch diesen Würfel verläuft. Denn dieser Pfad enthält den in 5.21 unten rechts dargestellten Würfel. Für diesen Würfel gilt genauso, dass die fusionierende Drehung in der Schnittmenge des Stabilisators des vorangehenden Würfels und des letzten Würfels im Pfad ρ enthalten ist. Daher wird auch hier die Untersuchung aller Pfade, die durch diesen Würfel verlaufen, abgebrochen.

Auf der rechten Seite in Abbildung 5.22 sind alle möglichen Würfelfärbungen dargestellt, die an der zweiten Position eines Pfades stehen können. Zu

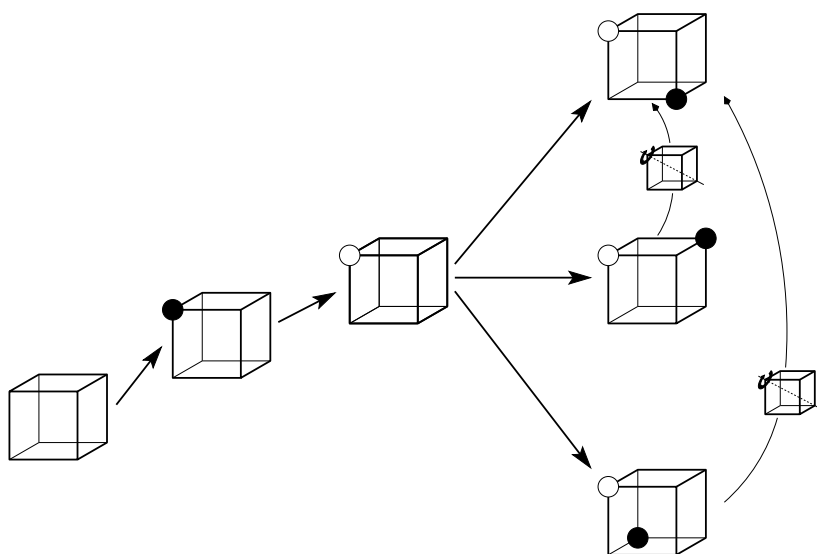


Abbildung 5.21: Abschneiden von zwei Ästen

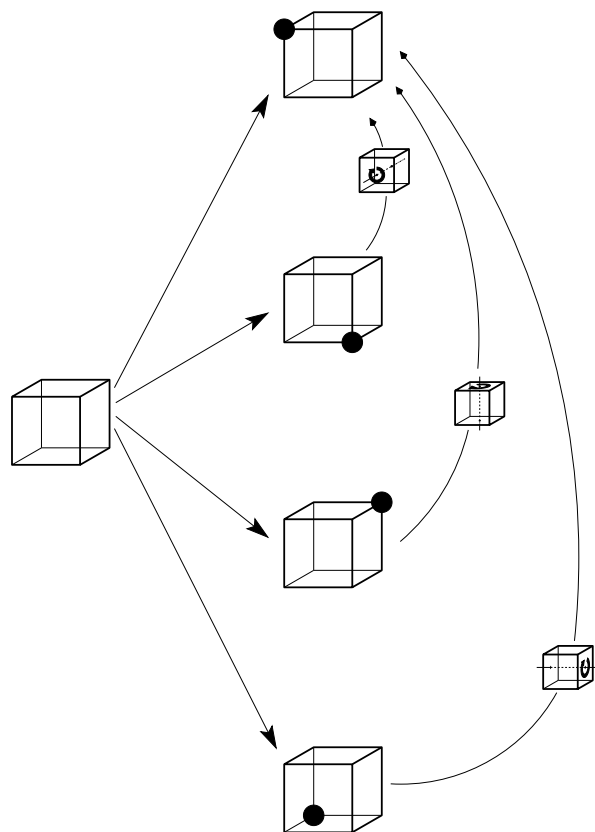


Abbildung 5.22: Abschneiden von drei Ästen

drei der vier Würfel wurde eine fusionierende Drehung skizziert, die sowohl im Stabilisator des vorangehenden unmarkierten Würfels als auch im Stabilisator der letzten Würfelfärbung des Pfades ρ enthalten ist. Daher kann, wie bereits zuvor erläutert wurde, jeder Pfad, der durch einen dieser drei Würfel verläuft, auf einen bereits untersuchten Pfad abgebildet werden.

5.14 Ergebnisse

Durch die Abschneidetechnik des hier vorgestellten Algorithmus konnten viele der in Abbildung 5.9 dargestellten Pfade bereits frühzeitig abgeschnitten werden. Dies ist in Abbildung 5.23 anschaulich dargestellt. Während der Tiefensuche wurde kein Würfel gefunden, dessen Färbungstupel kleiner ist als das des entsprechend gefärbten Würfels im Pfad ρ . Daraus kann gefolgert werden, dass der ursprüngliche in Abbildung 5.2 dargestellte Würfel kanonisch ist. Nachdem der Algorithmus vollständig durchlaufen wurde, ist die zuletzt berechnete Untergruppe des Stabilisators der letzten Würfelfärbung des Pfades ρ maximal. Diese wurde durch sukzessives Erweitern aus dem Stabilisator der vorletzten Würfelfärbung berechnet. Daher ist diese Gruppe identisch mit dem Stabilisator der ursprünglich untersuchten Würfelfärbung, der letzten Komponente des Pfades ρ .

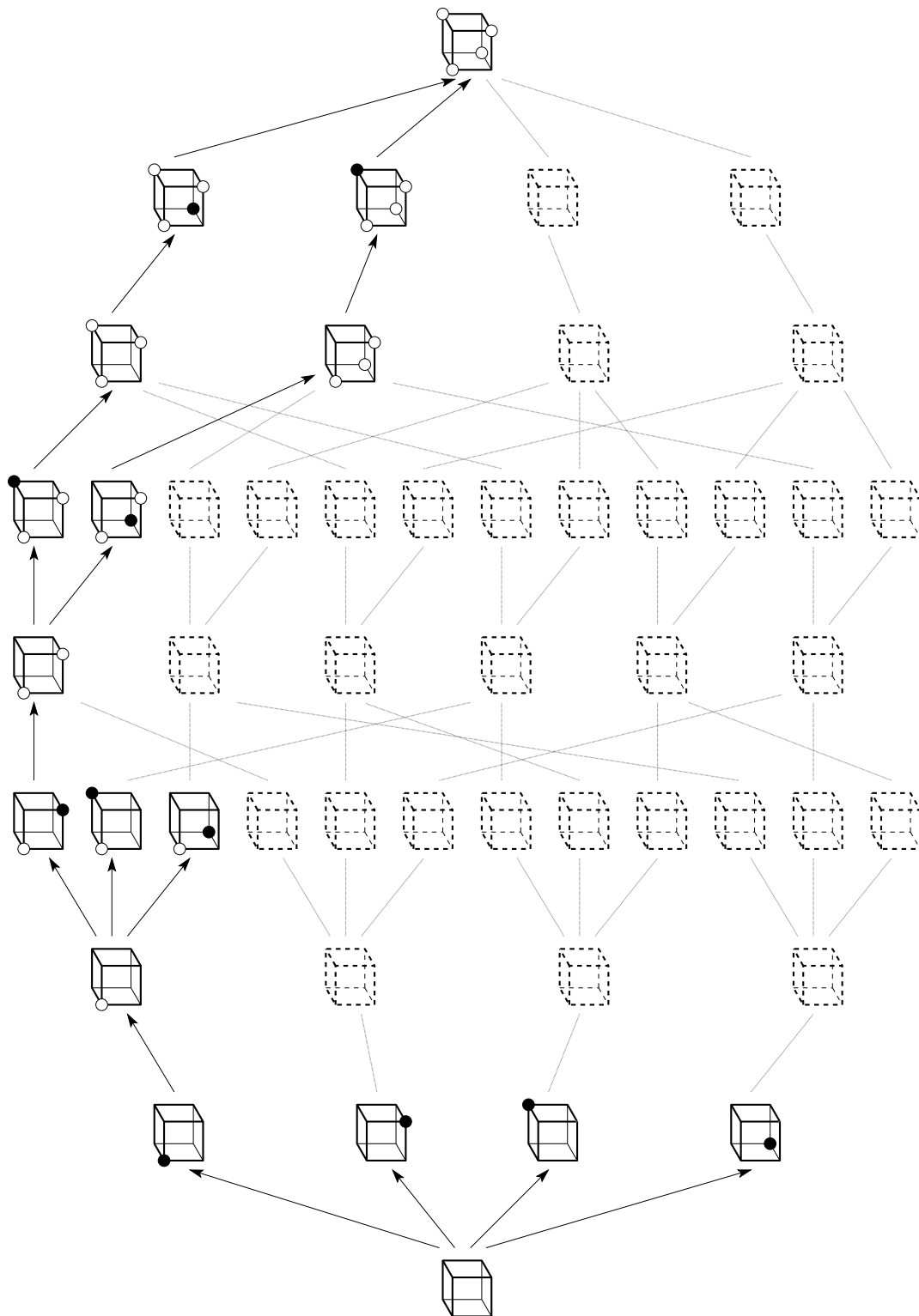


Abbildung 5.23: Anteil der im Algorithmus betrachteten Würfelfärbungen

Kapitel 6

Mathematische Grundlagen der Algorithmen

Beim Leiterspiel-Algorithmus von Schmalz müssen bei jedem Splitting- und Fusingschritt alle Bahnrepräsentanten aus dem vorangehenden Schritt im Speicher gehalten werden. Denn in den Fusingschritten des Algorithmus werden zur Konstruktion eines neuen Bahnrepräsentanten die Bahnrepräsentanten aus dem vorangehenden Schritt verwendet. Um einschränken zu können, welche vorangehenden Bahnrepräsentanten zur Konstruktion eines gesuchten Bahnrepräsentanten benötigt werden, müsste die fusionierende Abbildung berechnet werden. Diese fusionierende Abbildung kann jedoch nur aus der Gesamtmenge aller Bahnrepräsentanten des vorangehenden Schrittes bestimmt werden.

Zwar wäre es prinzipiell auch mit dem ursprünglichen Algorithmus von Schmalz möglich, die fusionierende Abbildung rekursiv zu bestimmen, dies ist aber aus Effizienzgründen nicht sinnvoll. Denn jeder in einem Fusingschritt durchgeführte rekursive Aufruf würde bis zu einer Anzahl von $(A_{j+1} : A_j)$ rekursiven Aufrufen nötig machen, wobei $(A_{j+1} : A_j)$ den Gruppenindex der beiden an diesem Fusingschritt beteiligten Leitergruppen bezeichnet. Aufgrund der Rekursivität führt jeder einzelne dieser Aufrufe im nächsten Fusingschritt wiederum zu einer entsprechenden Anzahl rekursiver Aufrufe. Dies führt schon ab einer geringen Anzahl von Fusingschritten zu einem völlig inakzeptablen Rechenaufwand.

Die wichtigste Erkenntnis der vorliegenden Arbeit ist, dass es mit Hilfe der in Kapitel 6.5 vorgestellten starken Untergruppenleitern möglich ist, zu bestimmen, welche Nebenklassen aus den vorangehenden Schritten zur Berechnung der fusionierenden Abbildung benötigt werden. Dies wird anhand der in Kapitel 6.2 definierten Konstruktionspfade dargestellt und nachgewiesen. In Kapitel 6.4 werden Blöcke bestehend aus Pfaden beschrieben, die für die Beweisführung und die Beschreibungen der Algorithmen unverzichtbar sind. Weiterhin wird in Satz 6.5.2 eine Bedingung gestellt, die hinreichend und notwendig für alle starken Leitern ist und die es ermöglicht, zu vielen Gruppen eine geeignete starke Untergruppenleiter zu erstellen.

Da es unter Anwendung der starken Untergruppenleitern möglich ist, vorherzusagen, welche Nebenklassen zur Berechnung der fusionierenden Abbildung benötigt werden, kann die Ordnungstreue Erzeugung zur Berechnung dieser Nebenklassen eingesetzt werden. Auch dies war beim ursprünglichen Leiterspiel nach Schmalz nicht der Fall. Die Frage, ob und wie sich das Leiterspiel mit einer Konstruktion in Tiefensuche und der Ordnungstreuen Erzeugung verbinden ließe, wurde bereits 1998 von McKay [McK98] aufgeworfen. Mit der vorliegenden Arbeit konnte diese Frage beantwortet und die Ordnungstreue Erzeugung mit dem Leiterspiel-Algorithmus verbunden werden.

6.1 Grundbegriffe

Es sei G eine Gruppe und $\{A_1, \dots, A_n\}$ eine Menge von Untergruppen von G . Die Folge (A_1, \dots, A_n) sei eine Untergruppenleiter von G nach A_n (siehe Definition 11) und B eine weitere Untergruppe von G . Zu jeder Gruppe $A_k \in \{A_1, \dots, A_n\}$ wird die Menge der Rechtsnebenklassen von A_k in der Gruppe G mit Ω_k bezeichnet. Rechtsnebenklassen werden im Folgenden auch kurz als Nebenklassen bezeichnet.

6.1.1 Erweiterungsfunktion

Zu allen $1 \leq k < n$ wird im Folgenden eine Abbildung Υ_k von der Menge Ω_k auf die Potenzmenge $\mathcal{P}(\Omega_{k+1})$ von Ω_{k+1} definiert, die Erweiterungsfunktion

genannt wird. Mit Hilfe dieser Funktion $\Upsilon_k : \Omega_k \rightarrow \mathcal{P}(\Omega_{k+1})$ kann aus der Menge Ω_k die Menge Ω_{k+1} konstruiert werden.

Splittingschritt

Ist für ein $k < n$ die Gruppe A_{k+1} Untergruppe von A_k , so wird der Konstruktionsschritt, der aus den Elementen von Ω_k die Elemente von Ω_{k+1} erzeugt, Abwärts- oder Splittingschritt genannt. Zu jedem Splittingschritt existiert ein G -Homomorphismus $\varphi : \Omega_{k+1} \rightarrow \Omega_k$, der jede Nebenklasse $A_{k+1}g \in \Omega_{k+1}$ auf die Nebenklasse $\varphi(A_{k+1}g) = A_k g$ abbildet. Die Erweiterungsfunktion Υ_k sei für Splittingschritte so definiert, dass jede Nebenklasse $A_k g$ auf ihre Urbildmenge unter dem G -Homomorphismus φ abgebildet wird.

Fusingschritt

Ist hingegen $A_k \leq A_{k+1}$, so wird der Konstruktionsschritt, der aus der Menge Ω_k die Menge Ω_{k+1} erzeugt, Fusingschritt genannt. Zu jedem Fusingschritt existiert ein G -Homomorphismus $\varphi : \Omega_k \rightarrow \Omega_{k+1}$, der jede Nebenklasse $A_k g \in \Omega_k$ auf die Nebenklasse $\varphi(A_k g) = A_{k+1} g$ abbildet. Die Erweiterungsfunktion Υ_k sei für Fusingschritte so definiert, dass jede Nebenklasse $A_k g \in \Omega_k$ auf diejenige einelementige Teilmenge von Ω_{k+1} abgebildet wird, die nur die Nebenklasse $\varphi(A_k g) = A_{k+1} g$ enthält.

Zusammenfassend lässt sich die Erweiterungsfunktion Υ_k wie folgt beschreiben:

$$\forall k < n : \quad \Upsilon_k(A_k g) = \begin{cases} \{A_{k+1}h \mid h \in G \wedge A_k h = A_k g\}, & \text{falls } A_k \geq A_{k+1}, \\ \{A_{k+1}g\}, & \text{falls } A_k \leq A_{k+1}. \end{cases}$$

6.1.2 Ordnungsrelationen

Der im Folgenden vorgestellte Kanonizitätstest kann so implementiert werden, dass dieser in Tiefensuche abläuft und nur einzelne kanonische Repräsentanten im Speicher gehalten werden müssen. Dies wird durch die Anwendung der in Kapitel 3.3 beschriebenen Ordnungstreuen Erzeugung nach Read [Rea78] ermöglicht. Um die Ordnungstreue Erzeugung anwenden zu können, muss für alle

$1 \leq k \leq n$ auf jeder der Mengen Ω_k eine Ordnungsrelation R_k definiert werden. Jede dieser Relationen sei eine totale Ordnungsrelation. Sind die Gruppen aus $\{A_1, \dots, A_n\}$ paarweise verschieden, so sind die Mengen $\{\Omega_1, \dots, \Omega_n\}$ disjunkt. Daher kann aus den an der Relation beteiligten Elementen erschlossen werden, welche der Relationen aus $\{R_1, \dots, R_n\}$ gemeint ist. Zur Vereinfachung der Schreibweise wird daher im Folgenden davon ausgegangen, dass die Gruppen in $\{A_1, \dots, A_n\}$ paarweise verschieden sind und es wird nicht mehr explizit angegeben, welche der Relationen gemeint ist. Statt der Angabe $(\omega_1, \omega_2) \in R_k$ wird die Schreibweise $\omega_1 \preceq \omega_2$ verwendet. Die Schreibweise $\omega_1 \prec \omega_2$ wird im weiteren Text angewendet um auszudrücken, dass $\omega_1 \preceq \omega_2$ und $\omega_1 \neq \omega_2$ ist.

Die Ordnungen auf den Mengen von Nebenklassen müssen den Anforderungen genügen, die bei der Ordnungstreu Erzeugung nach Read an die Ordnungen auf den Objektmengen gestellt werden. Zur Formulierung dieser Anforderungen wird eine Menge von Funktionen $\{f_1, \dots, f_{n-1}\}$ verwendet, wobei für alle $k < n$ mit f_k eine Abbildung $f_k : \Omega_{k+1} \rightarrow \Omega_k$ bezeichnet wird. Für alle $k < n$ sei die Funktion f_k so definiert, dass diese jedes Objekt $\omega_{k+1} \in \Omega_{k+1}$ auf das kleinste Objekt $\omega_k \in \Omega_k$ abbildet, dessen Bild $\Upsilon_k(\omega_k)$, als Menge betrachtet, das Objekt ω_{k+1} enthält. Dann soll für alle $k \in \{1, \dots, n-1\}$ und $\omega, \omega' \in \Omega_{k+1}$ gelten:

$$\omega \prec \omega' \implies f_k(\omega) \preceq f_k(\omega')$$

Den Anforderung an die Ordnungsrelationen auf den Mengen $\Omega_1, \dots, \Omega_n$ kann leicht entsprochen werden, indem die Ordnung auf den Nebenklassenmengen so definiert wird, dass diese genau den Anforderungen genügt. Die Ordnungsrelation auf der Menge Ω_{k+1} wird rekursiv definiert, das heißt es wird davon ausgegangen, dass auf den Mengen Ω_1 bis Ω_k bereits Ordnungen definiert wurden, die den geforderten Anforderungen genügen. Weiterhin sei auf der Menge Ω_{k+1} bereits eine beliebige totale Ordnungsrelation R'_{k+1} gegeben.

Ist der Schritt von Ω_k nach Ω_{k+1} ein Fusingschritt, so ist das Bild der Erweiterungsfunktion Υ_k immer eine einelementige Menge. Für jede Nebenklasse $A_{k+1}g$ aus der Menge Ω_{k+1} gilt, dass die einelementige Menge $\{A_{k+1}g\}$ das

Bild der Nebenklasse $A_k g$ unter der Abbildung Υ_k ist. Die gesamte Urbildmenge der einelementigen Menge $\{A_{k+1}g\}$ unter der Abbildung Υ_k ist gleich der Menge $\{A_k g' \mid A_{k+1}g' = A_{k+1}g\} = \{A_k a g \mid a \in A_{k+1}\}$. Wird aus dieser Menge die kleinste Nebenklasse ausgewählt, so entspricht diese Nebenklasse genau dem Bild $f_k(A_{k+1}g)$. Da bei Fusionsschritten die Bildmenge der Erweiterungsfunktion Υ_k ausschließlich aus einelementigen Mengen besteht, muss die Abbildung f_k injektiv sein. Daher kann die Ordnung so definiert werden, dass zu gegebenen Elementen $\omega_1, \omega_2 \in \Omega_{k+1}$ gilt:

$$(\omega_1, \omega_2) \in R_{k+1} \iff (f_k(\omega_1), f_k(\omega_2)) \in R_k$$

Ist der Schritt von Ω_k nach Ω_{k+1} ein Splittingschritt, so bezeichne φ den diesem Splittingschritt zugeordneten G -Homomorphismus. Für alle Elemente $g, g' \in G$ gilt, dass $A_k g = \varphi(A_{k+1}g)$ und $A_k g' = \varphi(A_{k+1}g')$ ist. Bei der Definition der Ordnungsrelation sollen zwei Fälle unterschieden werden:

Wenn $A_k g \neq A_k g'$ ist, so muss auch $f_k(A_{k+1}g) \neq f_k(A_{k+1}g')$ sein und es soll wieder gelten:

$$(A_{k+1}g, A_{k+1}g') \in R_{k+1} \iff (f_k(A_{k+1}g), f_k(A_{k+1}g')) \in R_k$$

Andernfalls ist $f_k(A_{k+1}g) = f_k(A_{k+1}g')$ und es soll gelten:

$$(A_{k+1}g, A_{k+1}g') \in R_{k+1} \iff (A_{k+1}g, A_{k+1}g') \in R'_{k+1}$$

Es lässt sich leicht nachprüfen, dass eine auf diese Weise definierte Ordnungsrelation den von Read geforderten Bedingungen genügt.

6.1.3 Kanonizitätsprädikat

Operiert eine Gruppe B auf einer Menge Ω und ist $\chi : \Omega \longrightarrow \{0, 1\}$ eine Abbildung, die genau einem Element aus jeder Bahn der Gruppe B auf Ω den Wert 1 zuweist, so wird χ ein Kanonizitätsprädikat genannt. Im weiteren Text wird, immer wenn von kanonischen Bahnrepräsentanten die Rede ist, davon ausgegangen, dass ein Kanonizitätsprädikat zugrunde gelegt wurde, das dem nach der jeweiligen Ordnung kleinsten Objekt in jeder Bahn das Prädikat kanonisch zuschreibt. Diese Festlegung der kanonischen Repräsentanten genügt der Bedingung, die Read als Voraussetzung für die Ordnungstreue Erzeugung

an die Kanonizitätsprädikate stellt: Es soll gelten, dass für alle $1 \leq k < n$ und jedes kanonische Element $\omega \in \Omega_{k+1}$ ein kanonisches Element $\omega' \in \Omega_k$ existiert, mit $\omega \in \Upsilon_k(\omega')$. Dabei bezeichnet Υ_k die bereits beschriebene Erweiterungsfunktion.

Der Nachweis, dass die so festgelegten Kanonizitätsprädikate der geforderten Bedingung entsprechen, ist leicht zu erbringen. Für alle $1 \leq k < n$ bezeichne f_k wieder die Abbildung, die jedes $\omega \in \Omega_{k+1}$ auf die kleinste Nebenklasse in Ω_k abbildet, deren Bild unter der Abbildung Υ_k , als Menge betrachtet, die Nebenklasse ω enthält. Ist $\omega \in \Omega_{k+1}$ ein kanonischer Bahnrepräsentant, so ist auch $f_k(\omega)$ ein kanonischer Repräsentant. Denn angenommen $f_k(\omega)$ ist kein kanonischer Repräsentant, so existiert ein $b \in B$ so dass gilt: $f_k(\omega)^b \prec f_k(\omega)$. Es können dabei zwei Fälle unterschieden werden:

Im Falle, dass $\omega^b = \omega$ ist, muss der Konstruktionsschritt von Ω_k nach Ω_{k+1} ein Fusingschritt sein. Der zugehörige G -Homomorphismus soll mit φ bezeichnet werden und es gilt:

$$\varphi(f_k(\omega)^b) = \varphi(f_k(\omega))^b = \omega^b = \omega$$

Daher ist $\Gamma_k(f_k(\omega)^b) = \Gamma_k(f_k(\omega)) = \{\omega\}$ und $f_k(\omega)^b \prec f_k(\omega)$ widerspricht der Definition der Abbildung f_k .

Andernfalls ist $\omega^b \neq \omega$ und aufgrund der Bedingungen, die im vorangehenden Kapitel an die Ordnungen auf den Mengen Ω_k und Ω_{k+1} gestellt wurden, müsste auch $\omega^b \prec \omega$ gelten, was der Annahme, dass ω kanonisch ist, widerspricht.

Grundsätzlich kann der Kanonizitätstest so abgewandelt werden, dass dieser auch für andere Kanonizitätsprädikate verwendbar wird. In jedem Fall muss aber die Bedingung erfüllt sein, dass jeder kanonische Repräsentant aus einem vorangehenden kanonischen Repräsentanten erzeugt werden kann. Der Algorithmus muss dann so abgewandelt werden, dass auch diejenigen Blöcke untersucht werden, die ausschließlich Pfade enthalten, die größer als der kleinste bisher bekannte Pfad sind.

6.2 Konstruktionspfade

Um die theoretische Grundlage des Kanonizitätstests leichter verständlich zu machen, wird im Folgenden der Begriff Konstruktionspfad eingeführt. Im Algorithmus selbst werden die Konstruktionspfade nicht verwendet.

Das Tupel (A_1, \dots, A_n) bezeichne nach wie vor eine Leiter von der Gruppe $A_1 = G$ zur Gruppe A_n . Für alle $1 \leq i \leq n$ bezeichne Ω_i wieder die Menge der Rechtsnebenklassen $A_i \backslash G$. Die Mengen $\Omega_1, \dots, \Omega_n$ seien paarweise disjunkt und die Vereinigungsmenge K der Mengen Ω_1 bis Ω_n sei die Knotenmenge eines gerichteten Graphen. In diesem Graphen soll genau dann eine gerichtete Kante von einem beliebigen Knoten $\omega \in \Omega_k$ zu einem Knoten ω' bestehen, wenn ω' in der Menge $\Upsilon_k(\omega)$ liegt.

In der Graphentheorie wird eine Folge von Knoten v_1, \dots, v_{l+1} genau dann als gerichteter Pfad der Länge l bezeichnet, wenn zu je zwei aufeinander folgenden Knoten v_i und v_{i+1} eine gerichtete Kante von v_i nach v_{i+1} existiert.

Definition 15. *Es sei $1 \leq l < n$ und (v_1, \dots, v_{l+1}) sei ein Tupel mit der Eigenschaft, dass für alle $1 \leq j \leq l+1$ die Komponente v_j in der Menge Ω_j liegt. Wenn weiterhin gilt, dass für alle $1 \leq j \leq l$ die Komponente v_{j+1} auch in der Menge $\Upsilon_j(v_j)$ enthalten ist, so soll (v_1, \dots, v_{l+1}) Konstruktionspfad der Länge l genannt werden.*

Für die weiteren Überlegungen sind nur diejenigen Pfade des zuvor beschriebenen Graphen von Bedeutung, die auch Konstruktionspfade sind. Konstruktionspfade werden daher im weiteren Text auch kurz als Pfade bezeichnet, ohne explizit zu erwähnen, dass mit dem Begriff Pfad an dieser Stelle nur Konstruktionspfade gemeint sind. Im Folgenden wird die Menge aller Konstruktionspfade, deren letzte Komponente in der Menge Ω_k liegt, mit P_k bezeichnet.

Für jedes $1 < k \leq n$ kann auf der Menge P_k eine totale Ordnungsrelation definiert werden. Diese Ordnungsrelation basiert auf den Ordnungen, die auf den Mengen Ω_1 bis Ω_n gegeben sind. Wie schon bei den Nebenklassen wird für diese Relationen statt $(\rho, \rho') \in R$ die Schreibweise $\rho \preceq \rho'$ und statt $\rho \preceq \rho' \wedge \rho \neq \rho'$ die Schreibweise $\rho \prec \rho'$ verwendet.

Zu je zwei verschiedenen Pfaden $\rho = (\omega_1, \dots, \omega_k)$ und $\rho' = (\omega'_1, \dots, \omega'_k)$ exis-

tiert ein $1 \leq i \leq k$, so dass $\omega_j = \omega'_j$ für alle $j < i$ und $\omega_i \neq \omega'_i$ ist. Bezeichnet i diesen Index, so dass für alle $j < i$ und $\omega_j = \omega'_j$ und $\omega_i \neq \omega'_i$ ist, dann soll für ρ und ρ' gelten:

$$\rho \preceq \rho' \iff \omega_i \preceq \omega'_i$$

Da jede Totalordnung auch reflexiv ist, muss für alle Pfade ρ auch $\rho \preceq \rho$ gelten.

Lemma 3. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Dann ist für alle $g \in G$ und alle $1 < k \leq n$ das Tupel (A_1g, \dots, A_kg) ein Pfad.*

Beweis. Zeige, dass für alle $1 \leq i < k$ die Nebenklasse $A_{i+1}g$ in der Menge $\Upsilon_i(A_i g)$ enthalten ist. Wenn $A_i \leq A_{i+1}$ ist, so besteht die Menge $\Upsilon_i(A_i g)$ aus dem einzigen Element $A_{i+1}g$. Ist umgekehrt $A_i \geq A_{i+1}$ so ist die Menge $\Upsilon_i(A_i g) = \{A_{i+1}h \mid A_i h = A_i g\}$ und diese Menge enthält offensichtlich die Nebenklasse $A_{i+1}g$.

□

Lemma 4. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Für alle $1 < k \leq n$, alle Pfade $\rho = (A_1h_1, \dots, A_kh_k)$ und alle Elemente $g \in G$ ist $(A_1h_1, \dots, A_kh_k)^g = (A_1h_1^g, \dots, A_kh_k^g) = (A_1h_1g, \dots, A_kh_kg)$ wieder ein Pfad.*

Die Menge P_k ist daher abgeschlossen unter der Operation der Gruppe G , die durch Rechtsmultiplikation komponentenweise auf den Einträgen der Pfadtupel operiert.

Beweis. Zeige, dass zu allen Pfaden $\rho = (A_1h_1, \dots, A_kh_k)$ das Tupel $(A_1h_1g, \dots, A_kh_kg)$ wieder in P_k liegt. Dies ist genau dann der Fall, wenn für alle $j < k$ gilt, dass $A_{j+1}h_{j+1}g$ in der Menge $\Upsilon_j(A_jh_jg)$ liegt.

Ist $A_j \leq A_{j+1}$, so ist $\Upsilon_j(A_jh_j) = \{A_{j+1}h_j\}$. Da ρ ein Pfad ist muss die Nebenklasse $A_{j+1}h_{j+1}$ in der Menge $\Upsilon_j(A_jh_j)$ liegen und es gilt daher $A_{j+1}h_{j+1} = A_{j+1}h_j$. Daraus folgt direkt $A_{j+1}h_{j+1}g = A_{j+1}h_jg \in \Upsilon_j(A_jh_jg)$.

Ist $A_j \geq A_{j+1}$, so liegt, da ρ ein Pfad ist, die Nebenklasse $A_{j+1}h_{j+1}$ in der Menge $\Upsilon_j(A_jh_j) = \{A_{j+1}h' \mid A_jh' = A_jh_j\}$. Daher ist A_jh_{j+1} gleich A_jh_j und daher auch $\Upsilon_j(A_jh_jg)$ gleich $\Upsilon_j(A_jh_{j+1}g)$. Diese Menge $\Upsilon_j(A_jh_{j+1}g) = \{A_{j+1}h' \mid A_jh' = A_jh_{j+1}g\}$ enthält die Nebenklasse $A_{j+1}h_{j+1}g$.

□

6.3 Ordnungstreue Abbildungen

Es sei G wieder eine Gruppe und (A_1, \dots, A_n) sei eine Untergruppenleiter von $A_1 = G$ nach A_n . Zu allen $1 < k \leq n$ existiert eine Abbildung $\vartheta_k : \Omega_k \rightarrow P_k$, die jede Nebenklasse $\omega_k \in \Omega_k$ auf den kleinsten Pfad $\rho = (\omega_1, \dots, \omega_k)$ abbildet, dessen k -te Komponente gleich ω_k ist. Die Abbildung ϑ_k ist für alle $1 < k \leq n$ wohldefiniert, denn nach Lemma 3 existiert zu jeder Nebenklasse $A_k g \in \Omega_k$ mindestens ein Pfad $\rho = (A_1 g, \dots, A_k g)$, dessen k -te Komponente gleich $A_k g$ ist. Die Abbildung ϑ_k ist ordnungserhaltend, in dem Sinne, dass für alle $\omega, \omega' \in \Omega_k$ mit $\omega \preceq \omega'$ die Bildelemente den Urbildern entsprechend geordnet sind.

Lemma 5. *Es sei $1 < k \leq n$ und $\vartheta_k : \Omega_k \rightarrow P_k$ eine Abbildung, die jedes $\omega \in \Omega_k$ auf den kleinsten Pfad in P_k abbildet, dessen letzte Komponente gleich ω ist. Dann gilt für alle $\alpha, \beta \in \Omega_k$:*

$$\alpha \preceq \beta \implies \vartheta_k(\alpha) \preceq \vartheta_k(\beta)$$

Beweis. Die Aussage wird gezeigt durch Induktion nach der Pfadlänge.

Induktionsanfang: $k = 2$

Für alle $\alpha, \beta \in \Omega_2$ existiert jeweils nur ein einziger Pfad in P_2 , dessen zweite Komponente gleich α beziehungsweise β ist. Denn nach Definition 11 ist die erste Gruppe A_1 der Leiter (A_1, \dots, A_n) gleich G , daher ist $\Omega_1 = G \setminus G$ eine einelementige Menge. Bezeichnet ω das einzige Element der Menge Ω_1 , so ist $\vartheta_2(\alpha) = (\omega, \alpha)$ und $\vartheta_2(\beta) = (\omega, \beta)$. Aus der Definition der Pfadordnung folgt $\alpha \preceq \beta \iff \vartheta_2(\alpha) \preceq \vartheta_2(\beta)$.

Induktionsschluss: $k \Rightarrow k + 1 \quad \forall k < n$

Es seien α und β zwei Elemente aus der Menge Ω_{k+1} . Ohne Beschränkung der Allgemeinheit sei $\alpha \preceq \beta$. Sind α und β identisch, so folgt daraus trivialerweise auch $\vartheta_{k+1}(\alpha) \preceq \vartheta_{k+1}(\beta)$. Betrachte daher im Folgenden den Fall $\alpha \neq \beta$.

Aufgrund der Eigenschaften der in Kapitel 6.1.2 definierten Ordnungen auf den Nebenklassenmengen Ω_1 bis Ω_n existiert ein Element $\omega_k \in \Omega_k$, so dass α in $\Upsilon_k(\omega_k)$ liegt und für alle $\omega'_k \in \Omega_k$ mit $\beta \in \Upsilon_k(\omega'_k)$ gilt, dass $\omega_k \preceq \omega'_k$ ist. Zu diesem Element ω_k sei $\rho = (\omega_1, \dots, \omega_k, \alpha)$ der eindeutig bestimmte Pfad, der in den ersten k Komponenten mit dem Pfad $\vartheta_k(\omega_k)$ übereinstimmt und dessen letzte Komponente gleich α ist.

Es soll gezeigt werden, dass ρ kleiner ist als jeder beliebige andere Pfad $\rho' = (\omega'_1, \dots, \omega'_k, \beta)$, dessen letzte Komponente gleich β ist. Im Falle, dass $\omega_k = \omega'_k$ ist, folgt dies direkt aus der Definition der Ordnung der Pfade. Andernfalls folgt mit der Induktionsvoraussetzung, dass $\vartheta_k(\omega_k) \preccurlyeq \vartheta_k(\omega'_k) \preccurlyeq (\omega'_1, \dots, \omega'_k)$ ist. Daher existiert ein $1 \leq i \leq k$, so dass $\omega_i \prec \omega'_i$ ist und für alle $j < i$ gilt, dass $\omega_j = \omega'_j$ ist. Daraus folgt sofort $(\omega_1, \dots, \omega_k, \alpha) \prec (\omega'_1, \dots, \omega'_k, \beta)$.

□

Lemma 6. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es sei $1 < k \leq n$ und $I = \{i_1, \dots, i_m\} \subseteq \{1, \dots, k\}$ eine Indexmenge mit $i_1 < \dots < i_m$. Es sei $\delta = (\delta_1, \dots, \delta_k)$ ein beliebiger Pfad aus P_k und Δ_δ bezeichne die Menge $\{(\delta'_1, \dots, \delta'_k) \in P_k \mid \forall i \in I : \delta'_i = \delta_i\}$. Dann existiert ein Pfad ρ in der Menge Δ_δ , der kleiner ist als jeder andere Pfad $\rho' = (\omega'_1, \dots, \omega'_k)$, dessen Komponente ω'_{i_1} an Position i_1 größer ist als δ_{i_1} .*

Beweis. Die Aussage wird gezeigt durch Induktion nach i_1 .

Induktionsanfang: $i_1 = 1$

Da nach der Definition der Untergruppenleiter die Gruppe $A_1 = G$ ist, sind die Voraussetzungen des Satzes im Fall $i_1 = 1$ nie gegeben. Denn die Menge Ω_1 besteht aus einer einzigen Nebenklasse, daher ist die erste Komponente bei allen Pfaden aus der Menge P_k identisch.

Induktionsschluss: $i_1 \Rightarrow i_1 + 1$

Es sei jetzt $1 < i_1 \leq k$ und $(\omega_1, \dots, \omega_{i_1}) \in P_{i_1}$ der kleinste Pfad aus der Menge P_{i_1} , dessen letzte Komponente ω_{i_1} gleich δ_{i_1} ist. Dann enthält die Menge Δ_δ einen Pfad $\rho = (\omega_1, \dots, \omega_{i_1}, \delta_{i_1+1}, \dots, \delta_k)$. Im Folgenden wird gezeigt, dass dieser Pfad ρ kleiner ist als jeder beliebige Pfad $\rho' = (\omega'_1, \dots, \omega'_k)$ mit $\delta_{i_1} \prec \omega'_{i_1}$. Der Pfad $(\omega'_1, \dots, \omega'_{i_1})$ bezeichne den Pfad, der durch Verkürzen des Pfades ρ' entsteht. Aus Lemma 5 folgt, dass $(\omega_1, \dots, \omega_{i_1})$ kleiner gleich $(\omega'_1, \dots, \omega'_{i_1})$ ist. Daher existiert ein $s \leq i_1$, so dass $\omega_t = \omega'_t$ für alle $t < s$ gilt und $\omega_s \prec \omega'_s$ ist. Daraus folgt sofort $(\omega_1, \dots, \omega_{i_1}, \delta_{i_1+1}, \dots, \delta_k) \preccurlyeq (\omega'_1, \dots, \omega'_{i_1}, \dots, \omega'_k)$.

□

6.4 Blöcke aus Konstruktionspfaden

Operiert die Gruppe G auf einer Menge P , so operiert G auch auf der Menge aller Teilmengen von P , indem zu jeder Teilmenge Δ von P und jedem Gruppenelement $g \in G$ festgelegt wird, dass $\Delta^g = \{\delta^g \mid \delta \in \Delta\}$ ist.

Wenn für eine Teilmenge Δ von P und für alle $g \in G$ gilt, dass entweder $\Delta^g = \Delta$ oder $\Delta^g \cap \Delta = \emptyset$ ist, so wird Δ Block genannt.

Lemma 7. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es sei $1 < k \leq n$ und $I = \{i_1, \dots, i_m\} \subseteq \{1, \dots, k\}$ eine Indexmenge. Die Menge $\Delta_I = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall i \in I : g_i \in A_i\}$ ist ein Block, dessen Stabilisator G_{Δ_I} gleich der Untergruppe $A_{i_1} \cap \dots \cap A_{i_m} = \{h \in G \mid \forall i \in I : h \in A_i\}$ ist.*

Beweis. Zeige zuerst, dass die Menge Δ_I ein Block ist. Für alle $h \in G$ ist $\Delta_I^h = \{(A_1 g_1 h, \dots, A_k g_k h) \in P_k \mid \forall i \in I : g_i \in A_i\} = \{(A_1 g_1 h, \dots, A_k g_k h) \in P_k \mid \forall i \in I : g_i h \in A_i h\} = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall i \in I : A_i g_i = A_i h\}$. Existiert zu einem $h \in G$ ein Pfad $\rho \in \Delta_I \cap \Delta_I^h$, so folgt daraus, dass für alle $i \in I$ die Nebenklasse $A_i h$ gleich A_i ist. Daraus folgt sofort, dass Δ_I^h gleich Δ_I ist.

Zeige jetzt, dass $G_{\Delta_I} \subseteq A_{i_1} \cap \dots \cap A_{i_m}$ ist:

Es bezeichne ρ den Pfad $(A_1, \dots, A_k) \in \Delta_I$. Angenommen es existiert ein $i \in I$ und ein $h \in G_{\Delta_I}$ mit $h \notin A_i$. Dann müsste der Pfad $\rho^h = (A_1 h, \dots, A_k h)$ wegen $h \in G_{\Delta_I}$ wieder in Δ_I liegen. Dies ist aber nicht der Fall, denn $A_i h \neq A_i$.

Zeige jetzt, dass $G_{\Delta_I} \supseteq A_{i_1} \cap \dots \cap A_{i_m}$ ist: Es bezeichne ρ wieder den Pfad $(A_1, \dots, A_k) \in \Delta_I$ und es bezeichne h ein Element aus der Schnittmenge $A_{i_1} \cap \dots \cap A_{i_m}$. Für alle $i \in I$ gilt $A_i h = A_i$, denn h liegt in A_i . Der Pfad $\rho^h = (A_1 h, \dots, A_k h)$ liegt daher auch in der Menge Δ_I . Da Δ_I ein Block ist, folgt daraus sofort $\Delta_I^h = \Delta_I$.

□

Lemma 8. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es sei $1 < k \leq n$, $I = \{i_1, \dots, i_l\} \subseteq \{1, \dots, k\}$ eine Indexmenge und $\Delta_I = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall i \in I : g_i \in A_i\}$ ein Block. Es sei $J = \{j_1, \dots, j_m\} \subseteq I$ eine weitere Indexmenge und $\Delta_J = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall j \in J : g_j \in A_j\}$ ein weiterer Block.*

Dann existiert eine Abbildung $\varphi_{(I,J)} : \Delta_I^G \longrightarrow \Delta_J^G$, die jeden Block Δ_I^g in

der Bahn von Δ_I auf den Block Δ_J^g abbildet. Diese Abbildung $\varphi_{(I,J)}$ ist ein G -Homomorphismus.

Beweis.

$$\forall g \in G : \quad \varphi_{(I,J)}(\Delta_I^g) = \Delta_J^g = \varphi_{(I,J)}(\Delta_I)^g$$

□

Lemma 9. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es sei $1 < k \leq n$, $I = \{i_1, \dots, i_l\} \subseteq \{1, \dots, k\}$ eine Indexmenge und $\Delta_I = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall i \in I : g_i \in A_i\}$ ein Block. Es sei $J = \{j_1, \dots, j_m\} \subseteq I$ eine weitere Indexmenge und $\Delta_J = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall j \in J : g_j \in A_j\}$ ein weiterer Block.*

Ist der Stabilisator G_{Δ_I} des Blocks Δ_I gleich dem Stabilisator G_{Δ_J} des Blockes Δ_J , so ist die Abbildung $\varphi_{(I,J)} : \Delta_I^G \longrightarrow \Delta_J^G$, die für alle $g \in G$ den Block Δ_I^g auf den Block Δ_J^g abbildet, ein G -Isomorphismus.

Beweis. Die Abbildung $i_I : \Delta_I^G \longrightarrow G_{\Delta_I} \backslash G$, die jeden Block Δ_I^g auf die Nebenklasse $G_{\Delta_I} g$ abbildet, ist nach dem Fundamentallemma 1 ein G -Isomorphismus. Auch die Abbildung $i_J : \Delta_J^G \longrightarrow G_{\Delta_J} \backslash G$, die jeden Block Δ_J^g auf die Nebenklasse $G_{\Delta_J} g$ abbildet, ist genauso ein G -Isomorphismus. Da nach Voraussetzung $G_{\Delta_I} = G_{\Delta_J}$ ist und $i_J^{-1}(i_I(\Delta_I^g)) = \Delta_J^g$ ist, ist $\varphi_{(I,J)} = i_J^{-1} \circ i_I$. Die Abbildung $\varphi_{(I,J)}$ ist daher als Verknüpfung zweier G -Isomorphismen selbst ein G -Isomorphismus.

□

Ist aus dem Zusammenhang erkennbar, welches die zugrunde liegende Pfadmenge P_k ist, so soll im weiteren Text für alle Indexmengen I der Block $\{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall i \in I : g_i \in A_i\}$ mit Δ_I bezeichnet werden. Die in Lemma 8 beschriebenen G -Homomorphismen werden im weiteren Text mit der Schreibweise $\varphi_{(I,J)}$ bezeichnet.

Zu allen $1 < k \leq n$ und Indexmengen $I \subseteq \{1, \dots, k\}$ und Blöcken Δ_I soll jetzt auf der Bahn $\Delta_I^G = \{\Delta_I^g \mid g \in G\}$ eine Totalordnung R definiert werden. Es seien ω und ω' zwei beliebige Blöcke aus der Bahn Δ_I^G . Die Ordnung sei so definiert, dass genau dann $(\omega, \omega') \in R$ ist, wenn der Block ω einen Pfad ρ enthält, der kleiner oder gleich jedem anderen Pfad ρ' aus dem Block ω' ist:

$$(\omega, \omega') \in R \quad \Longleftrightarrow \quad (\exists \rho \in \omega \, \forall \rho' \in \omega' : \rho \preceq \rho')$$

Auch bei dieser Relation wird im weiteren Text die Schreibweise $\omega \preceq \omega'$ statt der Schreibweise $(\omega, \omega') \in R$ angewendet. Gilt $\omega \preceq \omega'$ und ist $\omega \neq \omega'$, so wird dies durch die Schreibweise $\omega \prec \omega'$ ausgedrückt.

6.5 Starke Untergruppenleiter

Es sei G wieder eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Für alle in dieser Arbeit beschriebenen Algorithmen, in denen eine Untergruppenleiter verwendet wird, ist es von Vorteil, wenn der Gruppenindex zwischen zwei in der Untergruppenleiter aufeinander folgenden Gruppen klein ist. Insbesondere muss für alle diese Algorithmen gelten, dass der Gruppenindex zweier aufeinander folgender Gruppen endlich sein muss.

Die in diesem Kapitel vorgestellten starken Untergruppenleitern bilden das Rückgrat aller in dieser Arbeit beschriebenen Algorithmen zur Berechnung und Prüfung von kanonischen Bahnrepräsentanten. Satz 6.5.2 ermöglicht die Konstruktion dieser starken Untergruppenleitern.

Definition 16. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Erfüllt das Tupel (A_1, \dots, A_n) die folgende Bedingung, so soll (A_1, \dots, A_n) eine starke Untergruppenleiter genannt werden:*

Für alle $1 < k \leq n$ und zu allen Pfaden (A_1g_1, \dots, A_kg_k) mit $g_1, \dots, g_k \in G$ existiere ein $g' \in G$, so dass gilt:

$$(A_1g', A_2g', \dots, A_kg') = (A_1g_1, \dots, A_kg_k)$$

Lemma 10. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von $G = A_1$ nach A_n . Ist die Leiter (A_1, \dots, A_n) eine starke Untergruppenleiter nach Definition 16, so gilt auch:*

$$\forall 1 \leq i < n \forall v \in A_i \exists u \in A_{i+1} : uv \in A_1 \cap \dots \cap A_i$$

Beweis. Es sei $i < n$, v ein beliebiges Element aus der Gruppe A_i und $g = v^{-1}$. Nach Lemma 3 und der Definition der Erweiterungsfunktion Υ_i in Kapitel 6.1 ist das Tupel $(A_1g, \dots, A_i g, A_{i+1})$ ein Pfad.

Da die Leiter (A_1, \dots, A_n) eine starke Untergruppenleiter ist, existiert nach Definition 16 ein $u \in G$, so dass gilt: $(A_1g, \dots, A_i g, A_{i+1}) = (A_1u, \dots, A_i u, A_{i+1}u)$. Wegen $A_{i+1} = A_{i+1}u$ liegt das Element u in der Gruppe A_{i+1} . Für alle $j \leq i$ ist

$A_j g = A_j u$ und daher auch $u g^{-1} \in A_j$. Für beliebige $j \leq i$ ist wegen $g = v^{-1}$ die Forderung $uv \in A_j$ erfüllt.

□

Satz 6.5.1. *Starke Untergruppenleiter*

Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von $G = A_1$ nach A_n . Die folgende Bedingung ist genau dann erfüllt, wenn die Untergruppenleiter (A_1, \dots, A_n) nach Definition 16 stark ist:

$$\forall 1 \leq i < n \forall v \in A_i \exists u \in A_{i+1} : \quad uv \in A_1 \cap \dots \cap A_i$$

Beweis. Nach Lemma 10 muss nur noch gezeigt werden, dass jede Untergruppenleiter, die die geforderte Bedingung erfüllt, auch stark ist. Durch Induktion nach der Pfadlänge k soll gezeigt werden, dass die Bedingung aus Definition 16 für alle $1 < k \leq n$ erfüllt ist:

$$\forall \rho \in P_k, \rho = (A_1 g_1, \dots, A_k g_k) \exists g' \in G : \rho = (A_1 g', A_2 g', \dots, A_k g')$$

Induktionsanfang: $k = 2$

Da $A_1 = G$ ist, ist die Bedingung für alle Pfade aus der Menge Ω_2 immer erfüllt. Denn für alle $g_1, g_2 \in G$ gilt: $(A_1 g_1, A_2 g_2) = (A_1 g_2, A_2 g_2)$.

Induktionsschritt: $k \Rightarrow k + 1 \quad \forall k < n$

Zu jedem Pfad $\rho = (A_1 g_1, \dots, A_{k+1} g_{k+1})$ existiert nach Induktionsvoraussetzung ein $g' \in G$, so dass $\rho = (A_1 g', \dots, A_k g', A_{k+1} g_{k+1})$ ist.

Ist $A_k \leq A_{k+1}$, so muss, da ρ ein Pfad ist, $A_{k+1} g_{k+1}$ in der Menge $\Upsilon_k(A_k g')$ liegen. Diese Menge enthält aber nur das einzige Element $A_{k+1} g'$, daher lässt sich ρ als $(A_1 g', \dots, A_k g', A_{k+1} g')$ schreiben.

Ist $A_k \geq A_{k+1}$, so muss, da ρ ein Pfad ist, $A_{k+1} g_{k+1}$ wieder in der Menge $\Upsilon_k(A_k g') = \{A_{k+1} h \mid A_k h = A_k g'\}$ liegen. Daher muss $A_k g_{k+1} = A_k g'$ sein und es existiert ein $v \in A_k$, so dass $v g' = g_{k+1}$ ist. Zu diesem $v \in A_k$ existiert ein $u \in A_{k+1}$, so dass $uv \in A_1 \cap \dots \cap A_k$ ist und daher gilt:

$$\begin{aligned} (A_1 g_1, \dots, A_k g_k, A_{k+1} g_{k+1}) &= (A_1 g', \dots, A_k g', A_{k+1} v g') \\ (A_1 g', \dots, A_k g', A_{k+1} v g') &= (A_1 u v g', \dots, A_k u v g', A_{k+1} u v g') \end{aligned}$$

□

Beispiel 1. Es sei G die Symmetrische Gruppe auf der Menge $\{1, \dots, 20\}$. Für alle Partitionen $(\lambda_1, \dots, \lambda_m)$ der Menge $\{1, \dots, 20\}$ bezeichne $S_{(\lambda_1, \dots, \lambda_m)} = G_{(\lambda_1, \dots, \lambda_m)}$ den Stabilisator in G , der jede Komponente des Tupels $(\lambda_1, \dots, \lambda_m)$ stabilisiert. Es gilt also $S_{(\lambda_1, \dots, \lambda_m)} = \{g \in S_{20} \mid \forall 1 \leq j \leq m : \lambda_j^g = \lambda_j\}$. Es sei:

$$\begin{aligned} A_1 &= G \\ A_{2k} &= S_{(\{1, \dots, k\}, \{k+1, \dots, 20\})} \quad \forall 1 \leq k \leq 10 \\ A_{2k+1} &= S_{(\{1, \dots, k\}, \{k+1\}, \{k+2, \dots, 20\})} \quad \forall 1 \leq k < 10 \end{aligned}$$

Dann ist (A_1, \dots, A_{20}) eine starke Untergruppenleiter von G nach A_{20} , da die in Satz 6.5.1 beschriebene Bedingung erfüllt ist:

Für alle $1 < k \leq n$ bilden die beiden Zyklen $(1, 2)$ und $(1, \dots, k)$ ein Erzeugendensystem der Gruppe $S_{(\{1, \dots, k\}, \{k+1, \dots, n\})}$. Gleiches gilt für die beiden Erzeuger $(k+1, k+2)$, $(k+1, \dots, n)$ und die Gruppe $S_{(\{1\}, \dots, \{k\}, \{k+1, \dots, n\})}$. Jeder der beiden Erzeuger $(1, 2)$, $(1, \dots, k)$ ist disjunkt zu jedem der beiden Erzeuger $(k+1, k+2)$, $(k+1, \dots, n)$. Da disjunkte Zyklen kommutieren, gilt für alle $g_1 \in S_{(\{1, \dots, k\}, \{k+1, \dots, n\})}$ und $g_2 \in S_{(\{1\}, \dots, \{k\}, \{k+1, \dots, n\})}$, dass $g_1 g_2 = g_2 g_1$ ist.

Für alle geraden Zahlen $i < 20$ ist A_{i+1} Untergruppe der Gruppe A_i . Es sei $k = \frac{i}{2}$, $N_1 = S_{(\{1, \dots, k\}, \{k+1, \dots, 20\})}$ und $N_2 = S_{(\{1\}, \dots, \{k\}, \{k+1, \dots, 20\})}$. Die Menge $N_1 N_2 = \{n_1 n_2 \mid n_1 \in N_1, n_2 \in N_2\}$ enthält das neutrale Element und zu jedem Element $n_1 n_2$ auch dessen Inverses $n_1^{-1} n_2^{-1} = n_2^{-1} n_1^{-1}$. Daher ist $N_1 N_2$ mit der Gruppenverknüpfung von G eine Gruppe, die identisch ist mit der Gruppe A_i . Die Gruppe $U = S_{(\{1\}, \dots, \{k+1\}, \{k+2, \dots, 20\})}$ ist eine Untergruppe von N_2 , daher gilt ebenfalls für alle $n_1 \in N_1$ und $u \in U$, dass $n_1 u = u n_1$ ist. Die Gruppe A_{i+1} ist mit der Gruppe $N_1 U = \{n_1 u \mid n_1 \in N_1, u \in U\}$ identisch.

Zu jedem $v \in A_i$ existieren daher Elemente $n_1 \in N_1$, $n_2 \in N_2$ mit $v = n_1 n_2$ und ein Element $u = n_1^{-1} \in N_1 \leq A_{i+1}$, so dass $uv = n_1^{-1} n_1 n_2 = n_2$ in der Gruppe N_2 liegt. Für alle $j \leq i$ ist N_2 eine Untergruppe von A_j , daher ist uv in der Gruppe A_j enthalten.

Für ungerade Zahlen $1 < i < 20$ ist die Gruppe A_i Untergruppe der Gruppe A_{i+1} und jedes $v \in A_i$ liegt daher auch in der Gruppe A_{i+1} . Zu jedem $v \in A_i$ liegt auch das inverse Element $u = v^{-1}$ in der Gruppe A_{i+1} und $uv = 1_G$ ist in jeder der Gruppen A_1, \dots, A_i enthalten.

Und für $i = 1$ existiert, da $A_1 = G$ ist, trivialerweise auch für alle $v \in A_i$ ein $u \in A_{i+1}$, so dass uv in A_i liegt.

Lemma 11. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von $G = A_1$ nach A_n . Die Leiter (A_1, \dots, A_n) ist genau dann eine starke Untergruppenleiter, wenn für alle $1 < k \leq n$ gilt, dass die Gruppe G transitiv auf der Pfadmenge P_k operiert.*

Beweis. „ \Leftarrow “ Es sei $1 < k \leq n$ und (A_1g_1, \dots, A_kg_k) ein beliebiger Pfad. Nach Lemma 3 ist auch (A_1, \dots, A_k) ein Pfad. Operiert die Gruppe G transitiv auf der Menge P_k , so existiert ein $h \in G$, so dass $(A_1, \dots, A_k)^h = (A_1g_1, \dots, A_kg_k)$ ist. Daraus folgt aber sofort $(A_1g_1, \dots, A_kg_k) = (A_1h, \dots, A_kh)$.

„ \Rightarrow “ Es seien zu einem $1 < k \leq n$ die Pfade ρ und ρ' aus der Menge P_k gegeben. Nach Definition 16 existieren zu den Pfaden ρ und ρ' Elemente $g, g' \in G$, so dass $\rho = (A_1g, \dots, A_kg)$ ist und $\rho' = (A_1g', \dots, A_kg')$ ist. Nach Lemma 4 operiert G auf der Menge P_k durch komponentenweise Operation.

Setze $h = g^{-1}g'$ dann gilt:

$$\rho^h = (A_1g, \dots, A_kg)^h = (A_1gh, \dots, A_kgh) = (A_1g', \dots, A_kg') = \rho'$$

□

Lemma 12. *Es sei G eine Gruppe und (A_1, \dots, A_i) eine starke Untergruppenleiter von $A_1 = G$ nach A_i . Es sei A_{i+1} eine weitere Gruppe mit $A_i \leq A_{i+1}$. Dann ist die Untergruppenleiter $(A_1, \dots, A_i, A_{i+1})$ auch eine starke Untergruppenleiter.*

Beweis. Mit Satz 6.5.1 folgt, dass $(A_1, \dots, A_i, A_{i+1})$ genau dann stark ist, wenn für alle $v \in A_i$ ein Element $u \in A_{i+1}$ existiert, so dass $uv \in A_1 \cap \dots \cap A_i$ ist. Für alle $v \in A_i$ setze $u = v^{-1} \in A_{i+1}$ dann gilt $uv \in A_1 \cap \dots \cap A_i$.

□

Lemma 13. *Es sei G eine Gruppe und (A_1, \dots, A_i) eine starke Untergruppenleiter von G nach A_i . Es sei A_{i+1} eine Untergruppe von A_i . Es bezeichne C die Gruppe $A_1 \cap \dots \cap A_i$ und es bezeichne D die Gruppe $A_{i+1} \cap C$.*

Die Leiter (A_1, \dots, A_{i+1}) ist genau dann eine starke Untergruppenleiter, wenn eine bijektive Abbildung $\gamma : D \setminus A_{i+1} \longrightarrow C \setminus A_i$, mit $\gamma(Da) = Ca$ für alle $a \in A_{i+1}$, existiert.

Beweis. Zeige zuerst die Wohldefiniertheit der Abbildung.

Die Gruppe D ist eine Untergruppe von C daher existiert ein A_i -Homomorphismus $\gamma' : D \setminus A_i \rightarrow C \setminus A_i$, mit $\gamma'(Da) = Ca$ für alle $a \in A_i$. Die Abbildung γ ist wohldefiniert, da γ genau dem auf die Teilmenge $D \setminus A_{i+1} \subseteq D \setminus A_i$ eingeschränkten A_i -Homomorphismus γ' entspricht.

Zeige die Injektivität der Abbildung γ . Für je zwei Elemente $a_1, a_2 \in A_{i+1}$ mit $\gamma(Da_1) = \gamma(Da_2)$ muss $Ca_1 = Ca_2$ gelten. Daraus folgt $a_1a_2^{-1} \in C$ und da $a_1, a_2 \in A_{i+1}$ sind, muss auch $a_1a_2^{-1} \in D$ gelten. Daraus kann wiederum $Da_1 = Da_2$ gefolgert werden.

Zeige, dass die Abbildung γ surjektiv sein muss, wenn (A_1, \dots, A_{i+1}) eine starke Untergruppenleiter ist. Es sei $a \in A_i$ ein beliebiges Element und Ca die zugehörige Nebenklasse aus der Menge $C \setminus A_i$. Es bezeichne $v \in A_i$ das inverse Element zu a . Wenn (A_1, \dots, A_{i+1}) eine starke Untergruppenleiter ist, existiert nach Satz 6.5.1 ein Element $u \in A_{i+1}$, so dass uv in der Gruppe C enthalten ist. Aus $uv \in C$ folgt $Cu = Cv^{-1}$ und für die Nebenklasse Du aus der Menge $D \setminus A_{i+1}$ gilt $\gamma(Du) = Cu = Cv^{-1} = Ca$.

Zeige, dass aus der Surjektivität der Abbildung γ folgt, dass (A_1, \dots, A_{i+1}) eine starke Untergruppenleiter ist. Nach Voraussetzung ist (A_1, \dots, A_i) bereits eine starke Untergruppenleiter. Unter Verwendung von Satz 6.5.1 bleibt nur noch zu zeigen, dass für alle $v \in A_i$ ein Element $u \in A_{i+1}$ mit $uv \in C$ existiert. Aufgrund der Surjektivität existiert zur Nebenklasse Cv^{-1} ein Element $u \in A_{i+1}$, so dass $\gamma(Du) = Cu = Cv^{-1}$ ist. Daraus folgt sofort $Cuv = C$ und weiterhin, dass uv in der Gruppe C enthalten sein muss.

□

Satz 6.5.2. *Es sei G eine Gruppe und (A_1, \dots, A_i) eine starke Untergruppenleiter von $A_1 = G$ nach A_i mit endlichem Gruppenindex $(G : A_1 \cap \dots \cap A_i)$. Es sei A_{i+1} eine Untergruppe von A_i mit endlichem Gruppenindex $(A_i : A_{i+1})$. Die Gruppe $A_1 \cap \dots \cap A_i$ wird mit C und die Gruppe $A_{i+1} \cap C$ wird mit D bezeichnet.*

Die Untergruppenleiter $(A_1, \dots, A_i, A_{i+1})$ ist genau dann eine starke Untergruppenleiter, wenn gilt:

$$(A_i : C) = (A_{i+1} : D)$$

Beweis. Aus der Endlichkeit der Gruppenindizes $(A_i : C)$ und $(A_i : A_{i+1})$ und dem Satz von Lagrange 3.1.1 folgt:

$$\begin{aligned}
 (A_i : A_{i+1}) &= (C : A_{i+1} \cap C) && \Longleftrightarrow \\
 (A_i : C) \cdot (A_i : A_{i+1}) &= (A_i : C) \cdot (C : A_{i+1} \cap C) && \Longleftrightarrow \\
 (A_i : C) \cdot (A_i : A_{i+1}) &= (A_i : A_{i+1} \cap C) && \Longleftrightarrow \\
 (A_i : C) \cdot (A_i : A_{i+1}) &= (A_i : A_{i+1}) \cdot (A_{i+1} : A_{i+1} \cap C) && \Longleftrightarrow \\
 (A_i : C) &= (A_{i+1} : A_{i+1} \cap C)
 \end{aligned}$$

„ \Rightarrow “ Ist $(A_1, \dots, A_i, A_{i+1})$ eine starke Untergruppenleiter, so folgt daraus die Bijektivität der Abbildung γ in Lemma 13. Aus der Bijektivität dieser Abbildung folgt wiederum, dass $(A_{i+1} : D) = (A_i : C)$ ist.

„ \Leftarrow “ Da D eine Untergruppe von C ist, existiert ein G -Homomorphismus $\gamma' : D \setminus A_i \rightarrow C \setminus A_i$, mit $\gamma'(Da) = Ca$ für alle $a \in A_i$. Die in Lemma 13 beschriebene Abbildung γ entspricht dem auf die Menge $D \setminus A_{i+1} \subset D \setminus A_i$ eingeschränkten G -Homomorphismus γ' .

Die Injektivität der Abbildung γ kann wie folgt gezeigt werden. Für je zwei Elemente $a_1, a_2 \in A_{i+1}$ mit $\gamma(Da_1) = \gamma(Da_2)$ muss $Ca_1 = Ca_2$ gelten. Daraus folgt $a_1 a_2^{-1} \in C$ und da $a_1, a_2 \in A_{i+1}$ sind, muss auch $a_1 a_2^{-1} \in D$ gelten. Daraus kann wiederum $Da_1 = Da_2$ gefolgert werden.

Wenn $(A_i : A_{i+1}) = (C : D)$ ist, dann muss auch $(A_i : C) = (A_{i+1} : D)$ gelten. Aus $(A_i : C) = (A_{i+1} : D)$ und der Injektivität der Abbildung γ folgt die Bijektivität der Abbildung γ . Mit Lemma 13 folgt, dass (A_1, \dots, A_{i+1}) eine starke Untergruppenleiter ist.

□

Satz 6.5.2 und Lemma 12 ermöglichen es, zu beliebigen Gruppen mit endlichem Gruppenindex eine starke Untergruppenleiter zu konstruieren. Nach Lemma 12 kann jede starke Untergruppenleiter um eine beliebige Gruppe A_{i+1} mit $A_i \leq A_{i+1}$ verlängert werden und die so verlängerte Leiter ist wieder stark. Soll eine starke Untergruppenleiter (A_1, \dots, A_i) um eine Gruppe $A_{i+1} \leq A_i$ verlängert werden, so kann die Menge aller maximalen Untergruppen von A_i berechnet werden. Jede Untergruppe aus dieser Menge, die die Bedingung aus Satz 6.5.2 erfüllt, kann dazu verwendet werden, die Leiter zu verlängern. Die neue Leiter, die auf diese Weise konstruiert wurde, ist nach Satz 6.5.2 wieder eine starke Untergruppenleiter.

Die Berechnung von maximalen Untergruppen ist in den meisten Fällen nicht einfach. Der Aufwand relativiert sich aber, da dieselbe Leiter zur Lösung vieler Probleme eingesetzt werden kann.

Soll aus den Untergruppen einer Permutationsgruppe G eine starke Untergruppenleiter konstruiert werden, so kann der folgende Satz bei der Konstruktion behilflich sein.

Satz 6.5.3. *Es sei S_n die Symmetrische Gruppe auf der Menge $\{1, \dots, n\}$, G eine Untergruppe der S_n und H eine Untergruppe von G . Für alle Partitionen $(\lambda_1, \dots, \lambda_m)$ der Menge $\{1, \dots, n\}$ bezeichne $S_{(\lambda_1, \dots, \lambda_m)}$ den Stabilisator in der Gruppe S_n , der jede Komponente des Tupels $(\lambda_1, \dots, \lambda_m)$ stabilisiert. Es gilt also $S_{(\lambda_1, \dots, \lambda_m)} = \{g \in S_n \mid \forall 1 \leq j \leq m : \lambda_j^g = \lambda_j\}$.*

Für alle $1 \leq i \leq 2k$ sei die Menge A_i wie folgt definiert:

$$A_1 = G$$

$$A_{2k} = \{g_1 g_2 \mid g_1 \in H \cap S_{(\{1, \dots, k\}, \{k+1\}, \dots, \{n\})}, g_2 \in G \cap S_{(\{1\}, \dots, \{k\}, \{k+1, \dots, n\})}\}$$

$$A_{2k+1} = \{g_1 g_2 \mid g_1 \in H \cap S_{(\{1, \dots, k\}, \{k+1\}, \dots, \{n\})}, g_2 \in G \cap S_{(\{1\}, \dots, \{k+1\}, \{k+2, \dots, n\})}\}$$

Dann ist jede der Mengen A_1, \dots, A_{2n} eine Untergruppe von G und das Tupel (A_1, \dots, A_{2n}) ist eine starke Untergruppenleiter von G nach $A_{2n} = H$.

Beweis. Für alle $1 < k \leq n$ bilden die beiden Zykeln $(1, 2)$ und $(1, \dots, k)$ ein Erzeugendensystem der Gruppe $S_{(\{1, \dots, k\}, \{k+1\}, \dots, \{n\})}$. Gleiches gilt für die beiden Erzeuger $(k+1, k+2)$, $(k+1, \dots, n)$ und die Gruppe $S_{(\{1\}, \dots, \{k\}, \{k+1, \dots, n\})}$. Jeder der beiden Erzeuger $(1, 2)$, $(1, \dots, k)$ ist disjunkt zu jedem der beiden Erzeuger $(k+1, k+2)$, $(k+1, \dots, n)$. Da disjunkte Zykeln kommutieren, gilt auch für alle $g_1 \in S_{(\{1, \dots, k\}, \{k+1\}, \dots, \{n\})}$ und $g_2 \in S_{(\{1\}, \dots, \{k\}, \{k+1, \dots, n\})}$, dass $g_1 g_2 = g_2 g_1$ ist. Daher enthält für alle $1 \leq j \leq 2n$ die Menge A_j nicht nur das neutrale Element, sondern enthält auch zu jedem Element dessen Inverses und ist somit eine Gruppe. Mit Hilfe der Bedingung aus Satz 6.5.1 kann jetzt nachgewiesen werden, dass die Untergruppenleiter (A_1, \dots, A_{2n}) stark ist:

Für alle geraden Zahlen $i < 2n$ ist die Gruppe A_{i+1} Untergruppe der Gruppe A_i . Es sei $k = \frac{i}{2}$ und es bezeichne N_1 die Gruppe $H \cap S_{(\{1, \dots, k\}, \{k+1\}, \dots, \{n\})}$ und N_2 die Gruppe $G \cap S_{(\{1\}, \dots, \{k\}, \{k+1, \dots, n\})}$. Dann ist $A_i = \{g_1 g_2 \mid g_1 \in N_1, g_2 \in N_2\}$. Die Gruppe A_{i+1} ist gleich $\{g_1 g_2 \mid g_1 \in N_1, g_2 \in N_2 \cap S_{(\{k+1\})}\}$. Zu jedem $v \in A_i$

existieren Elemente $g_1 \in N_1$ und $g_2 \in N_2$ mit $v = g_1 g_2$ und für das Element $u = g_1^{-1} \in N_1 \leq A_{i+1}$ gilt, dass $uv = g_1^{-1} g_1 g_2 = g_2$ in der Gruppe N_2 liegt. Für alle $j \leq i$ ist N_2 eine Untergruppe von A_j . Daher ist $uv = g_2$ auch in der Gruppe A_j enthalten.

Für ungerade Zahlen $1 < i < 2n$ ist die Gruppe A_i Untergruppe der Gruppe A_{i+1} und jedes $v \in A_i$ liegt daher auch in der Gruppe A_{i+1} . Zu jedem $v \in A_i$ liegt auch das inverse Element $u = v^{-1}$ in der Gruppe A_{i+1} und $uv = 1_G$ ist in jeder der Gruppen A_1, \dots, A_i enthalten.

Für die Gruppe A_1 gilt, dass zu jedem $v \in A_1 = G$ und jedem $u \in A_2 \leq G$ trivialerweise auch das Element uv in A_1 liegt.

□

Eine wichtige Eigenschaft der in Satz 6.5.3 beschriebenen Leiter soll an dieser Stelle noch einmal hervorgehoben werden. Beim Leiterspiel-Algorithmus werden schwierige Probleme gelöst, indem diese in viele kleine Splitting und Fusing Orbits Schritte zerlegt werden. Der Aufwand, der zur Lösung dieser Einzelschritte notwendig ist, hängt entscheidend davon ab, ob der Gruppenindex zweier aufeinander folgender Gruppen der Untergruppenleiter klein ist. Für die in Satz 6.5.3 beschriebene Untergruppenleiter und je zwei aufeinander folgende Gruppen A_i und A_{i+1} dieser Leiter gilt: Wenn $A_{i+1} \leq A_i$ ist, so ist der Index $(A_i : A_{i+1}) \leq n$. Wenn $A_i \leq A_{i+1}$ ist, so ist der Index $(A_{i+1} : A_i) \leq \frac{i+1}{2}$. Daher eignet sich die in Satz 6.5.3 beschriebene Leiter hervorragend für den Einsatz in einem Leiterspiel-Algorithmus.

Satz 6.5.3 ermöglicht es zu jeder beliebigen Untergruppe G der Symmetrischen Gruppe S_n und jeder Untergruppe H von G eine starke Untergruppenleiter von G nach H zu berechnen. Dies ermöglicht es, zu allen Gruppen $G \leq S_n$ und allen Untergruppen $A, B \leq G$ mit dem im Folgenden vorgestellten Algorithmus die kanonischen Nebenklasse aller Doppelnebenklassen $A \backslash G / B$ berechnen zu können.

Lemma 14. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es seien ein $k \leq n$ und ein $i < k$ mit $A_i \leq A_{i+1}$ gegeben. Es bezeichne E_i die Gruppe $A_i \cap A_k$ und E_{i+1} die Gruppe $A_{i+1} \cap A_k$.*

Ist die Leiter (A_1, \dots, A_n) eine starke Untergruppenleiter, so ist die Abbildung $\nu : E_i \setminus E_{i+1} \longrightarrow A_i \setminus A_{i+1}$, die jede Rechtsnebenklasse $E_i g \in E_i \setminus E_{i+1}$ auf die Rechtsnebenklasse $\nu(E_i g) = A_i g \in A_i \setminus A_{i+1}$ abbildet, bijektiv.

Beweis. Zeige zuerst, dass für alle $a \in A_{i+1}$ ein $e \in E_{i+1}$ mit $A_i a = A_i e$ existiert: Nach Lemma 3 ist $(A_1 a, \dots, A_n a)$ und auch (A_1, \dots, A_n) ein Pfad. Da $A_{i+1} a = A_{i+1}$ in der Menge $\Upsilon_i(A_i a)$ liegt, muss auch $(A_1 a, \dots, A_i a, A_{i+1}, \dots, A_k)$ ein Pfad sein. Nach Definition 16 existiert zu diesem Pfad auch ein Element $e \in G$, so dass $(A_1 a, \dots, A_i a, A_{i+1}, \dots, A_k) = (A_1 e, \dots, A_i e, A_{i+1} e, \dots, A_k e)$ ist. Da $A_{i+1} = A_{i+1} e$ und $A_k = A_k e$ ist, liegt e sowohl in der Gruppe A_k als auch in A_{i+1} und daher auch in E_{i+1} .

Daraus folgt, dass die Abbildung ν surjektiv ist, denn zu jeder Nebenklasse $A_i a \in A_i \setminus A_{i+1}$ existiert ein $e \in E_{i+1}$ mit $\nu(E_i e) = A_i e = A_i a$.

Zeige jetzt, dass die Abbildung ν injektiv ist: Für alle $e_1, e_2 \in E_{i+1}$ mit $\nu(E_i e_1) = \nu(E_i e_2)$ gilt auch $A_i e_1 = A_i e_2$. Daher gilt auch $e_1 e_2^{-1} \in A_i$ und wegen $e_1, e_2 \in E_{i+1}$ liegt $e_1 e_2^{-1}$ auch in A_k . Daraus folgt aber $e_1 e_2^{-1} \in A_i \cap A_k = E_i$ und es muss auch $E_i e_1 = E_i e_2$ gelten.

□

Satz 6.5.4. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es seien ein $k \leq n$ und ein $i < k$ mit $A_i \leq A_{i+1}$ gegeben. Es bezeichne E_i die Gruppe $A_i \cap A_k$ und E_{i+1} die Gruppe $A_{i+1} \cap A_k$.*

Ist (A_1, \dots, A_n) eine starke Untergruppenleiter, so sind die Gruppenindizes $(A_{i+1} : A_i)$ und $(E_{i+1} : E_i)$ gleich groß:

$$\frac{|A_{i+1}|}{|A_i|} = \frac{|E_{i+1}|}{|E_i|}$$

Beweis. Nach Lemma 14 existiert eine bijektive Abbildung zwischen der Menge $E_i \setminus E_{i+1}$ und der Menge $A_i \setminus A_{i+1}$. Daraus folgt sofort die Behauptung.

□

Im weiteren Text wird immer davon ausgegangen, dass die verwendeten Leitern starke Untergruppenleiter sind.

6.6 Urbilder von G -Homomorphismen

Es sei G wieder eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Für den im Folgenden beschriebenen Kanonizitätstest wird eine Methode benötigt, die es erlaubt, bei gegebenen Indexmengen $I \subseteq \{1, \dots, n\}$ und $J \subset I$ und gegebenem Block Δ_J^h die Menge der Urbilder von Δ_J^h unter dem in Lemma 8 beschriebenen G -Homomorphismus $\varphi_{(I,J)}$ zu bestimmen. Genauer ausgedrückt soll es möglich sein, zu allen $1 < k \leq n$ und allen $h \in G$ folgende Mengen bestimmen zu können:

- Es soll die Menge der Urbilder des Blockes $\Delta_{\{k\}}^h$ unter dem in Lemma 8 beschriebenen G -Homomorphismus $\varphi_{(\{1,k\},\{k\})}$ bestimmt werden können. Dies ermöglicht es, die maximale Teilmenge $\Gamma_1 \subseteq \Omega_1$ zu ermitteln, so dass für jedes $\gamma \in \Gamma_1$ ein Pfad in $\Delta_{\{k\}}^h$ existiert, an dessen erster Position die Nebenklasse γ steht.
- Zu allen $1 \leq i < k$ mit $A_i \geq A_{i+1}$ soll die Menge der Urbilder des Blockes $\Delta_{\{i,k\}}^h$ unter dem in Lemma 8 beschriebenen G -Homomorphismus $\varphi_{(\{i,i+1,k\},\{i,k\})}$ bestimmt werden können. Dies ermöglicht es, die maximale Teilmenge $\Gamma_{i+1} \subseteq \Omega_{i+1}$ zu ermitteln, so dass für jedes $\gamma \in \Gamma_{i+1}$ ein Pfad in $\Delta_{\{i,k\}}^h$ existiert, an dessen $i+1$ -ter Position die Nebenklasse γ steht.
- Zu allen $1 < i \leq k$ mit $A_{i-1} \leq A_i$ soll die Menge der Urbilder des Blockes $\Delta_{\{i,k\}}^h$ unter dem in Lemma 8 beschriebenen G -Homomorphismus $\varphi_{(\{i-1,i,k\},\{i,k\})}$ bestimmt werden können. Dies ermöglicht es, die maximale Teilmenge $\Gamma_{i-1} \subseteq \Omega_{i-1}$ zu ermitteln, so dass für jedes $\gamma \in \Gamma_{i-1}$ ein Pfad in $\Delta_{\{i,k\}}^h$ existiert, an dessen $i-1$ -ter Position die Nebenklasse γ steht.

Lemma 15. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es seien $1 \leq i < k \leq n$ so gewählt, dass $A_i \geq A_{i+1}$ ist. Es bezeichne E_i die Gruppe $A_i \cap A_k$ und E_{i+1} die Gruppe $A_{i+1} \cap A_k$.*

Dann ist $|E_{i+1} \setminus E_i| \leq |A_{i+1} \setminus A_i|$.

Beweis. Es sei $\varphi : E_{i+1} \setminus E_i \longrightarrow A_{i+1} \setminus A_i$ eine Abbildung mit $\varphi(E_{i+1}e) = A_{i+1}e$ für alle $e \in E_i$. Die Abbildung φ ist injektiv, denn für alle $e_1, e_2 \in E_i$ folgt aus $\varphi(E_{i+1}e_1) = \varphi(E_{i+1}e_2)$, dass $A_{i+1}e_1 = A_{i+1}e_2$ ist.

Dann ist $e_1 e_2^{-1} \in A_{i+1}$ und da $e_1, e_2 \in E_i$ sind, gilt auch $e_1 e_2^{-1} \in E_i$. Daher ist $e_1 e_2^{-1} \in A_{i+1} \cap E_i = A_{i+1} \cap A_i \cap A_k$ und da $A_{i+1} \leq A_i$ ist folgt $e_1 e_2^{-1} \in A_{i+1} \cap A_k = E_{i+1}$. Es gilt daher $E_{i+1} e_1 = E_{i+1} e_2$ und daraus folgt sofort die Behauptung. \square

Lemma 16. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Es sei $k \leq n$ und $I = \{i_1, \dots, i_m\} \subseteq \{1, \dots, k\}$ eine Indexmenge. Es sei $J \subseteq I$ eine weitere Indexmenge und $\varphi_{(I,J)}$ bezeichne den in Lemma 8 beschriebenen G -Homomorphismus $\varphi_{(I,J)} : \Delta_I^G \longrightarrow \Delta_J^G$. Für alle $g \in G$ enthält die Menge $\{\Delta_I^{hg} \mid \forall j \in J : h \in A_j\}$ genau diejenigen Blöcke aus der Menge Δ_I^G , die im Urbild des Blockes Δ_J^g unter dem G -Homomorphismus $\varphi_{(I,J)}$ liegen.*

Beweis. „ \subseteq “ Es sei ein $g \in G$ fest vorgegeben und $\Delta_I^{g'}$ ein Block, für den $\varphi_{(I,J)}(\Delta_I^{g'}) = \Delta_J^g$ ist. Da $\varphi_{(I,J)}$ ein G -Homomorphismus ist, gilt $\Delta_J^{g'g^{-1}} = \varphi_{(I,J)}(\Delta_I^{g'})^{g^{-1}} = \Delta_J^{gg^{-1}} = \Delta_{\{J\}}$. Daher liegt $h = g'g^{-1}$ im Stabilisator von $\Delta_{\{J\}}$ und nach Lemma 7 liegt h in $\bigcap_{j \in J} A_j$ und es gilt $\Delta_I^{g'} = \Delta_I^{g'g^{-1}g} = \Delta_I^{hg}$. „ \supseteq “ Es sei $g \in G$ wieder fest vorgegeben und $h \in \bigcap_{j \in J} A_j$. Dann ist $\varphi_{(I,J)}(\Delta_I^{hg}) = \varphi_{(I,J)}(\Delta_I)^{hg} = \Delta_J^{hg} = \Delta_J^g$. \square

Die in diesem Kapitel zu Beginn aufgeführten Urbilder und Mengen von Nebenklassen können alle mit Hilfe von Lemma 16 berechnet werden.

Lemma 17. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Es seien ein beliebiges Element g aus G und $i, k \leq n$ mit $i+1 < k$ und $A_i \leq A_{i+1}$ gegeben.*

Es bezeichne \mathcal{S} die Menge der Urbilder des Blockes $\Delta_{\{i+1,k\}}^g$ unter dem in Lemma 8 beschriebenen Homomorphismus $\varphi_{(\{i,i+1,k\},\{i+1,k\})}$. Weiterhin bezeichne \mathcal{T} die Menge der Urbilder des Blockes $\Delta_{\{i+1\}}^g$ unter dem in Lemma 8 beschriebenen Homomorphismus $\varphi_{(\{i,i+1\},\{i+1\})}$. Dann existiert eine bijektive Abbildung $\zeta : \mathcal{S} \longrightarrow \mathcal{T}$, die jeden Block $\Delta_{\{i,i+1,k\}}^{g'} \in \mathcal{S}$ auf den Block $\Delta_{\{i,i+1\}}^{g'} \in \mathcal{T}$ abbildet.

Beweis. Zu allen $a \in A_{i+1}$ existiert ein $e \in E_{i+1}$, so dass $A_i a = A_i e$ ist. Nach Lemma 3 ist sowohl das Tupel (A_1, \dots, A_k) als auch das Tupel $(A_1 a, \dots, A_k a)$ ein Pfad. Das Tupel $(A_1 a, \dots, A_i a, A_{i+1}, \dots, A_k)$ muss auch ein Pfad sein, da

$A_{i+1}a = A_{i+1}$ in der Menge $\Upsilon_i(A_i a)$ liegt. Nach Definition 16 existiert zu diesem Pfad ein $e \in G$, so dass $(A_1 a, \dots, A_i a, A_{i+1}, \dots, A_k) = (A_1 e, \dots, A_k e)$ ist. Da $A_{i+1} = A_{i+1}e$ und $A_k = A_k e$ ist, liegt e sowohl in der Gruppe A_k als auch in A_{i+1} und daher auch in E_{i+1} .

Zu jedem Block $\Delta_{\{i,i+1\}}^{g'} \in \mathcal{T}$ existiert nach Lemma 16 ein $a \in A_{i+1}$ mit $\Delta_{\{i,i+1\}}^{g'} = \Delta_{\{i,i+1\}}^{ag}$. Zu diesem $a \in A_{i+1}$ existiert, wie zuvor gezeigt wurde, ein $e \in E_{i+1}$ mit $ea^{-1} \in A_i \cap A_{i+1} = G_{\Delta_{\{i,i+1\}}}$. Daher ist $\Delta_{\{i,i+1\}}^{g'} = \Delta_{\{i,i+1\}}^{ag} = \Delta_{\{i,i+1\}}^{ea^{-1}ag} = \Delta_{\{i,i+1\}}^{eg} = \zeta(\Delta_{\{i,i+1,k\}}^{eg})$ und die Abbildung ζ ist surjektiv.

Zu allen $s_1, s_2 \in \mathcal{S}$ existieren nach Lemma 16 Elemente $e_1, e_2 \in E_{i+1}$ mit $s_1 = \Delta_{\{i,i+1,k\}}^{e_1g}$ und $s_2 = \Delta_{\{i,i+1,k\}}^{e_2g}$. Aus $\zeta(\Delta_{\{i,i+1,k\}}^{e_1g}) = \zeta(\Delta_{\{i,i+1,k\}}^{e_2g})$ folgt $\Delta_{\{i,i+1\}}^{e_1g} = \Delta_{\{i,i+1\}}^{e_2g}$ und daraus folgt wiederum $e_1 e_2^{-1} \in E_i$. Daher muss auch $\Delta_{\{i,i+1,k\}}^{e_1g}$ gleich $\Delta_{\{i,i+1,k\}}^{e_2g}$ sein und die Abbildung ζ ist injektiv und damit bijektiv. \square

Lemma 18. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine Untergruppenleiter von G nach A_n . Es sei $k \leq n$, $I \subseteq \{1, \dots, n\}$ eine Indexmenge und $\Delta_I = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall j \in I : g_j \in A_j\}$ ein Block. Es sei $J \subseteq I$ eine weitere Indexmenge und $\Delta_J = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall j \in J : g_j \in A_j\}$ ein weiterer Block. Es sei $\varphi_{(I,J)} : \Delta_I^G \longrightarrow \Delta_J^G$ eine Abbildung, die für jedes $g \in G$ den Block Δ_I^g auf den Block Δ_J^g abbildet.*

Für alle Elemente $g \in G$ bilden die als Pfadmengen betrachteten Blöcke im Urbild eines Blockes Δ_J^g unter der Abbildung $\varphi_{(I,J)}$ eine Partition der Menge Δ_I^g .

Beweis. Für alle $h \in G$ und Blöcke Δ_I^h gilt: $\Delta_I^h = \{(A_1 g_1 h, \dots, A_k g_k h) \in P_k \mid \forall i \in I : g_i \in A_i\} = \{(A_1 g_1 h, \dots, A_k g_k h) \in P_k \mid \forall i \in I : g_i h \in A_i h\} = \{(A_1 g_1, \dots, A_k g_k) \in P_k \mid \forall i \in I : A_i g_i = A_i h\}$. Angenommen es existiert im Block $\Delta_{\{J\}}^g$ ein Pfad $(A_1 g_1, \dots, A_k g_k)$, der in zwei Blöcken $\Delta_{\{I\}}^h$ und $\Delta_{\{I\}}^{h'}$ im Urbild von $\Delta_{\{J\}}^g$ enthalten ist. Dann muss für alle $i \in I$ gelten: $A_i h = A_i g_i = A_i h'$. Daraus folgt aber sofort $\Delta_{\{I\}}^h = \Delta_{\{I\}}^{h'}$. \square

Kapitel 7

Kanonizitätstests für Doppelnebenklassen

Die Fragestellung, ob zwei diskrete Strukturen „gleichgestaltig“ sind, wird in der Mathematik Isomorphieproblem genannt. Gleichgestaltig bedeutet in diesem Zusammenhang, dass eine strukturerhaltende Abbildung, ein sogenannter Homomorphismus zwischen den beiden Strukturen existiert. Diese Art von Problem ist meist nicht leicht zu lösen.

Ein prominenter Fall ist das Isomorphieproblem auf der Menge der Graphen, von dem nicht bekannt ist, ob es zur Klasse der NP -vollständigen Probleme gehört. Ein weiteres Problem, das auch zu den Isomorphieproblemen gehört, ist das Teilgraphenisomorphieproblem, bei dem entschieden werden muss, ob ein gegebener Graph ein Teilgraph eines anderen Graphen ist. Von diesem Problem ist bekannt, dass es zur Klasse der NP -vollständigen Probleme gehört und daher sehr schwer zu lösen ist. Sowohl das Graphenisomorphieproblem als auch das Teilgraphenisomorphieproblem können mit den in diesem Kapitel vorgestellten Algorithmen gelöst werden.

Unter Verwendung des Split Lemmas 3.5.1 können viele dieser Isomorphieprobleme auf eine Problemstellung aus dem Bereich der Doppelnebenklassen zurückgeführt werden. Dies ermöglicht es, denselben Lösungsansatz zur Lösung unterschiedlichster Probleme anzuwenden. In den Kapiteln 7.1 und 7.2 wird beschrieben, wie das Graphen- und das Teilgraphenisomorphieproblem mit Hilfe des Split Lemmas auf ein Doppelnebenklassenproblem abgebildet werden können.

Der ursprüngliche Algorithmus des Leiterspiels, wie er von Bernd Schmalz [Sch90] entwickelt wurde, ist außerordentlich effizient bei der Konstruktion von Doppelnebenklassen. Die Effizienz dieses Ansatzes ist jedoch an die Voraussetzung geknüpft, dass der Zugriff auf die im Speicher liegenden Zwischenergebnisse sehr schnell erfolgen kann. Da die Anforderungen an die Größe des Speicherplatzes jedoch enorm sind, bildet dies bei umfangreicheren Berechnungen häufig den limitierenden Faktor. Sobald im Arbeitsspeicher kein Platz mehr vorhanden ist und auf die viel langsamere Festplatte zugegriffen werden muss, stellt sich die Frage nach einer alternativen Strategie.

In diesem Kapitel werden drei Algorithmen beschrieben, die auf dem Leiterspiel aufbauen, aber mit sehr viel weniger Speicherplatz auskommen. Diese Algorithmen können genauso zur Konstruktion von Doppelnebenklassen eingesetzt werden.

Das Anwendungsfeld dieser neuen Algorithmen ist jedoch wesentlich breiter, da es mit jedem der drei Algorithmen möglich ist, auch für eine einzelne Nebenklasse zu prüfen, ob diese kanonisch ist. In Kapitel 7.6.3 wird weiterhin beschrieben, wie diese Kanonizitätstests zur Berechnung des kanonischen Repräsentanten der Bahn dieser Nebenklasse verwendet werden können. Beides ist mit der ursprünglichen Version des Leiterspiels so nicht möglich.

Der Breadthfirst Strong Ladders Leiterspiel (Breadthfirst StroLL) Algorithmus basiert genau wie der ursprüngliche Leiterspiel-Algorithmus von Schmalz auf einer Breitensuche, bei der alle Zwischenergebnisse im Speicher gehalten werden müssen. Der Speicherbedarf des Depthfirst StroLL ist trotzdem wesentlich geringer als der des ursprünglichen Leiterspiel-Algorithmus, da es die starken Leitern ermöglichen, die Anzahl der zu berechnenden Nebenklassen auf ein Minimum zu reduzieren. Im Gegensatz zum ursprünglichen Leiterspiel-Algorithmus nach Schmalz kann mit diesem Algorithmus der kanonische Repräsentant zu einer einzelnen Nebenklasse bestimmt werden.

Mit dem Depthfirst Strong Ladders Leiterspiel (Depthfirst StroLL) Algorithmus ist es möglich, für eine einzelne Nebenklasse in Tiefensuche zu bestimmen, ob diese kanonisch ist. Dieser Algorithmus hat einen sehr geringen

Speicherbedarf und ist auch zur Lösung von sehr großen Isomorphieproblemen geeignet, die sich aufgrund ihres Speicherbedarfs nicht mit dem Breadthfirst StroLL berechnen lassen.

Der Leiterspiel Light Algorithmus ist ein kleiner schneller Algorithmus, mit dem eine einzelne Nebenklasse daraufhin überprüft werden kann, ob diese kanonisch ist. Wie auch der Depthfirst StroLL hat auch das Leiterspiel Light nur einen sehr geringen Speicherbedarf. Das Leiterspiel Light zeichnet sich insbesondere durch seine Effizienz bei der Lösung von kleineren Isomorphieproblemen aus.

7.1 Teilgraphenisomorphieproblem und Doppelnebenklassen

Es seien zwei Graphen \mathcal{A} und \mathcal{B} auf endlich vielen Knoten gegeben und es soll die Frage beantwortet werden, ob der Graph \mathcal{A} einen Teilgraphen enthält, der isomorph zum Graphen \mathcal{B} ist. Diese Problemstellung wird Teilgraphenisomorphieproblem genannt und kann auf ein Doppelnebenklassenproblem abgebildet werden. Auf diese Weise kann das Teilgraphenisomorphieproblem mit Hilfe von jedem der drei neuen Algorithmen, die im Rahmen dieser Arbeit entstanden sind, gelöst werden. Das Teilgraphenisomorphieproblem ist insbesondere deshalb interessant, weil die Spezialfälle Cliquensuche und Hamiltonkreisproblem und daher auch das Teilgraphenisomorphieproblem selbst zur Klasse der NP-vollständigen Probleme gehören.

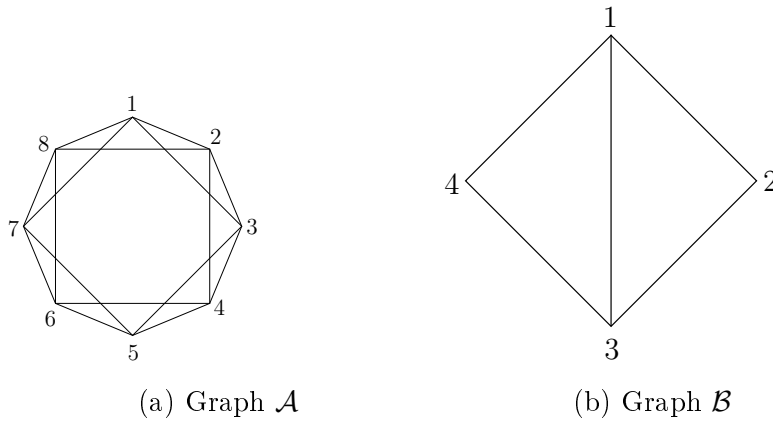
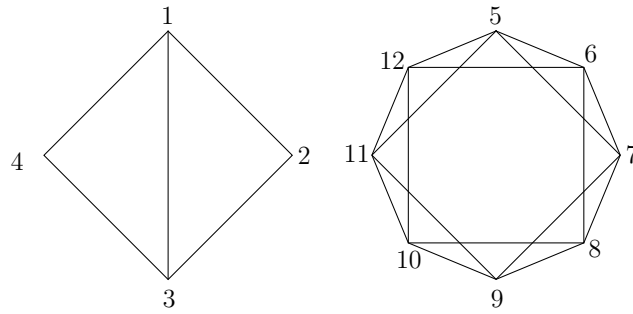


Abbildung 7.1: Zwei schlichte Graphen auf acht und auf vier Knoten

Es bezeichne (V_1, E_1) den Graphen \mathcal{A} und $k = |E_1|$ die Anzahl der Kanten des Graphen \mathcal{A} . Es bezeichne (V_2, E_2) den Graphen \mathcal{B} und $i = |E_2|$ die Anzahl der Kanten des Graphen \mathcal{B} . Die beiden Graphen \mathcal{A} und \mathcal{B} können zu einem einzigen Graphen \mathcal{C} vereinigt werden, der die beiden ursprünglichen Graphen als disjunkte Teilgraphen enthält, wie in Abbildung 7.2 dargestellt. Dieser Vereinigungsgraph \mathcal{C} besteht aus $m = |V_1| + |V_2|$ Knoten und $i + k$ Kanten.

Abbildungung 7.2: Vereinigungsgraph \mathcal{C} auf 12 Knoten

Der Graph \mathcal{C} ist isomorph zu einem Teilgraph des vollständigen Graphen auf m Knoten. Es bezeichne (V', E') einen der Teilgraphen im vollständigen Graphen auf m Knoten, der isomorph zum Graphen \mathcal{C} ist. Der Graph (V', E') kann in zwei disjunkte Teilgraphen zerlegt werden, so dass der eine Teil isomorph zum Graphen \mathcal{A} und der andere isomorph zum Graphen \mathcal{B} ist. Die $n = \binom{m}{2}$ Kanten des vollständigen Graphen sollen jetzt so auf die Zahlen 1 bis n abgebildet werden, dass die Kanten aus E' die Nummern 1 bis $i + k$ erhalten. Dabei sollen die Kanten der beiden disjunkten Teilgraphen so auf die Nummern 1 bis $i + k$ abgebildet werden, dass die Kanten des Teilgraphen von (V', E') , der isomorph zum Graphen \mathcal{A} ist, auf die Zahlen 1 bis k abgebildet werden. Weiterhin soll gelten, dass die Kanten des Teilgraphen von (V', E') , der isomorph zum Graphen \mathcal{B} ist, auf die Nummern $k + 1$ bis $i + k$ abgebildet werden.

Die weiteren Schritte laufen analog zum Beispiel in Kapitel 3.5 und werden daher nur sehr knapp erläutert. Die Gruppe S_m operiert durch Permutation der Knotennummierungen auf dem vollständigen Graphen. Analog zu Kapitel 3.5 operiert die Gruppe S_m auf der Menge der Kanten, indem jede Kante (v_1, v_2) von jedem Element $g \in S_m$ auf die Kante (v_1^g, v_2^g) abgebildet wird. Daher existiert ein Gruppenmonomorphismus φ von S_m auf eine Untergruppe B der S_n und ein Homomorphismus von Gruppenoperationen, so dass für alle Elemente $g \in S_m$ und alle Kanten ω des vollständigen Graphen $\omega^g = \omega^{\varphi(g)}$ gilt.

Es bezeichne

- λ_i die Partition $\{\{1, \dots, i\}, \{i + 1, \dots, n\}\}$,

- λ_k die Partition $\{\{1, \dots, k\}, \{k+1, \dots, n\}\}$,
- G die Gruppe S_n ,
- A_i den Stabilisator der Partition λ_i in G ,
- A_k den Stabilisator der Partition λ_k in G und
- $g \in S_n$ die Permutation $\begin{pmatrix} 1 & 2 & \dots & n-k & n-k+1 & n-k+2 & \dots & n \\ k+1 & k+2 & \dots & n & 1 & 2 & \dots & k \end{pmatrix}$.

Es bezeichne Ω die Menge aller Teilgraphen des gegebenen vollständigen Graphen, die aus m Knoten und i Kanten bestehen. Weiterhin bezeichne $\omega \in \Omega$ den Teilgraphen, bestehend aus den Kanten mit den Nummern 1 bis i . Die Gruppe G operiert durch Permutation der Kanten transitiv auf der Menge Ω und durch Rechtsmultiplikation transitiv auf der Menge $A_i \backslash G$. Nach Lemma 1 existiert zum Teilgraphen ω ein G -Isomorphismus, der zu allen $h \in G$ den Teilgraphen $\omega^h \in \Omega$ auf die Nebenklasse $A_i h$ abbildet.

Wenn die Menge $\{A_i h \mid h \in A_k\}$ eine Nebenklasse enthält, die in der Bahn von $A_i g$ unter der Operation der Gruppe B liegt, so enthält der Graph \mathcal{A} einen Teilgraphen, der isomorph ist zum Graphen \mathcal{B} . Ob dies der Fall ist, kann mit Hilfe des Algorithmus aus Kapitel 7.3 berechnet werden. Eine weitere Möglichkeit dies zu berechnen besteht darin, den Algorithmus in Kapitel 7.6.3 mit einem der beiden Algorithmen, die in den Kapiteln 7.4 und 7.5 beschrieben werden, zu kombinieren.

Für die Berechnungen wird eine starke Untergruppenleiter von G nach A_i benötigt. Diese kann beispielsweise, wie in Satz 6.5.3 beschrieben, gewählt werden. Unter Zuhilfenahme dieser starken Untergruppenleiter, kann mit jedem der drei Algorithmen sukzessive zu jeder Untergruppe U dieser Leiter die Menge $\{Uh \mid h \in A_k \wedge Uh \in Ug^B\}$ berechnet werden. Da die letzte Untergruppe dieser Leiter gleich A_i ist, ist $\{A_i h \mid h \in A_k \wedge A_i h \in A_i g^B\}$ die entsprechende Menge zur Untergruppe A_i .

Da aber die Menge $\{A_i h \mid h \in A_k \wedge A_i h \in A_i g^B\}$ isomorph zu einer Menge von Teilgraphen des Graphen \mathcal{A} ist, von denen jeder isomorph zum Graphen \mathcal{B} ist, kann auf diese Weise überprüft werden, ob der Graph \mathcal{A} den gesuchten Teilgraphen enthält.

7.2 Würfelfärbungen und Rechtsnebenklassen

Die in diesem Kapitel beschriebenen Algorithmen behandeln ausschließlich Isomorphieprobleme in Verbindung mit Nebenklassen. Der Zusammenhang zwischen den Würfelfärbungen aus Kapitel 5 und den Nebenklassen, die in den Beschreibungen der Algorithmen in diesem Kapitel verwendet werden, wird durch Lemma 1 und das Split Lemma 3.5.1 hergestellt.

Eine Färbung der Ecken eines Würfels, bei der jede Ecke entweder schwarz, weiß oder ungefärbt ist, kann wie folgt beschrieben werden: Die Würfecken werden mit den Zahlen 1 bis 8 gekennzeichnet. Anschließend kann die Färbung als Abbildung $g : \{1, \dots, 8\} \longrightarrow \{\text{schwarz}, \text{weiß}, \text{ungefärbt}\}$ von der Menge der Ecken in die Menge der Farben dargestellt werden.

Die Symmetrische Gruppe S_8 operiert auf der Menge $\{1, \dots, 8\}$. Jede Permutation kann als bijektive Abbildung $f : \{1, \dots, 8\} \longrightarrow \{1, \dots, 8\}$ dargestellt werden. Dann operiert die S_8 auf der Menge aller Würfelfärbungen durch Komposition $g \circ f$ der beiden jeweiligen Abbildungen.

In Anlehnung an Lemma 1 wird die S_8 in den weiteren Ausführungen mit G bezeichnet. Nach Lemma 1 existiert zu jeder Bahn von G auf der Menge der Würfelfärbungen und jedem Element ω aus dieser Bahn ein G -Isomorphismus, durch den jede Würfelfärbung aus dieser Bahn auf eine Rechtsnebenklasse abgebildet werden kann. Dieser G -Isomorphismus ermöglicht es, jede Situation, die im Beispiel aus Kapitel 5 anhand von Würfeln beschrieben wurde, auf eine entsprechende Situation im Zusammenhang mit Nebenklassen abzubilden.

Jede Bahn von G in der Menge der Würfelfärbungen ist eine Teilmenge, die aus allen Färbungen besteht, die dieselbe Anzahl schwarz und weiß gefärbter Ecken besitzen. Die Gruppe aller Drehungen des Würfels im Raum ist eine Untergruppe der S_8 die mit B bezeichnet wird. Im Beispiel aus Kapitel 5 operiert diese Untergruppe B auf jeweils einer der Bahnen der Gruppe G in der Menge der Würfelfärbungen.

Da B eine Untergruppe von G ist, ist der durch ein Bahnelement ω festgeleg-

te G -Isomorphismus nach Lemma 1 auch ein B -Isomorphismus. Daher ist der Stabilisator jeder Würfelfärbung in der Gruppe B identisch mit dem Stabilisator der entsprechenden Nebenklasse im Bild.

Es bezeichne Ω_1 und Ω_2 zwei Bahnen der Gruppe G in der Menge der Würfelfärbungen und $\psi : \Omega_1 \rightarrow \Omega_2$ einen B -Homomorphismus. Zu jedem Element ω aus einer der Mengen Ω_1 und Ω_2 bezeichne $\varphi_\omega : \omega^G \rightarrow G_\omega \backslash G$ den in Lemma 1 beschriebenen G -Isomorphismus zum Element ω , so dass $\varphi_\omega(\omega^g) = G_\omega g$ für alle $g \in G$ gilt. Weiterhin bezeichne $\pi : G_\omega \backslash G \rightarrow G_{\psi(\omega)} \backslash G$ für alle $\omega \in \Omega_1$ einen G -Homomorphismus, mit $\pi_\omega(G_\omega g) = G_{\psi(\omega)} g$ für alle $g \in G$. Dann kommutiert das Diagramm in Abbildung 7.3.

$$\begin{array}{ccc}
 \Omega_1 & \xrightarrow{\varphi_\omega} & G_\omega \backslash G \\
 \downarrow \psi & & \downarrow \pi_\omega \\
 \Omega_2 & \xrightarrow{\varphi_{\psi(\omega)}} & G_{\psi(\omega)} \backslash G
 \end{array}$$

Abbildung 7.3: Kommutatives Diagramm

Auf diese Weise können alle anhand von Würfelfärbungen beschriebenen Situationen und alle dazu verwendeten Homomorphismen von Gruppenoperationen aus Kapitel 5 auf entsprechende Situationen in Zusammenhang mit Nebenklassen abgebildet werden.

7.3 Breadthfirst StroLL Algorithmus

Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Für je zwei aufeinander folgende Gruppen der Untergruppenleiter sei der Gruppenindex klein und insbesondere endlich. Es sei eine Untergruppe B von G , ein Index $k \leq n$ und eine Nebenklasse $A_k q$ gegeben.

Für alle Indizes $j \leq n$ operiere die Gruppe B durch Rechtsmultiplikation auf der Menge der Nebenklassen $A_j \backslash G$. Der im Folgenden beschriebene Algorithmus ist dazu geeignet, zur Nebenklasse $A_k q$ den kanonischen Bahnrepräsentanten in der Bahn $A_k q^B = \{A_k qb \mid b \in B\}$ und dessen Stabilisator zu berechnen.

In diesem Kapitel wird ein Kanonisierungsalgorithmus beschrieben, mit dem zu einem gegebenen $k \leq n$ und einer gegebenen Nebenklasse $A_k q \in A_k \backslash G$ der kanonische Repräsentant der Bahn $A_k q^B$ berechnet werden kann. Anhand der Nebenklasse $A_k q$ soll gezeigt werden, wie der kanonische Repräsentant der Bahn $A_k q^B = \{A_k qb \mid b \in B\}$ berechnet werden kann. Dabei wird davon ausgegangen, dass bereits eine Untergruppe H des Stabilisators der Nebenklasse $A_k q$ in der Gruppe B bekannt ist.

Es bezeichne $J \subset \{1, \dots, n\}$ eine beliebige Indexmenge und Δ_J bezeichne, wie in Lemma 7 beschrieben, den entsprechenden Block zu dieser Indexmenge. Das Fundamentallema 1 ermöglicht es, einen G -Isomorphismus von der Menge $\{\Delta_J^g \mid g \in G\}$ in die Menge der Nebenklassen des Stabilisators G_{Δ_J} anzugeben, so dass für alle $g \in G$ der Block Δ_J^g auf die Nebenklasse $G_{\Delta_J} g$ abgebildet wird.

Aufgrund dieser Isomorphie wird im folgenden Text nicht immer genau zwischen der Betrachtung der Blöcke und der Betrachtung der entsprechenden Nebenklassen unterschieden. Immer dann, wenn ohne weitere Hinweise von Betrachtung der Blöcke zur Betrachtung der entsprechenden Nebenklassen übergegangen wird, wird dieser G -Isomorphismus zugrunde gelegt.

7.3.1 Überblick

Für alle Indizes $j \leq k$ bezeichne E_j die Untergruppe $A_j \cap A_k$. Die Untergruppenleiter $(E_1, \dots, E_{k-1}, A_k)$ soll dazu verwendet werden, zu der gegebenen Nebenklasse $A_k q$ den kanonischen Repräsentanten der Bahn $A_k q^B$ zu berechnen. Diese Leiter $(E_1, \dots, E_{k-1}, A_k)$ dient dazu, für alle $j < k$ die Menge T_j der kanonischen Bahnrepräsentanten von Bahnen der Gruppe qHq^{-1} in der Menge $E_j \setminus A_k$ zu konstruieren. Zu allen $j \leq k$ kann ein geeigneter Homomorphismus von Gruppenoperationen τ_j bestimmt werden, der die Bahnrepräsentanten aus der Menge T_j auf Nebenklassen aus der Menge $A_j \setminus G$ abbildet.

Zu jeder dieser Nebenklassen aus der Menge $A_j \setminus G$ soll dann der kanonische Repräsentant aus der Bahn dieser Nebenklasse unter der Operation der Gruppe B berechnet werden. Die Menge aller dieser kanonischen Bahnrepräsentanten wird mit R_j bezeichnet. Die Besonderheit an dieser Konstruktion ist, dass zur Berechnung der Bahnrepräsentanten aus der Menge R_j nur Bahnrepräsentanten aus der Menge R_{j-1} benötigt werden, die im vorangehenden Schritt auf dieselbe Weise berechnet wurden. Die im letzten Schritt berechnete Menge R_k enthält dann als einziges Element den gesuchten kanonischen Bahnrepräsentanten der Bahn $A_k q^B$.

7.3.2 Bestimmung der zu untersuchenden Nebenklassen

Jeder Konstruktionspfad, dessen letzte Komponente die Nebenklasse $A_k q$ ist, beschreibt einen Weg, wie diese Nebenklasse aus den vorangehenden Nebenklassen konstruiert werden kann. Der Block $\Delta_{\{k\}}^q$ enthält genau diejenigen Konstruktionspfade, deren letzte Komponente $A_k q$ ist. Für alle Indizes $i \leq k$ kann der Block $\Delta_{\{k\}}^q$ durch den G -Homomorphismus $\varphi_{(\{i,k\}, \{k\})}$ so in kleinere Blöcke im Urbild des Blockes $\Delta_{\{k\}}^q$ zerlegt werden, dass alle Pfade, deren i -te Komponenten übereinstimmen, jeweils in demselben Block liegen. Mit Hilfe von Lemma 16 kann die Menge der Urbilder von $\Delta_{\{i\}}^q$ unter dem Homomorphismus $\varphi_{(\{i,k\}, \{i\})}$ bestimmt werden.

Jeder Block im Urbild von $\Delta_{\{i\}}^q$ unter dem Homomorphismus $\varphi_{(\{i,k\}, \{i\})}$ kann durch den G -Homomorphismus $\varphi_{(\{i,k\}, \{i\})}$ auf einen Block aus der Menge $\Delta_{\{i\}}^G$ abgebildet werden. Indem jeder dieser Blöcke aus der Menge $\Delta_{\{i\}}^G$ durch den

im Fundamentallemma 1 beschriebenen G -Isomorphismus auf diejenige Nebenklasse aus der Menge Ω_i abgebildet wird, die genau der i -ten Komponente aller in diesem Block enthaltenen Pfade entspricht, kann für alle $i < k$ bestimmt werden, welche Nebenklassen aus der Menge Ω_i Komponenten eines Konstruktionspfades sind, dessen letzte Komponente $A_k q$ ist.

Die Menge aller Nebenklassen aus der Menge Ω_i , die in einem Konstruktionspfad vorkommen, dessen letzte Komponente die Nebenklasse $A_k q$ ist, ist daher genau die Menge $\{A_i h q \mid h \in A_k\}$. Für alle $1 \leq i \leq k$ wird die Menge $\{A_i h q \mid h \in A_k\}$ im folgenden Text mit Θ_i bezeichnet.

7.3.3 Ein Isomorphismus von Gruppenoperationen

Für alle $i \leq k$ existiert eine bijektive Abbildung $\tau_i : E_i \backslash A_k \longrightarrow \Theta_i$, die jede Rechtsnebenklasse $E_i h$ auf die Nebenklasse $\tau_i(E_i h) = A_i h q$ abbildet. Die Abbildung τ_i ist surjektiv, denn für jede Nebenklasse $A_i g \in \Theta_i$ existiert nach Definition der Menge Θ_i ein $h \in A_k$ mit $A_i h q = A_i g$. Daher ist $E_i h \in E_i \backslash A_k$ eine Nebenklasse mit $\tau_i(E_i h) = A_i h q = A_i g$. Die Abbildung τ_i ist auch injektiv, denn für alle $a_1, a_2 \in A_k$ mit $\tau_i(E_i a_1) = \tau_i(E_i a_2)$ ist $A_i a_1 q = A_i a_2 q$ und daher $a_1 q q^{-1} a_2^{-1} = a_1 a_2^{-1} \in A_i$. Weil a_1 und a_2 aus der Gruppe A_k sind, gilt $a_1 a_2^{-1} \in E_i$ und daraus folgt sofort $E_i a_1 = E_i a_2$ und es folgt die Injektivität. Die Gruppe A_k operiert durch Rechtsmultiplikation auf der Menge $E_i \backslash A_k$ und die Gruppe $q^{-1} A_k q$ operiert genauso durch Rechtsmultiplikation auf der Menge $\{A_i h q \mid h \in A_k\}$. Die auf die Untergruppe A_k eingeschränkte q -Konjugation in der Gruppe G ist ein Gruppenisomorphismus $\sigma : A_k \longrightarrow q^{-1} A_k q$ mit $\sigma(g) = q^{-1} g q$. Das Abbildungspaar (σ, τ_i) ist für alle $i \leq k$ ein Homomorphismus von Gruppenoperationen:

$$\begin{aligned} \forall a \in A_k \forall g \in A_k : \quad \tau_i(E_i a^g) &= \tau_i(E_i a g) &= A_i a g q \\ \tau_i(E_i a)^{\sigma(g)} &= A_i a q^{\sigma(g)} = A_i a q^{q^{-1} g q} &= A_i a g q \end{aligned}$$

Da für alle $i \leq k$ die Abbildung τ_i eine Bijektion und σ ein Gruppenisomorphismus ist, ist (σ, τ_i) sogar ein Isomorphismus von Gruppenoperationen.

7.3.4 Eine weitere Untergruppenleiter

Für alle $i \leq k$ wird die Gruppe $A_i \cap A_k$ mit E_i bezeichnet. Da die Gruppe A_1 nach Definition gleich G ist, ist $E_1 = A_k \cap G = A_k = E_k$. Das Tupel (E_1, \dots, E_k) ist daher eine Untergruppenleiter von A_k nach A_k . Es bezeichne weiterhin \tilde{H} die Gruppe qHq^{-1} . Die Gruppe H ist nach Voraussetzung eine Untergruppe des Stabilisators von A_kq in B und daher eine Untergruppe von $G_{A_kq} = q^{-1}A_kq$. Die Gruppe \tilde{H} ist eine Untergruppe von A_k und operiert daher auf jeder der Mengen $E_1 \setminus A_k$ bis $E_k \setminus A_k$.

Für alle $i \leq k$ wird die nach einem beliebigen Kanonizitätsprädikat ausgezeichnete Menge der kanonischen Bahnrepräsentanten der Bahnen von \tilde{H} in der Menge $E_i \setminus A_k$ mit \tilde{T}_i bezeichnet. An dieser Stelle wird zugunsten der Übersicht darauf verzichtet, zu erläutern, wie die Mengen $\tilde{T}_1, \dots, \tilde{T}_k$ berechnet werden können. Eine mögliche Technik zur Berechnung dieser kanonischen Repräsentanten ist, die Untergruppenleiter (E_1, \dots, E_k) zusammen mit dem ursprünglichen Leiterspiel-Algorithmus von Schmalz zur Berechnung der Doppelnebenklassenrepräsentanten aller Doppelnebenklassen in $E_i \setminus A_k / \tilde{H}$ zu verwenden. Die folgenden Kapitel enthalten weitere Varianten des Leiterspiel-Algorithmus, die es erlauben, die Repräsentanten $\tilde{T}_1, \dots, \tilde{T}_k$ auf anderem Wege zu berechnen.

7.3.5 Inklusions-Homomorphismus

Die Gruppe H ist eine Untergruppe von B , daher induziert die Inklusionsabbildung $\iota : H \rightarrow B$ mit $\iota(h) = h$ zusammen mit der Inklusionsabbildung $\kappa_i : \{A_i a q \mid a \in A_k\} \rightarrow A_i \setminus G$ mit $\kappa_i(A_i g) = A_i g$ einen Homomorphismus von Gruppenoperationen (ι, κ_i) .

Auch die Verkettung der beiden Homomorphismen von Gruppenoperationen $(\kappa_i \circ \tau_i, \iota \circ \sigma)$ ist ein Homomorphismus von Gruppenoperationen. Bei diesem Homomorphismus werden mehrere Bahnen der Gruppe \tilde{H} auf der Menge $E_i \setminus A_k$ zu einer einzigen Bahn der Gruppe B auf der Menge $A_i \setminus G$ vereinigt.

$$E_i \setminus A_k \xrightarrow[(\tau_i, \sigma)]{Iso.} \{A_i a q \mid a \in A_k\} \xrightarrow{(\kappa_i, \iota)} A_i \setminus G$$

Für alle $1 \leq i \leq k$ wird die Menge aller kanonischen Bahnrepräsentanten von Bahnen der Gruppe B , die ein Element aus der Menge Θ_i enthalten, mit $T_i = \{\delta^b \mid \delta \in \Theta_i \wedge \forall b' \in B : \delta^b \preceq \delta^{b'}\}$ bezeichnet. Die Menge T_k enthält nur ein einziges Element, dies ist der gesuchte kanonische Bahnrepräsentant der Bahn der Nebenklasse $A_k q$ unter der Operation der Gruppe B .

7.3.6 Fusionierende Abbildung

Für alle $1 < i \leq k$ wird neben der Menge T_i und der Menge aller Stabilisatoren der Elemente aus T_i eine fusionierende Abbildung $\eta_i : \{\delta^b \mid \delta \in \Theta_i, b \in B\} \rightarrow B$ benötigt. Diese fusionierende Abbildung ordnet jedem Element ω aus der Menge Ω_i , das in der Bahn δ^B eines Elementes $\delta \in \Theta_i$ liegt, ein fusionierendes Element $b \in B$ zu, so dass ω^b in der Menge T_i liegt.

Die fusionierende Abbildung η_i kann, um Speicherplatz zu sparen, aus einer Menge von Abbildungen f_1, \dots, f_i berechnet werden. Die Abbildung $f_1 : \Omega_1 \rightarrow B$ bildet das einzige Element der Menge Ω_1 auf das neutrale Element der Gruppe G ab. Für $1 < j \leq i$ entspricht die Funktion f_j einer auf die Menge $\{\omega \in \Omega_j \mid \exists t \in T_{j-1} : \omega \in \Upsilon_{j-1}(t)\}$ eingeschränkten fusionierenden Abbildung. Die Abbildungen f_1, \dots, f_i können schrittweise zusammen mit den kanonischen Repräsentanten der Mengen T_1, \dots, T_i berechnet werden.

Sind für ein $1 < i \leq k$ die Abbildungen f_1 bis f_i bekannt und soll zu einer gegebenen Nebenklasse $A_i g \in \{A_i h b \mid A_i h \in \Theta_i, b \in B\}$ das fusionierende Element berechnet werden, so kann die fusionierende Abbildung η_i wie folgt rekursiv bestimmt werden: Zuerst wird das Element $b_1 = \eta_{i-1}(A_{i-1}g) \in B$ berechnet. Dann ist $A_{i-1}g b_1$ der kanonische Repräsentant der Bahn $A_{i-1}g^B$ und die Nebenklasse $A_i g b_1$ liegt in der Definitionsmenge der Funktion f_i . Anschließend wird das Element $b_2 = f_i(A_i g b_1) \in B$ bestimmt. Dann ist das Element $b_1 b_2$ ein fusionierendes Element zur Nebenklasse $A_i g$. Auf diese Weise kann die fusionierende Abbildung η_i bestimmt werden.

7.3.7 Splitting Orbits

Ist für ein $i < k$ die Gruppe A_i eine Untergruppe der Gruppe A_{i+1} , so wird der Konstruktionsschritt, in dem aus der Menge T_i die Menge T_{i+1} berechnet wird, Splittingschritt genannt. Es seien die Mengen T_i, \tilde{T}_i , die fusionierende

Abbildung $\eta_i : \{\delta^b \mid \delta \in \Theta_i, b \in B\} \longrightarrow B$ und zu jedem $t \in T_i$ der Stabilisator $B_t = \{b \in B \mid t^b = t\}$ gegeben. Es sei die Abbildung $\tau_i : E_i \setminus A_k \longrightarrow \Theta_i$ gegeben, die jede Nebenklasse $E_i g \in E_i \setminus A_k$ auf die Nebenklasse $A_i g \in \Theta_i$ abbildet. Es seien weiterhin zwei Algorithmen gegeben, die zu einem gegebenen Element $\omega \in \Omega_i$ und einer gegebenen Gruppe $U \leq B$, deren Bahn ω^U kurz ist, Folgendes berechnen:

Der Algorithmus *FindOrbitRep*(ω, U) berechnet ein $u \in U$, so dass $\forall u' \in U : \omega^u \preceq \omega^{u'}$ gilt.

Der Algorithmus *ReduceStab*(ω, U) berechnet den Stabilisator von ω in U .

Vorgehen beim Splitting Orbits

1. Setze Σ gleich \tilde{T}_i , der Menge der Bahnrepräsentanten in $E_i \setminus A_k$, und $T_{i+1} = \emptyset$.
2. Wähle ein beliebiges Element s aus der Menge Σ aus und bestimme das fusionierende Element $b = \eta_i(\tau_i(s))$. Dann liegt das Element $t = \tau_i(s)^b$ in der Menge T_i .
3. Bestimme die Menge $\Pi = \{\gamma^b \mid \gamma \in \Upsilon_i(\tau_i(s)) \cap \Theta_{i+1}\}$. Die Menge Π enthält dann alle Elemente γ aus der Menge $\Upsilon_i(t)$, für die ein $\omega \in E_{i+1} \setminus A_k$ existiert, mit $\tau_{i+1}(\omega)^b = \gamma$.
4. Wähle ein beliebiges Element δ aus der Menge Π . Nach Voraussetzung ist das Kanonizitätsprädikat so gewählt, dass der kanonische Repräsentant der Bahn δ^B in der Menge $\Upsilon_i(t)$ liegen muss. Daher kann mit Hilfe des Algorithmus *FindOrbitRep*(δ, B_t) das fusionierende Element c zur Nebenklasse δ berechnet werden. Speichere den Wert c als Bildelement von δ unter der Abbildung f_{i+1} .
5. Ist $\delta^c = \delta$, so ist δ ein kanonischer Bahnrepräsentant seiner Bahn. Ist δ noch nicht in der Menge T_{i+1} enthalten, so füge δ zur Menge T_{i+1} hinzu.
6. Der Stabilisator von δ in B kann mit Hilfe des Algorithmus *ReduceStab*(δ, B_t) berechnet werden.

7. Entferne δ aus der Menge Π . Wenn die Menge Π danach nicht leer ist, fahre fort mit Schritt 4.
8. Entferne s aus der Menge Σ . Wenn die Menge Σ danach nicht leer ist, fahre fort mit Schritt 2.

Pseudocode zu Splitting Orbits

Mit dem folgenden Algorithmus können die eingeschränkte fusionierende Abbildung $f_{i+1} : \{A_{i+1}g \mid A_i g \in T\} \longrightarrow B$, die Menge der kanonischen Bahnrepräsentanten T_{i+1} und zu allen Elementen $t \in T_{i+1}$ der Stabilisator B_t bestimmt werden:

```

1    $T_{i+1} \leftarrow \emptyset$ 
2   foreach (  $s \in \tilde{T}_i$  )
3        $b \leftarrow \eta_i(\tau_i(s))$ 
4        $t \leftarrow \tau_i(s)^b$   $\backslash \backslash t \in T_i$ 
5        $\Pi \leftarrow \{\gamma \in \Upsilon_i(t) \mid \gamma^{b^{-1}} \in \Theta_{i+1}\}$ 
6       foreach (  $\delta \in \Pi$  )
7            $c \leftarrow FindOrbitRep(\delta, B_t)$ 
8            $f_{i+1}(\delta) \leftarrow c$ 
9           if (  $\delta^c == \delta$  )  $\backslash \backslash \Rightarrow \forall b \in B : \delta \preccurlyeq \delta^b$ 
10              if (  $\delta \notin T_{i+1}$  )
11                   $T_{i+1} \leftarrow T_{i+1} \cup \{\delta\}$ 
12                   $B_\delta \leftarrow ReduceStab(\delta, B_t)$ 
13              endif
14          endif
15      end
16  end

```

7.3.8 Fusing Orbits

Ist für ein $i < k$ die Gruppe A_{i+1} eine Untergruppe der Gruppe A_i , so wird der Konstruktionsschritt, in dem aus der Menge T_i die Menge T_{i+1} berechnet wird, Fusingschritt genannt. Es seien die Mengen T_i , die fusionierende Abbildung $\eta_i : \{\delta^b \mid \delta \in \Theta_i, b \in B\} \longrightarrow B$ und zu jedem $t \in T_i$ der Stabilisator $B_t = \{b \in B \mid t^b = t\}$ gegeben. Es sei die Abbildung $\tau_i : E_i \setminus A_k \longrightarrow \Theta_i$ gegeben, die jede Nebenklasse $E_i g \in E_i \setminus A_k$ auf die Nebenklasse $A_i g \in \Theta_i$ abbildet. Es sei weiterhin ein Algorithmus $ExtendGroup(v, U)$ gegeben, der zu einem gegebenen Element $v \in G$ und einer gegebenen Gruppe $U \leq B$ die kleinste Untergruppe von G berechnet, die sowohl die Untergruppe U als auch das Element v enthält.

Bestimmung der Bahnrepräsentanten

1. Setze Π gleich der Menge T_i und $T_{i+1} = \emptyset$.
2. Wähle ein beliebiges Element $A_i g$ aus der Menge Π aus und bestimme die Menge $\Sigma = \{A_i e g^b \mid e \in A_{i+1}, b = \eta_i(A_i e g)\}$. Damit entspricht die Menge Σ der Menge aller Bahnrepräsentanten $s \in T_i$ mit $\Upsilon_i(s) \subseteq A_{i+1} g^B$.
3. Bestimme das Element $t = \min_{s \in \Sigma} s$. Die Menge $\Upsilon_i(t)$ besteht aus dem kanonischen Bahnrepräsentant der Bahn $A_{i+1} g^B$. Füge daher das Element aus $\Upsilon_i(t)$ zur Menge T_{i+1} hinzu.
4. Entferne alle Elemente, die in der Menge Σ enthalten sind, aus der Menge Π . Wenn die Menge Π danach nicht leer ist, fahre fort mit Schritt 2.

Bestimmung der Stabilisatoren und der Abbildung f_{i+1}

Die Abbildung f_{i+1} bezeichnet die auf die Menge $\{\omega \in \Omega_{i+1} \mid \exists t \in T_i : \omega \in \Upsilon_i(t)\}$ eingeschränkte fusionierende Abbildung η_{i+1} . Mit Hilfe der Abbildungen f_1, \dots, f_{i+1} kann die auf der Menge Ω_{i+1} definierte fusionierende Abbildung η_{i+1} auf effiziente Weise berechnet werden.

Zu jeder Untergruppe U von B und zu jedem Element $b \in B$ sei ein Algorithmus $ExtendGroup(b, U)$ gegeben, der die kleinste Untergruppe von B bestimmt, die sowohl das Element b als auch die Gruppe U enthält.

1. Setze Π gleich der Menge aller Bahnrepräsentanten T_{i+1} .
2. Wähle ein Element $A_{i+1}g$ aus der Menge Π aus und setze U gleich dem Stabilisator von $A_i g$ in B . Dann ist U eine Untergruppe des Stabilisators von $A_{i+1}g$ in B .
3. Setze Σ gleich der Menge $\{A_i e g \mid e \in A_{i+1}\}$. Damit entspricht die Menge Σ der Menge aller Nebenklassen $\omega \in \Omega_i$ mit $\Upsilon_i(\omega) = \{A_{i+1}g\}$.
4. Wähle eine Nebenklasse s aus der Menge Σ und berechne das fusionierende Element $\eta_i(s)$, das mit b bezeichnet wird.
5. Das Element s^b ist in der Menge T_i enthalten und $\Upsilon_i(s^b) = \{A_{i+1}g^b\} \subseteq \{\omega \in \Omega_{i+1} \mid \exists t \in T_i : \omega \in \Upsilon_i(t)\}$. Setze daher $f_{i+1}(A_{i+1}g^b) = b^{-1}$.
6. Wenn $A_{i+1}g^b = A_{i+1}g$ ist, so ersetze U durch das Ergebnis des Algorithmus $ExtendGroup(b, U)$.
7. Entferne s aus der Menge Σ . Wenn die Menge Σ danach nicht leer ist, fahre fort mit Schritt 4.
8. Entferne $A_{i+1}g$ aus der Menge Π . Der Stabilisator von $A_{i+1}g$ in B ist gleich der Gruppe U . Wenn die Menge Π danach nicht leer ist, fahre fort mit Schritt 2.

7.3.9 Varianten des Algorithmus

Das folgende Kapitel enthält die Beschreibung eines Algorithmus, mit dem zu einer gegebenen Nebenklasse $A_k q$ überprüft werden kann, ob diese kanonisch ist. Im Gegensatz zu dem in diesem Kapitel beschriebenen Algorithmus wird zur Berechnung der Mengen \tilde{T}_1 bis \tilde{T}_k nicht der ursprüngliche Algorithmus von Schmalz verwendet. Stattdessen wird eine Strategie angewendet, die sich durch zwei wesentliche Eigenschaften auszeichnet.

Erstens basiert der im Folgenden vorgestellte Algorithmus auf der Annahme, dass in den Bahnen der Gruppe B auf den Mengen Ω_1 bis Ω_n die jeweils kleinste Nebenklasse jeder Bahn die kanonische ist. Aufgrund dieser Annahme muss nur ein kleiner Teil der kanonischen Repräsentanten aus den Mengen T_1 bis T_k untersucht werden, um festzustellen, ob eine Nebenklasse $A_k q$ kanonisch ist. Zu

Beginn des Algorithmus wird der kleinste Pfad zur Nebenklasse $A_k q$ berechnet. Die Berechnung dieses Pfades kann ohne großen Aufwand durchgeführt werden. Dann brauchen anschließend nur die kanonischen Repräsentanten aus den Mengen T_1 bis T_k berechnet werden, die kleiner oder gleich der entsprechenden Komponente dieses Pfades sind. Genauer gesagt, kann die Untersuchung sofort abgebrochen werden, sobald in einer der Mengen T_1 bis T_k ein kanonischer Repräsentant gefunden wurde, der kleiner als die entsprechende Komponente des genannten Pfades ist. Denn in diesem Fall steht das Ergebnis, dass die Nebenklasse $A_k q$ nicht kanonisch ist, sofort fest. Dies ermöglicht es, die Untersuchung wesentlich zu beschleunigen.

Zweitens kann, unter Verwendung einer starken Untergruppenleiter, die Menge der Repräsentanten aus den Mengen T_1 bis T_k zusammen mit deren Stabilisatoren und der fusionierenden Abbildung rekursiv berechnet werden. Das heißt die fusionierende Abbildung und die Mengen \tilde{T}_1 bis \tilde{T}_k und T_1 bis T_k brauchen nicht mehr im Speicher gehalten werden. Dabei muss natürlich in Kauf genommen werden, dass sich der Rechenaufwand entsprechend vergrößert, da ja, im Gegensatz dazu, beim Algorithmus nach Schmalz nur ein Speicherzugriff notwendig ist. Andererseits ermöglicht es dieser Ansatz, den Leiterspiel-Algorithmus auch für größere Probleme anwenden zu können, die ansonsten aufgrund des übermäßigen Speicherbedarfs mit dem Algorithmus von Schmalz nicht berechnet werden könnten.

Weitere Varianten des Leiterspiel-Algorithmus, bei denen die Eigenschaften der beschriebenen Algorithmen kombiniert und den Bedürfnissen anpasst werden, sind möglich. Ein Beispiel dafür sind die in Kapitel 7.6.3 beschriebenen Algorithmen, mit denen zu einer gegebenen Nebenklasse der Stabilisator dieser Nebenklasse und deren kanonischer Bahnrepräsentant berechnet werden können.

7.4 Depthfirst StroLL Algorithmus

Es sei G eine Gruppe, (A_1, \dots, A_n) eine starke Untergruppenleiter, bestehend aus Untergruppen von G . Für je zwei aufeinander folgende Gruppen der Untergruppenleiter sei der Gruppenindex klein und insbesondere endlich. Es sei eine Untergruppe B von G , ein Index $k \leq n$ und eine Nebenklasse $A_k q$ gegeben, für die überprüft werden soll, ob sie kanonisch in ihrer Bahn ist. Der im Folgenden beschriebene Algorithmus ist dazu geeignet, zu überprüfen, ob die Nebenklasse $A_k q$ kanonisch in der Bahn $A_k q^B = \{A_k q b \mid b \in B\}$ ist und berechnet im Falle, dass die Nebenklasse kanonisch ist, den zugehörigen Stabilisator.

Es bezeichne $J \subset \{1, \dots, n\}$ eine beliebige Indexmenge und Δ_J bezeichne, wie in Lemma 7 beschrieben, den entsprechenden Block zu dieser Indexmenge. Das Fundamentallemma 1 ermöglicht es, einen G -Isomorphismus von der Menge $\{\Delta_J^g \mid g \in G\}$ in die Menge der Nebenklassen des Stabilisators G_{Δ_J} anzugeben, so dass für alle $g \in G$ der Block Δ_J^g auf die Nebenklasse $G_{\Delta_J} g$ abgebildet wird. Aufgrund dieser Isomorphie wird im folgenden Text nicht immer genau zwischen der Betrachtung der Blöcke und der Betrachtung der entsprechenden Nebenklassen unterschieden. Immer dann, wenn ohne weitere Hinweise von Betrachtung der Blöcke zur Betrachtung der entsprechenden Nebenklassen übergegangen wird, wird dieser G -Isomorphismus zugrunde gelegt.

7.4.1 Überblick

Um mit dem Depthfirst StroLL Algorithmus überprüfen zu können, ob die Nebenklasse $A_k q$ kanonisch ist, wird eine weitere Leiter benötigt. Für alle $1 \leq i \leq k$ bezeichne E_i die Schnittmenge $A_i \cap A_k$, die auch wieder eine Gruppe ist. Dann ist (E_1, \dots, E_k) auch wieder eine Untergruppenleiter und für alle $1 \leq i \leq k$ existiert zwischen den Leitern ein G -Homomorphismus $\varphi : E_i \backslash G \rightarrow A_i \backslash G$, mit $\varphi(E_i g) = A_i g$ für alle $g \in G$.

Diese Homomorphismen zwischen den Nebenklassenmengen der beiden Leitern werden beim Depthfirst StroLL genutzt, um die benötigten Zwischenergebnisse in den Splitting- und Fusingschritten zu berechnen.

In Abbildung 7.4 sind die entsprechenden G -Homomorphismen zwischen

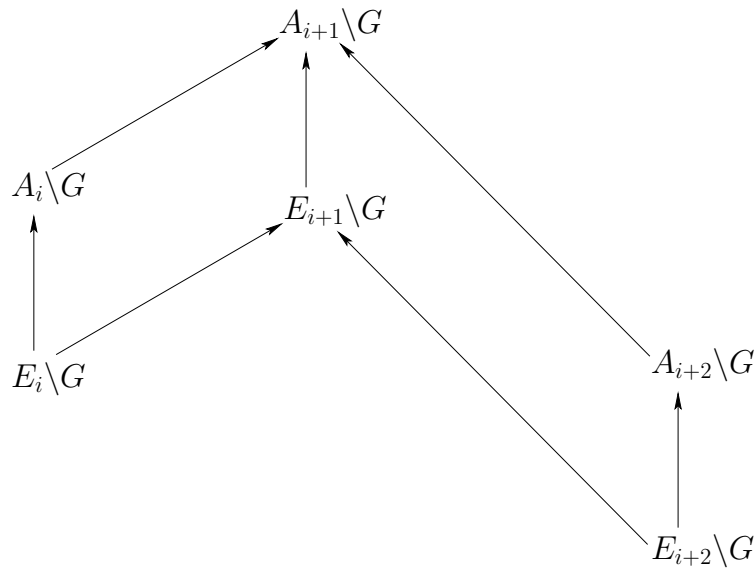


Abbildung 7.4: Homomorphismen zwischen den Leitern

den Nebenklassenmengen der beiden Leitern skizziert. Der Schritt von $A_i \setminus G$ nach $A_{i+1} \setminus G$ ist ein Fusingschritt und der Schritt von $A_{i+1} \setminus G$ nach $A_{i+2} \setminus G$ ist ein Splittingschritt.

Im Depthfirsts StroLL Algorithmus wird die Tatsache ausgenutzt, dass der Stabilisator und ein fusionierendes Elementes zu einer Nebenklasse $E_i g$ wesentlich effizienter berechnet werden können, wenn der entsprechende Stabilisator und ein fusionierendes Element zur Nebenklasse $A_i g$ bereits bekannt sind.

Wurde auf diese Weise der Stabilisator und ein fusionierendes Element zur Nebenklasse $E_k q$ bestimmt, so können diese unverändert für die Nebenklasse $A_k q$ übernommen werden. Denn es gilt $E_k = A_k \cap A_k = A_k$, daher sind die Nebenklassen $E_k q$ und $A_k q$ identisch.

Um auf diese Weise mit dem Depthfirst StroLL einen vollständigen Kanonizitätstest durchführen zu können oder den Stabilisator zu einer Nebenklasse $A_k q$ berechnen zu können, müssen eine Reihe von Voraussetzungen erfüllt sein:

- die Untergruppen der Leiter (E_1, \dots, E_k) müssen bekannt sein,
- der kleinste Pfad $\rho = (A_1 p, \dots, A_k p)$ im Block $\Delta_{\{k\}}^q$ muss bekannt sein,

- für alle $i < k$ muss der Stabilisator $B_{A_i p} = B \cap p^{-1} A_i p$, der im Folgenden auch mit C_i bezeichnet wird, bekannt sein.

Im Gegensatz zu den Stabilisatoren C_1, \dots, C_{k-1} kann der Stabilisator C_k der Nebenklasse $A_k q$ unter der Operation der Gruppe B natürlich nicht als bekannt vorausgesetzt werden, da dieser ja berechnet werden soll. Daher muss, falls der letzte Leiterschritt ein Fusingschritt ist, für diesen letzten Leiterschritt eine andere Strategie wie bei den übrigen Fusingschritten verwendet werden.

7.4.2 Vorbereitungen

Berechnung der Leiter (E_1, \dots, E_k)

Für alle $1 \leq i \leq k$ bezeichne E_i diejenige Gruppe, die genau aus den Elementen der Schnittmenge der beiden Gruppen $A_i \cap A_k$ besteht. Dann ist die Folge von Untergruppen (E_1, \dots, E_k) auch eine Untergruppenleiter. Da die Leiter (E_1, \dots, E_k) außer vom Index k unabhängig von anderen Eingabeparametern ist und nur von der verwendeten Leiter (A_1, \dots, A_n) abhängt, genügt es, diese für jedes $k \leq n$ nur ein einziges Mal zu berechnen und abzuspeichern.

Operiert die Gruppe A_k durch Rechtsmultiplikation auf der Menge $A_i \setminus G$, so ist der Stabilisator der Nebenklasse A_i gleich der Schnittmenge $A_i \cap A_k$. Dieser Stabilisator kann zum Beispiel durch einen Aufruf des Depthfirst StroLL Algorithmus berechnet werden. Die Gruppen E_1, \dots, E_{k-1} können unter Verwendung der entsprechenden Leitern $(A_1 \cap A_j, \dots, A_{j-1} \cap A_j, A_j)$ mit Indizes j kleiner als k berechnet werden, so dass keine Probleme mit zirkulären Voraussetzungen entstehen.

Die Gruppen E_1, \dots, E_k werden unter anderem dazu benötigt, um zu jedem Block $\Delta_{\{i,k\}}^g$ dessen Urbild unter den in Lemma 8 beschriebenen G -Homomorphismen $\varphi_{(\{i,i+1,k\},\{i,k\})}$ und $\varphi_{(\{i-1,i,k\},\{i,k\})}$ berechnen zu können.

Nach Lemma 7 ist $E_i = A_i \cap A_k$ gleich dem Stabilisator des Blockes $\Delta_{\{i,k\}}$ in der Gruppe G . Da die Gruppe G transitiv auf der Menge der Pfade und damit auch transitiv auf der Menge der Blöcke operiert, folgt mit Satz 3.1.3, dass die Menge $\{\Delta_{\{i,i+1,k\}}^e \mid e \in E_i\}$ gleich der Menge der Urbilder des Blockes $\Delta_{\{i,k\}}$ unter dem G -Homomorphismus $\varphi_{(\{i,i+1,k\},\{i,k\})}$ ist. Aufgrund der Tran-

sitivität und Satz 3.1.2 gilt genauso, dass die Menge $\{\Delta_{\{i,i+1,k\}}^{ge} \mid e \in g^{-1}E_i g\} = \{\Delta_{\{i,i+1,k\}}^{eg} \mid e \in E_i\}$ gleich der Menge der Urbilder des Blockes $\Delta_{\{i,k\}}^g$ unter dem G -Homomorphismus $\varphi_{(\{i,i+1,k\},\{i,k\})}$ ist. Für jedes $g \in G$ enthält die als Menge betrachtete Bahn $\Delta_{\{i,i+1,k\}}^{gg^{-1}E_i g}$ genau die Urbilder des Blockes $\Delta_{\{i,k\}}^g$ unter dem G -Homomorphismus $\varphi_{(\{i,i+1,k\},\{i,k\})}$.

Ermittle kleinsten Pfad im Block $\Delta_{\{k\}}^q$

Um zu überprüfen, ob die Nebenklasse $A_k q$ kanonisch ist, wird zuerst der kleinste Pfad ρ im Block $\Delta_{\{k\}}^q$ ermittelt und ein Gruppenelement p bestimmt, für das $\rho = (A_1 p, \dots, A_k p)$ gilt. Nach Lemma 5 ist $A_k q$ genau dann kanonisch, wenn in keinem der Blöcke der Bahn $\Delta_{\{k\}}^{qB}$ ein Pfad enthalten ist, der kleiner ist als ρ .

Mit Hilfe des Unterprogrammes *FindSmallestPath*, das in Kapitel 7.6.6 beschrieben wird, kann der Pfad ρ ohne großen Aufwand berechnet werden.

Stabilisatoren C_1, \dots, C_{k-1}

In den Splitting- und Fusingschritten des Depthfirst StroLL Algorithmus werden die Stabilisatoren C_1, \dots, C_{k-1} der Nebenklassen $A_1 p$ bis $A_{k-1} p$ unter der Operation der Gruppe B als bekannt vorausgesetzt. Unter der Voraussetzung, dass die Nebenklassen $A_1 p$ bis $A_{k-1} p$ kanonisch sind, können diese Stabilisatoren durch einen Aufruf des Unterprogramms *NextStep* berechnet werden. Dieses Unterprogramm wird im folgenden Text, beginnend auf Kapitel 7.4.5 genauer beschrieben.

Sollte hingegen eine der Nebenklassen $A_1 p$ bis $A_{k-1} p$ nicht kanonisch sein, so ist auch die Nebenklasse $A_k p = A_k q$ nicht kanonisch und der Algorithmus kann mit diesem Ergebnis sofort abgebrochen werden.

Denn angenommen eine Nebenklasse $A_i p$ ist nicht kanonisch, so existiert ein $b \in B$ für das gilt, dass $A_i p b$ kleiner ist als $A_i p$. Nach Lemma 6 enthält in diesem Fall der Block $\Delta_{\{i,k\}}^{pb}$ einen Pfad, der kleiner ist als der Pfad ρ . Daraus folgt, dass auch der Block $\Delta_{\{k\}}^{pb}$ einen Pfad enthält, der kleiner ist als der Pfad

ρ und dies wiederum widerspräche der Annahme, dass die Nebenklasse $A_i q$ kanonisch ist.

Splitting Orbits als letzter Schritt

Im Falle, dass die Nebenklasse $A_{k-1}p$ kanonisch und A_k eine Untergruppe von A_{k-1} ist, kann eine einfache Methode angewendet werden, um zu überprüfen, ob $A_k q$ kanonisch ist. Aus der Definition der Erweiterungsfunktion Υ_{k-1} geht hervor, dass $A_{k-1}p$ die einzige Nebenklasse ist, aus der die Nebenklasse $A_k q$ erzeugt werden kann. Aufgrund der Definition der Ordnung auf der Menge der Nebenklassen ist die Nebenklasse $A_k q$ kleiner als jede andere Nebenklasse der Bahn $A_k q^B$, die nicht aus der Nebenklasse $A_{k-1}p$ erzeugt wurde. Daher ist die Nebenklasse $A_k q$ genau dann kanonisch, wenn sie die kleinste Nebenklasse in der Bahn $A_k q^B$ ist, die aus der Nebenklasse $A_{k-1}p$ erzeugt wurde.

Nach dem Homomorphieprinzip 3.2.1 liegen alle Nebenklassen der Bahn $A_k q^B$, die gleichzeitig im Urbild der Nebenklasse $A_{k-1}p$ liegen, in derselben Bahn unter der Operation des Stabilisators der Nebenklasse $A_{k-1}p$. Dieser Stabilisator C_{k-1} kann mit dem Unterprogramm *NextStep*, das auf Seite 127 beschrieben wird, berechnet werden. Ob die Nebenklasse $A_k q$ kanonisch in ihrer Bahn $A_k q^{C_{k-1}}$ ist, kann wiederum mit dem Unterprogramm *FindOrbitRep*, das in Kapitel 7.6.5 beschrieben wird, überprüft werden.

Auch der Stabilisator der Nebenklasse $A_k q$ kann, falls $A_k \leq A_{k-1}$ ist, leicht berechnet werden. Nach dem Homomorphieprinzip 3.2.1 ist der Stabilisator von $A_k q$ eine Untergruppe des Stabilisators der Nebenklasse $A_{k-1}p$ und kann mit Hilfe des Unterprogrammes *ReduceStab* berechnet werden, das in Kapitel 7.6.2 beschrieben wird.

Fusing Orbits als letzter Schritt

Ist A_{k-1} Untergruppe von A_k , so ist der Konstruktionsschritt, in dem aus den Bahnen von B in der Menge Ω_{k-1} die Bahnen von B in der Menge Ω_k konstruiert werden, ein Fusingschritt. Dieser Fusingschritt unterscheidet sich von den vorangehenden Fusingschritten des Depthfirst StroLL, da bei diesem Kon-

struktionschritt nicht mehr vorausgesetzt werden kann, dass der Stabilisator der Nebenklasse $A_k p$ bereits bekannt ist. Daher muss der Fusingschritt von Ω_{k-1} nach Ω_k gesondert betrachtet werden. Auf den Seiten 121 und 122 in Kapitel 7.4.4 wird dieser Fusingschritt im Detail erläutert.

Die Überprüfung, ob die Nebenklasse $A_k q$ kanonisch in ihrer Bahn unter der Operation der Gruppe B ist, wird bereits vollständig in den anderen Splitting- und Fusingschritten, die in Kapitel 7.4.5 und 7.4.5 beschrieben werden, durchgeführt. Daher braucht in diesem letzten Fusingschritt lediglich der Stabilisator der Nebenklasse $A_k q$ berechnet werden.

Nach dem Homomorphieprinzip 3.2.1 ist der Stabilisator C_{k-1} der Nebenklasse $A_{k-1} p$ eine Untergruppe des Stabilisators der Nebenklasse $A_k q = A_k p$. Diese Untergruppe des Stabilisators wird, immer wenn ein neues, bisher unbekanntes Element des Stabilisators gefunden wurde, erweitert, bis der volle Stabilisator bekannt ist. Die Effizienz des Depthfirst StroLL Algorithmus kann zusätzlich erhöht werden, indem zu Beginn eine möglichst große, bereits bekannte Untergruppe H des Stabilisators von $A_k q$ übergeben wird. Je größer diese Untergruppe ist, um so weniger Blöcke müssen untersucht werden, um festzustellen, ob $A_k q$ kanonisch ist und um so effizienter arbeitet der Algorithmus.

7.4.3 Splitting Orbits

Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Es sei B eine Untergruppe von G und $A_k q$ eine Nebenklasse, die daraufhin untersucht werden soll, ob sie die kanonische Nebenklasse in ihrer Bahn ist und deren Stabilisator berechnet werden soll. Es bezeichne $\rho = (A_1 p, \dots, A_k p)$ den kleinsten Pfad des Blockes $\Delta_{\{k\}}^q$, das heißt ρ ist der kleinste Pfad, dessen letzte Komponente die Nebenklasse $A_k q = A_k p$ ist. Weiterhin sei ein $i < k$ mit $A_{i+1} \leq A_i$ und eine Gruppe $H \leq B \cap q^{-1} A_k q$ gegeben.

Benennungen

Einige Blöcken und G -Homomorphismen in diesem Kapitel sollen durch eine einfachere Schreibweise bezeichnet werden: Die Gruppe $A_i \cap A_{i+1} = A_{i+1}$ ist nach Lemma 7 sowohl der Stabilisator des Blockes $\Delta_{\{i,i+1\}}$ als auch der Stabilisator des Blockes $\Delta_{\{i+1\}}$. Daher ist der in Lemma 8 beschriebene G -Homomorphismus $\varphi_{(\{i,i+1\},\{i+1\})}$ nach Lemma 9 ein G -Isomorphismus. Nach Lemma 18 ist daher der Block $\Delta_{\{i,i+1\}}$ identisch mit dem Block $\Delta_{\{i+1\}}$. Der in Lemma 8 beschriebene G -Homomorphismus $\varphi_{(\{i,i+1\},\{i\})}$ wird daher im weiteren Text mit $\varphi_{(\{i+1\},\{i\})}$ bezeichnet.

Ähnliches gilt für die beiden Blöcke $\Delta_{\{i,i+1,k\}}$ und $\Delta_{\{i+1,k\}}$, die auch beide denselben Stabilisator $A_i \cap A_{i+1} \cap A_k = A_{i+1} \cap A_k$ haben. Die beiden Blöcke $\Delta_{\{i,i+1,k\}}$ und $\Delta_{\{i+1,k\}}$ sind daher identisch und der in Lemma 8 beschriebene G -Homomorphismus $\varphi_{(\{i,i+1,k\},\{i,k\})}$ wird im weiteren Text mit $\varphi_{(\{i+1,k\},\{i,k\})}$ bezeichnet.

Eingrenzung des Suchraumes

Die Menge der im Depthfirst StroLL Algorithmus zu untersuchenden Blockmengen kann unter Verwendung der Ordnungstreuenerzeugung drastisch reduziert werden. Voraussetzung dafür sind die Bedingungen, die in Kapitel 6.1 an die Ordnungen auf den zu untersuchenden Mengen von Nebenklassen und Blöcken gestellt wurden. Eine weitere Voraussetzung dafür ist die Bedingung, die in Kapitel 6.1 an die Kanonizitätsprädikate gestellt wurde, die wiederum eng mit den Ordnungen auf den Nebenklassenmengen verknüpft ist.

Angenommen die Nebenklasse $A_k q$ ist nicht die kanonische Nebenklasse in ihrer Bahn, dann existiert nach Lemma 5 ein Pfad $(A_1 r, \dots, A_k r) \prec \rho$, dessen letzte Komponente $A_k r$ in der Bahn $A_k q^B$ liegt. In diesem Fall müsste die $i + 1$ -te Komponente des Pfades $(A_1 r, \dots, A_k r)$ kleiner oder gleich der $i + 1$ -ten Komponente des Pfades ρ sein. Denn wäre $A_{i+1} p \prec A_{i+1} r$, so würde aus Lemma 6 folgen, dass der kleinste Pfad im Block $\Delta_{\{i+1,k\}}^p$ kleiner ist als der Pfad $(A_1 r, \dots, A_k r)$. Da aber ρ der kleinste Pfad im Block $\Delta_{\{k\}}^p$ ist und der Block $\Delta_{\{i+1,k\}}^p$ eine Teilmenge von $\Delta_{\{k\}}^p$ ist, die den Pfad ρ enthält, muss ρ auch

der kleinste Pfad im Block $\Delta_{\{i+1,k\}}^p$ sein und dies widerspricht der Annahme $(A_1r, \dots, A_kr) \prec \rho$.

Angenommen (A_1r, \dots, A_kr) ist kleiner als (A_1p, \dots, A_kp) dann existiert ein $1 < j \leq k$, so dass für alle $h < j$ gilt, dass $A_hr = A_hp$ ist und $A_jr \prec A_jp$ ist. Nach der Definition der Erweiterungsfunktion Υ_j und der Definition der Pfade ist A_j dann notwendigerweise eine Untergruppe von A_{j-1} und der Konstruktionsschritt, in dem der Block $\Delta_{\{j,k\}}^r$ aus dem Block $\Delta_{\{j-1,k\}}^r$ konstruiert wurde, ist ein Splittingschritt.

Da nach Voraussetzung $A_kr \in A_kp^B$ ist, existiert ein $b \in B$ mit $A_kr = A_kpb$. Der Block $\Delta_{\{j,k\}}^{rb^{-1}}$ liegt im Urbild von $\Delta_{\{k\}}^p$ unter dem in Lemma 8 beschriebenen G -Homomorphismus $\varphi_{(\{j,k\}, \{k\})}$. Gleichzeitig liegt der Block $\Delta_{\{j,k\}}^{rb^{-1}}$ in der Bahn des Blockes $\Delta_{\{j,k\}}^r$, der den Pfad (A_1r, \dots, A_kr) enthält. Der Block $\Delta_{\{j,k\}}^{rb^{-1}}$ wird daher in einem Splittingschritt aus dem Block $\Delta_{\{j-1,k\}}^{rb^{-1}}$ erzeugt.

Die Bahnen aller Blöcke, die die genannten Bedingungen erfüllen und in deren Bahnen sich möglicherweise ein Block befindet, der einen Pfad kleiner ρ enthält, werden in den Splittingschritten des Depthfirst StroLL Algorithmus untersucht. Wird dabei ein Pfad kleiner als der Pfad ρ gefunden, so kann der Kanonizitätstest mit dem Ergebnis, dass A_kq nicht kanonisch ist, abgebrochen werden.

Je nachdem, ob der Index j größer, kleiner oder gleich $i + 1$ ist, lassen sich drei Fälle unterscheiden:

Ist $j < i + 1$, so kann davon ausgegangen werden, dass der Block $\Delta_{\{j,k\}}^r$ in einem der vorangehenden Splittingschritte gefunden wird und dabei festgestellt wird, dass die Nebenklasse A_kq nicht kanonisch ist.

Ist $j > i + 1$, so folgt daraus sofort, dass $A_{i+1}r = A_{i+1}p$ ist. Da nach Voraussetzung $A_kr \in A_kp^B$ ist, existiert ein $b \in B$ mit $A_kr = A_kpb$. Der Block $\Delta_{\{i+1,k\}}^{rb^{-1}}$ liegt daher im Urbild des Blockes $\Delta_{\{k\}}^p$ unter dem Homomorphismus $\varphi_{(\{i+1,k\}, \{k\})}$ und gleichzeitig in der Bahn des Blockes $\Delta_{\{i+1,k\}}^r$, der den Pfad (A_1r, \dots, A_kr) enthält.

Angenommen es gilt $j = i + 1$, so ist $A_ir = A_ip$ und $A_{i+1}r \prec A_{i+1}p$. In diesem Fall soll der Block $\Delta_{\{i+1,k\}}^r$ in dem Splittingschritt, in dem das erste Mal ein

Block aus der Bahn $\Delta_{\{i,k\}}^{rB}$ in kleinere Blöcke zerlegt wird, gefunden werden. Da nach Voraussetzung $A_k r \in A_k p^B$ ist, existiert ein $b \in B$ mit $A_k r = A_k p b$. Der Block $\Delta_{\{i+1,k\}}^{rb^{-1}}$ liegt daher im Urbild des Blockes $\Delta_{\{k\}}^p$ unter dem Homomorphismus $\varphi_{(\{i+1,k\},\{k\})}$ und gleichzeitig in der Bahn des Blockes $\Delta_{\{i+1,k\}}^r$, der den Pfad $(A_1 r, \dots, A_k r)$ enthält.

Die Mengen Λ_i und Λ_{i+1}

Zusammenfassend ist daher jeder Pfad $(A_1 r, \dots, A_k r) \prec \rho$, zu dem ein $b \in B$ existiert mit $A_k r = A_k p b$, entweder in dem Block $\Delta_{\{i\}}^p$ enthalten oder er kann in einem der vorangehenden Splittingschritte gefunden werden. Wenn der Pfad $(A_1 r, \dots, A_k r)$ im Block $\Delta_{\{i\}}^p$ liegt, so gilt weiterhin, dass der Block $\Delta_{\{i,k\}}^r$, der den Pfad $(A_1 r, \dots, A_k r)$ enthält, in der Bahn des Blockes $\Delta_{\{i,k\}}^{rb^{-1}}$ liegt, der ein Urbild von $\Delta_{\{k\}}^p$ unter der Abbildung $\varphi_{(\{i,k\},\{k\})}$ ist. Daher brauchen in dem hier vorgestellten Splittingschritt nur diejenigen Blöcke betrachtet werden, die in der Menge $\Lambda_i = \{\Delta_{\{i,k\}}^g \mid A_k g = A_k p \wedge \exists b' \in B : A_i g b' = A_i p\}$ enthalten sind.

Die Urbilder aller dieser Blöcke der Menge Λ_i unter dem G -Homomorphismus $\varphi_{(\{i+1,k\},\{i,k\})}$ werden daraufhin untersucht, ob in deren Bahn ein Block enthalten ist, dessen kleinster Pfad kleiner ist als ρ . Ob in der Bahn ein entsprechender Block enthalten ist, kann nicht so leicht herausgefunden werden. Ist allerdings die $i+1$ -te Komponente aller Pfade eines Blockes kleiner als die Nebenklasse $A_{i+1} p$, so kann daraus geschlossen werden, dass die Nebenklasse $A_k q$ nicht kanonisch ist und der Kanonizitätstest wird abgebrochen.

Andernfalls muss der Pfad $(A_1 r, \dots, A_k r) \prec \rho$ sowohl im Block $\Delta_{\{i+1\}}^p$ als auch im Block $\Delta_{\{i+1,k\}}^r$ liegen. Daher ist der Block $\Delta_{\{i+1,k\}}^r$ in der Bahn eines Blockes aus der Menge $\Lambda_{i+1} = \{\Delta_{\{i+1,k\}}^g \mid A_k g = A_k p \wedge \exists b' \in B : A_{i+1} g b' = A_{i+1} p\}$ enthalten. Diese Menge Λ_{i+1} kann in einem Splittingschritt aus der Menge Λ_i konstruiert werden.

Abgeschlossenheit der Mengen Λ_i und Λ_{i+1}

Die Menge $\Lambda_j = \{\Delta_{\{j,k\}}^g \mid A_k g = A_k p \wedge \exists b' \in B : A_j g b' = A_j p\}$ ist für alle Indizes $1 \leq j < k$ abgeschlossen unter der Operation der Gruppe $H \leq B \cap p^{-1} A_k p$. Die Operation der Gruppe H auf der Menge Λ_j ist daher eine Gruppenopera-

tion. Dies kann wie folgt gezeigt werden:

Es sei $\Delta_{\{j,k\}}^g$ ein Block aus der Menge Λ_j und $b \in B$ ein Element für das $A_jgb = A_jp$ gilt. Weiterhin sei h ein beliebiges Element aus der Gruppe H . Dann liegt auch der Block $\Delta_{\{j,k\}}^{gh}$ in der Menge Λ_j , denn für das Element $b' = h^{-1}b \in B$ gilt $A_jghb' = A_jgb = A_jp$ und da H eine Untergruppe des Stabilisators der Nebenklasse A_kp ist, muss auch $A_kgh = A_kph = A_kp$ gelten.

Bahnen und Repräsentantensysteme in Λ_i und Λ_{i+1}

Angenommen es existiert ein Pfad $(A_1r, \dots, A_kr) \prec \rho$ mit $A_kr \in A_kp^B$, so wurde bereits gezeigt, dass dieser Pfad entweder in einem der vorangehenden Splittingschritten gefunden werden kann oder ein $b \in B$ existiert, so dass $\Delta_{\{i,k\}}^{rb} \in \Lambda_i$ und $(A_1r, \dots, A_kr) \in \Delta_{\{i,k\}}^r$ ist.

Jedes minimale Repräsentantensystem aller Bahnen der Gruppe H auf der Menge Λ_i enthält dann genau einen Repräsentanten aus der Bahn $\Delta_{\{i,k\}}^{rB}$. Um einen Pfad (A_1r, \dots, A_kr) zu finden, durch den sich die Nebenklasse A_kq als nicht kanonisch entlarven lässt, genügt es daher statt der Menge Λ_i ein minimales Repräsentantensystem aller Bahnen der Gruppe H auf der Menge Λ_i zu konstruieren. Im weiteren Text stehen die Bezeichnungen T_1, \dots, T_{k-1} entsprechend ihrer Indizes jeweils für ein minimales Repräsentantensystem der Bahnen von H auf einer der Mengen Λ_1 bis Λ_{k-1} .

Auf Seite 113 wird beschrieben, wie in den Splittingschritten aus einem minimalen Repräsentantensystem T_i ein minimales Repräsentantensystem T_{i+1} konstruiert wird. Die Repräsentanten aus der Menge T_{i+1} können so gewählt werden, dass jeder Repräsentant $t \in T_{i+1}$ im Urbild eines Repräsentanten $t' \in T_i$ unter dem Homomorphismus $\varphi_{(\{i+1,k\}, \{i,k\})}$ liegt. Dies kann wie folgt gezeigt werden:

Es sei $\Delta_{\{i+1,k\}}^g$ ein beliebiger Block aus der Menge Λ_{i+1} . Dann existiert ein $b \in B$, so dass $A_{i+1}gb = A_{i+1}p$ ist. Da A_{i+1} eine Untergruppe von A_i ist, muss auch $A_i gb = A_i p$ gelten und der Block $\Delta_{\{i,k\}}^g$ liegt daher in der Menge Λ_i . Es sei $h \in H$ ein Element, so dass $\Delta_{\{i,k\}}^{gh}$ im Repräsentantensystem T_i enthalten ist. Aufgrund der Abgeschlossenheit der Menge Λ_{i+1} unter der Operation der Gruppe H ist auch der Block $\Delta_{\{i+1,k\}}^{gh}$ in der Menge Λ_{i+1} enthalten. Gleichzeitig

liegt der Block $\Delta_{\{i+1,k\}}^{gh}$ in der Bahn des Blockes $\Delta_{\{i+1,k\}}^g$ unter der Operation der Gruppe H und der Block $\varphi_{(\{i+1,k\},\{i,k\})}(\Delta_{\{i+1,k\}}^{gh}) = \Delta_{\{i,k\}}^{gh}$ liegt in der Menge T_i .

In jeder Bahn der Gruppe H auf der Menge Λ_{i+1} existiert daher ein Block, der im Urbild eines Blockes aus der Menge T_i liegt. Wenn die Menge T_{i+1} so gewählt wird, dass diese nur aus Blöcken besteht, deren Bildblöcke unter der Abbildung $\varphi_{(\{i+1,k\},\{i,k\})}$ in der Menge T_i liegen, so kann die Menge T_{i+1} aus der Menge T_i konstruiert werden.

Ergebnisse des Splitting Orbits Schrittes

Es sei ein minimales Repräsentantensystem T_i aller Bahnen der Gruppe H auf der Menge Λ_i gegeben, das bedeutet, die Menge T_i enthält genau einen Block aus jeder Bahn. Zu jedem Block $\Delta_{\{i,k\}}^g \in T_i$ sei auch ein fusionierendes Element $b \in B$ gegeben, für das $A_i g b = A_i p$ gilt. Es sei weiterhin der Stabilisator $C_i = B \cap p^{-1} A_i p$ der Nebenklasse $A_i p$ in der Gruppe B gegeben und zu jedem Block $\Delta_{\{i,k\}}^g \in T_i$ sei auch der Stabilisator $D_i = H \cap g^{-1} A_i g$ dieses Blockes in der Gruppe H gegeben.

Aus diesen Angaben werden im Splitting Orbits Schritt folgende Dinge berechnet:

- Ein minimales Repräsentantensystem T_{i+1} aller Bahnen der Gruppe H auf der Menge Λ_{i+1} .
- Zu jedem Block $\Delta_{\{i,k\}}^g \in T_i$ wird überprüft, ob die Bahn $\Delta_{\{i,k\}}^{gB}$ einen Block enthält, dessen kleinster Pfad kleiner ist als ρ .
- Zu jedem Block $\Delta_{\{i+1,k\}}^g \in T_{i+1}$ wird der zugehörige Stabilisator dieses Blockes unter der Operation der Gruppe H berechnet.
- Zu jedem Block $\Delta_{\{i+1,k\}}^g \in T_{i+1}$ wird ein fusionierendes Element $b \in B$ berechnet, für das $A_{i+1} g b = A_{i+1} p$ ist.

Stabilisatoren der Elemente aus T_{i+1}

In den Splittingschritten des Depthfirst StroLL Algorithmus müssen zu allen Blöcken aus dem Repräsentantensystem T_{i+1} deren Stabilisatoren unter der

Operation der Gruppe H bestimmt werden. Auf Seite 110 wurde bereits festgestellt, dass das Repräsentantensystem T_{i+1} so gewählt werden kann, dass für jeden Block t aus der Menge T_{i+1} der Block $\varphi_{(\{i+1,k\},\{i,k\})}(t)$ in der Menge T_i enthalten ist. Ist der Block $\varphi_{(\{i+1,k\},\{i,k\})}(t)$ in der Menge T_i enthalten, so kann dessen zugehöriger Stabilisator $D_i = H \cap g^{-1}A_i g \cap g^{-1}A_k g$ als bekannt vorausgesetzt werden.

Nach dem Homomorphieprinzip 3.2.1 ist der Stabilisator jedes Blockes $t \in T_{i+1}$ eine Untergruppe des Stabilisators des Blockes $\varphi_{(\{i+1,k\},\{i,k\})}(t)$. Der Stabilisator D_i des Blockes $\varphi_{(\{i+1,k\},\{i,k\})}(t)$ operiert auf der Menge seiner Urbilder unter dem G -Homomorphismus $\varphi_{(\{i+1,k\},\{i,k\})}$. Ist der Gruppenindex $(A_i : A_{i+1})$ klein, so ist nach Lemma 15 auch der Gruppenindex $(E_i : E_{i+1})$ klein und daher ist auch die Menge der Urbilder des Blockes $\varphi_{(\{i+1,k\},\{i,k\})}(t) \in T_i$ unter dem G -Homomorphismus $\varphi_{(\{i+1,k\},\{i,k\})}$ klein. Daher ist die Bahn des Blockes $\Delta_{\{i+1,k\}}^g$ unter der Operation des Stabilisators D_i kurz und der Stabilisator des Blockes $\Delta_{\{i+1,k\}}^g$ kann mit Hilfe eines einfachen und effizienten Algorithmus berechnet werden. Ein Beispiel für so einen Algorithmus ist der in Kapitel 7.6.2 beschriebene erweiterte Bahnenalgorithmus.

Fusionierende Elemente

Um die nächsten Fusing- und Splittingschritte des Kanonizitätstests zu ermöglichen, muss zu jedem Block $\Delta_{\{i+1,k\}}^g$ aus der Menge T_{i+1} ein fusionierendes Element berechnet werden. Genauer gesagt soll zum Block $\Delta_{\{i+1\}}^g = \varphi_{(\{i+1,k\},\{i+1\})}(\Delta_{\{i+1,k\}}^g)$ ein $b' \in B$ berechnet werden, so dass $\Delta_{\{i+1\}}^{gb'} = \Delta_{\{i+1\}}^p$ gilt. Da jeder Block aus der Menge T_{i+1} auch in der Menge Λ_{i+1} liegen muss, ist sichergestellt, dass dieses Element b' immer existiert.

Es seien der Block $\Delta_{\{i+1,k\}}^g \in T_{i+1}$ und dessen Bildblock $\Delta_{\{i,k\}}^g$ unter dem G -Homomorphismus $\varphi_{(\{i+1,k\},\{i,k\})}$ gegeben und ein fusionierendes Element b , für das $\varphi_{(\{i,k\},\{i\})}(\Delta_{\{i,k\}}^g)^b = \Delta_{\{i\}}^p$ ist. Der Block $\Delta_{\{i+1,k\}}^{gb}$ liegt daher im Urbild des Blockes $\Delta_{\{i,k\}}^p$ unter dem G -Homomorphismus $\varphi_{(\{i+1,k\},\{i,k\})}$. Bezeichnet $C_i = B \cap p^{-1}A_i p$ den Stabilisator des Blockes $\Delta_{\{i\}}^p$ in B , dann existiert auch ein Element $c \in C_i$, so dass $\Delta_{\{i+1\}}^{gbc} = \Delta_{\{i+1\}}^p$ ist.

Dieses Element c kann unter Verwendung des *FindOrbitRep* Algorithmus, der in Kapitel 7.6.5 beschrieben wird, berechnet werden. Das Element $b' = bc \in B$ ist dann das gesuchte fusionierende Element zum Block $\Delta_{\{i+1,k\}}^g$, da $\varphi(\{i+1,k\},\{i+1\})(\Delta_{\{i+1,k\}}^g)^{b'} = \Delta_{\{i+1\}}^{gbc} = \Delta_{\{i+1\}}^p$ ist.

Auswahl der Repräsentanten

Die Menge T_{i+1} , die in diesem Splittingschritt konstruiert werden soll, ist ein minimales Repräsentantensystem aller Bahnen der Gruppe H auf der Menge Λ_{i+1} . Die Menge T_{i+1} soll so gewählt werden, dass jeder Block aus der Menge T_{i+1} im Urbild eines Repräsentanten der Menge T_i unter dem G -Homomorphismus $\varphi(\{i+1,k\},\{i,k\})$ liegt.

Die Menge T_{i+1} kann konstruiert werden, indem zu jedem Block $s \in T_i$ mit Stabilisator D_i folgende Schritte durchgeführt werden:

- berechne die Menge der Blöcke im Urbild von s unter der Abbildung $\varphi(\{i+1,k\},\{i,k\})$,
- prüfe, welche der Urbilder in der Menge Λ_{i+1} enthalten sind und entferne alle anderen Blöcke,
- bestimme in der verbleibenden Menge zu jeder Bahn unter der Operation des Stabilisators D_i genau einen Repräsentanten und füge diesen zur Menge T_{i+1} hinzu.

Die Grundlage für dieses Vorgehen ist das Homomorphieprinzip 3.2.1. Zeige zuerst, dass auf diese Weise zu jedem Block $\Delta_{\{i+1,k\}}^g \in \Lambda_{i+1}$ mindestens ein Repräsentant aus der Bahn dieses Blockes unter der Operation der Gruppe H in der Menge T_{i+1} enthalten ist:

Da der Block $\Delta_{\{i+1,k\}}^g$ in der Menge Λ_{i+1} enthalten ist, existiert ein $b \in B$, so dass $A_{i+1}gb = A_{i+1}p$ ist. Daher liegt auch der Block $\Delta_{\{i,k\}}^g$ in der Menge Λ_i , denn es gilt $A_i gb = A_i p$. Wenn $\Delta_{\{i,k\}}^g$ in der Menge Λ_i enthalten ist, so existiert ein $h \in H$, so dass $\Delta_{\{i,k\}}^{gh}$ der Repräsentant der Bahn $\Delta_{\{i,k\}}^{gB}$ in der Menge T_i ist. Da $\varphi(\{i+1,k\},\{i,k\})$ ein G -Homomorphismus ist, liegt der Block $\Delta_{\{i+1,k\}}^{gh}$ im

Urbild von $\Delta_{\{i,k\}}^{gh}$. Aus der Bahn dieses Blockes $\Delta_{\{i+1,k\}}^{gh}$ unter der Operation der Gruppe D_i wurde ein Repräsentant ausgewählt, der zu der Menge T_{i+1} hinzugefügt wurde. Da D_i eine Untergruppe der Gruppe H ist, liegt dieser Repräsentant auch in der Bahn des ursprünglichen Blockes $\Delta_{\{i+1,k\}}^g$.

Zeige jetzt noch, dass in der Menge T_{i+1} aus jeder Bahn der Gruppe H auf der Menge Λ_{i+1} nur ein einziger Repräsentant enthalten ist:

Angenommen es existiert ein $h \in H$ und zwei Blöcke $\Delta_{\{i+1,k\}}^g$ und $\Delta_{\{i+1,k\}}^{gh}$, die beide in der Menge T_{i+1} enthalten sind. Dann muss $\varphi_{(\{i+1,k\},\{i,k\})}(\Delta_{\{i+1,k\}}^{gh}) = \varphi_{(\{i+1,k\},\{i,k\})}(\Delta_{\{i+1,k\}}^g)$ sein. Denn nach Voraussetzung liegen die Bilder beider Blöcke in der Menge T_i und da $\Delta_{\{i,k\}}^{gh}$ und $\Delta_{\{i,k\}}^g$ in derselben Bahn liegen und T_i ein minimales Repräsentantensystem ist, müssen die Blöcke identisch sein. Wenn aber $\Delta_{\{i,k\}}^{gh} = \Delta_{\{i,k\}}^g$ ist, so muss das Element h im Stabilisator D_i dieses Blockes liegen. Dann liegen die beiden Blöcke $\Delta_{\{i+1,k\}}^{gh}$ und $\Delta_{\{i+1,k\}}^g$ im Urbild desselben Blockes $\Delta_{\{i,k\}}^g$ und in derselben Bahn unter der Operation der Gruppe D_i . Aus dieser Bahn wurde aber nur ein einziger Repräsentant zur Menge T_{i+1} hinzugefügt, daher sind die beiden Blöcke identisch.

Mit Hilfe des *FindOrbitRep* Algorithmus, der in Kapitel 7.6.5 beschrieben wird, kann zu jedem Block $\Delta_{\{i+1,k\}}^g$ ein Element $d \in D_i$ berechnet werden, so dass $\Delta_{\{i+1,k\}}^{gd}$ der entsprechende Repräsentant der Bahn $\Delta_{\{i+1,k\}}^{gD_i}$ ist.

Implementierung des Splittingschrittes

Angenommen es existiert ein Pfad (A_1r, \dots, A_kr) mit $A_kr \in A_kq^B$, so dass $(A_1r, \dots, A_kr) \prec (A_1p, \dots, A_kp)$. Es soll vorausgesetzt werden, dass $A_i r = A_i p$ ist, denn andernfalls wäre der Pfad bereits in einem der vorangehenden Splittingschritte gefunden worden. Dann existiert, wie in Kapitel 7.4.3 bereits erläutert wurde, ein Block $\Delta_{\{i,k\}}^{g'} \in T_i$ und ein $b \in B$, so dass $A_i g' b = A_i p$ und $A_k r = A_k g' b$ ist. Darüber hinaus enthält der Block $\Delta_{\{i,k\}}^{g'b}$ den Pfad (A_1r, \dots, A_kr) .

Wenn $A_{i+1}r = A_{i+1}p$ ist, so existiert im Urbild dieses Blockes $\Delta_{\{i,k\}}^{g'}$ unter der Abbildung $\varphi_{(\{i+1,k\},\{i,k\})}$ ein Block $\Delta_{\{i+1,k\}}^g \in T_{i+1}$ und ein $c \in C_i$, dem

Stabilisator des Blockes $\Delta_{\{i\}}^p$, so dass $A_{i+1}gbc = A_{i+1}p$ und $A_kgbc = A_kr$ ist. In diesem Fall enthält der Block $\Delta_{\{i+1,k\}}^{gbc}$ den Pfad (A_1r, \dots, A_kr) . In den folgenden Splitting- und Fusingschritten würde der Block $\Delta_{\{i+1,k\}}^g \in T_{i+1}$ weiter untersucht werden bis der Pfad (A_1r, \dots, A_kr) dann in einem späteren Splittingsschritt gefunden werden würde.

Andernfalls ist $A_{i+1}r \prec A_{i+1}p$ und es existiert im Urbild von $\Delta_{\{i,k\}}^{g'}$ unter der Abbildung $\varphi_{(\{i+1,k\},\{i,k\})}$ ein Block $\Delta_{\{i+1,k\}}^g$ und ein $c \in C_i$, dem Stabilisator des Blockes $\Delta_{\{i\}}^p$, so dass der Pfad (A_1r, \dots, A_kr) im Block $\Delta_{\{i+1,k\}}^{gbc}$ enthalten ist. Daher müssen alle Urbilder des Blockes $\Delta_{\{i,k\}}^{g'b}$ daraufhin untersucht werden, ob sie einen Pfad kleiner ρ enthalten.

Nach dem Homomorphieprinzip 3.2.1 enthalten die Urbilder von $\Delta_{\{i,k\}}^{g'}$ und $\Delta_{\{i,k\}}^{g'b}$ Blöcke aus denselben Bahnen. Daher kann der Pfad (A_1r, \dots, A_kr) gefunden werden, indem die Blöcke im Urbild des Blockes $\Delta_{\{i,k\}}^{g'}$ unter der Abbildung $\varphi_{(\{i+1,k\},\{i,k\})}$ berechnet werden und die Bahn jedes dieser Blöcke daraufhin untersucht wird, ob darin einen Pfad kleiner als ρ enthalten ist. Jeder Block im Urbild von $\Delta_{\{i,k\}}^{g'b}$ dessen Pfade an Position $i+1$ eine Nebenklasse kleiner als $A_{i+1}p$ tragen, enthält nach Lemma 6 einen Pfad kleiner als ρ .

Bei der Implementierung des Splittingschrittes wird zu jedem Block $\Delta_{\{i+1,k\}}^g$ im Urbild von $\Delta_{\{i,k\}}^{g'}$ die Nebenklasse $A_{i+1}gb$ bestimmt, die an Position $i+1$ aller Pfade des Blockes $\Delta_{\{i+1,k\}}^{gb}$ steht. Es bezeichne $c \in C_i$ ein Element für das gilt, dass $A_{i+1}gbc$ die kleinste Nebenklasse aus der Bahn der Nebenklasse $A_{i+1}gb$ unter der Operation von C_i ist. Dieses Element c kann mit dem in Kapitel 7.6.5 beschriebenen Algorithmus gefunden werden, unter der Voraussetzung, dass der vom diesem Algorithmus berechnete Repräsentant der kleinste Repräsentant seiner Bahn ist.

Aufgrund der Bedingung, die in Kapitel 6.1.2 an die Ordnung auf der Menge der Nebenklassen $A_{i+1} \backslash G$ gestellt wurde, muss gelten, dass der Block $\Delta_{\{i+1,k\}}^{gbc}$ unter allen Blöcken aus der Bahn $\Delta_{\{i+1,k\}}^{gB}$ die Pfade mit der kleinstmöglichen Nebenklasse an Position $i+1$ enthält. Nach Lemma 6 muss der Block $\Delta_{\{i+1,k\}}^{gbc}$ auch den kleinsten Pfad unter allen Pfaden aus Blöcken der Bahn $\Delta_{\{i+1,k\}}^{gB}$ enthalten.

Wenn $A_{i+1}gbc \prec A_{i+1}p$ ist, so enthält der Block $\Delta_{\{i+1,k\}}^{gbc}$ nach Lemma 6 einen Pfad kleiner als ρ und es folgt daraus, dass die Nebenklasse A_kp nicht kanonisch ist.

Wenn $A_{i+1}gbc \succ A_{i+1}p$ ist, enthält nach Lemma 6 keiner der Blöcke aus der Bahn $\Delta_{\{i+1,k\}}^{gB}$ einen Pfad kleiner als ρ .

Andernfalls, wenn die Nebenklasse $A_{i+1}gbc = A_{i+1}p$ ist, so kann der Pfad (A_1r, \dots, A_kr) in einem der folgenden Splittingschritte gefunden werden.

7.4.4 Fusing Orbits

Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Es sei B eine Untergruppe von G und A_kq eine Nebenklasse, die daraufhin untersucht werden soll, ob sie die kanonische Nebenklasse in ihrer Bahn ist. Es bezeichne $\rho = (A_1p, \dots, A_kp)$ den kleinsten Pfad im Block $\Delta_{\{k\}}^q$, das heißt ρ ist der kleinste Pfad, dessen letzte Komponente die Nebenklasse A_kq ist. Weiterhin sei ein $i < k$ mit $A_i \leq A_{i+1}$ und eine Gruppe $H \leq B \cap q^{-1}A_kq$ gegeben.

Benennungen

nach der Regel aus Kapitel 7.4.4 Die Gruppe $A_i \cap A_{i+1} = A_i$ ist nach Lemma 7 sowohl der Stabilisator des Blockes $\Delta_{\{i,i+1\}}$ als auch der Stabilisator des Blockes $\Delta_{\{i\}}$. Daher ist der in Lemma 8 beschriebene G -Homomorphismus $\varphi_{(\{i,i+1\},\{i\})}$ nach Lemma 9 ein G -Isomorphismus. Nach Lemma 18 ist daher der Block $\Delta_{\{i,i+1\}}$ identisch mit dem Block $\Delta_{\{i\}}$. Der in Lemma 8 beschriebene G -Homomorphismus $\varphi_{(\{i,i+1\},\{i+1\})}$ wird daher im weiteren Text mit $\varphi_{(\{i\},\{i+1\})}$ bezeichnet.

Ähnliches gilt für die beiden Blöcke $\Delta_{\{i,i+1,k\}}$ und $\Delta_{\{i,k\}}$, die auch beide denselben Stabilisator $A_i \cap A_{i+1} \cap A_k = A_i \cap A_k$ haben. Die beiden Blöcke $\Delta_{\{i,i+1,k\}}$ und $\Delta_{\{i,k\}}$ sind daher identisch und der in Lemma 8 beschriebene G -Homomorphismus $\varphi_{(\{i,i+1,k\},\{i+1,k\})}$ wird daher im weiteren Text mit $\varphi_{(\{i,k\},\{i+1,k\})}$ bezeichnet.

Eingrenzung des Suchraumes

Die Menge der im Depthfirst StroLL Algorithmus zu untersuchenden Blockmengen kann unter Verwendung der Ordnungstreuenerzeugung drastisch reduziert werden. Voraussetzung dafür sind die Bedingungen, die in Kapitel 6.1 an die Ordnungen auf den zu untersuchenden Mengen von Nebenklassen und Blöcken gestellt wurden. Eine weitere Voraussetzung dafür ist die Bedingung, die in Kapitel 6.1 an die Kanonizitätsprädikate gestellt wurde, die wiederum eng mit den Ordnungen auf den Nebenklassenmengen verknüpft ist.

Angenommen die Nebenklasse $A_k q$ ist nicht die kanonische Nebenklasse in ihrer Bahn, dann existiert nach Lemma 5 ein Pfad $(A_1 r, \dots, A_k r) \prec \rho$, dessen letzte Komponente $A_k r$ in der Bahn $A_k q^B$ liegt. In diesem Fall müsste die i -te Komponente des Pfades $(A_1 r, \dots, A_k r)$ kleiner oder gleich der i -ten Komponente des Pfades ρ sein. Denn wäre $A_i p \prec A_i r$, so würde aus Lemma 6 folgen, dass der kleinste Pfad im Block $\Delta_{\{i,k\}}^p$ kleiner ist als der Pfad $(A_1 r, \dots, A_k r)$. Da aber ρ der kleinste Pfad im Block $\Delta_{\{k\}}^p$ ist und der Block $\Delta_{\{i,k\}}^p$ eine Teilmenge von $\Delta_{\{k\}}^p$ ist, die den Pfad ρ enthält, muss ρ auch der kleinste Pfad im Block $\Delta_{\{i,k\}}^p$ sein und dies widerspricht der Annahme $(A_1 r, \dots, A_k r) \prec \rho$.

Angenommen $(A_1 r, \dots, A_k r)$ ist kleiner als $(A_1 p, \dots, A_k p)$ dann existiert ein $1 < j \leq k$, so dass für alle $h < j$ gilt, dass $A_h r = A_h p$ ist und $A_j r \prec A_j p$ ist. Nach der Definition der Erweiterungsfunktion Υ_j und der Definition der Pfade ist A_j dann notwendigerweise eine Untergruppe von A_{j-1} und der Konstruktionsschritt, in dem der Block $\Delta_{\{j,k\}}^r$ aus dem Block $\Delta_{\{j-1,k\}}^r$ konstruiert wurde, ist ein Splittingschritt.

Da nach Voraussetzung $A_k r \in A_k p^B$ ist, existiert ein $b \in B$ mit $A_k r = A_k p b$. Der Block $\Delta_{\{j,k\}}^{rb^{-1}}$ liegt im Urbild von $\Delta_{\{k\}}^p$ unter dem in Lemma 8 beschriebenen G -Homomorphismus $\varphi_{(\{j,k\}, \{k\})}$. Gleichzeitig liegt der Block $\Delta_{\{j,k\}}^{rb^{-1}}$ in der Bahn des Blockes $\Delta_{\{j,k\}}^r$, der den Pfad $(A_1 r, \dots, A_k r)$ enthält. Der Block $\Delta_{\{j,k\}}^{rb^{-1}}$ wird daher in einem Splittingschritt aus dem Block $\Delta_{\{j-1,k\}}^{rb^{-1}}$ erzeugt.

Die Bahnen aller Blöcke, die die genannten Bedingungen erfüllen und in deren Bahnen sich möglicherweise ein Block befindet, der einen Pfad kleiner ρ enthält, werden in den Splittingschritten des Depthfirst StroLL Algorithmus

untersucht. Wird dabei ein Pfad kleiner als der Pfad ρ gefunden, so kann der Kanonizitätstest mit dem Ergebnis, dass A_kq nicht kanonisch ist, abgebrochen werden.

Je nachdem, ob der Index j größer oder kleiner $i + 1$ ist, lassen sich zwei Fälle unterscheiden:

Ist $j < i + 1$, so kann davon ausgegangen werden, dass der Block $\Delta_{\{j,k\}}^r$ in einem der vorangehenden Splittingschritte gefunden wird und dabei festgestellt wird, dass die Nebenklasse A_kq nicht kanonisch ist.

Ist $j > i + 1$, so folgt daraus sofort, dass $A_i r = A_i p$ ist. Da nach Voraussetzung $A_k r \in A_k p^B$ ist, existiert ein $b \in B$ mit $A_k r = A_k p b$. Der Block $\Delta_{\{i,k\}}^{rb^{-1}}$ liegt daher im Urbild des Blockes $\Delta_{\{k\}}^p$ unter dem Homomorphismus $\varphi_{(\{i,k\},\{k\})}$ und gleichzeitig in der Bahn des Blockes $\Delta_{\{i,k\}}^r$, der den Pfad $(A_1 r, \dots, A_k r)$ enthält.

Die Mengen Λ_i und Λ_{i+1}

Zusammenfassend ist daher jeder Pfad $(A_1 r, \dots, A_k r) \prec \rho$, zu dem ein $b \in B$ existiert mit $A_k r = A_k p b$, entweder in dem Block $\Delta_{\{i\}}^p$ enthalten oder er kann in einem der vorangehenden Splittingschritte gefunden werden. Wenn der Pfad $(A_1 r, \dots, A_k r)$ im Block $\Delta_{\{i\}}^p$ liegt, so gilt weiterhin, dass der Block $\Delta_{\{i,k\}}^r$, der den Pfad $(A_1 r, \dots, A_k r)$ enthält, in der Bahn des Blockes $\Delta_{\{i,k\}}^{rb^{-1}}$ liegt, der ein Urbild von $\Delta_{\{k\}}^p$ unter der Abbildung $\varphi_{(\{i,k\},\{k\})}$ ist. Daher brauchen in dem hier vorgestellten Fusingschritt nur diejenigen Blöcke betrachtet werden, die in der Menge $\Lambda_i = \{\Delta_{\{i,k\}}^g \mid A_k g = A_k p \wedge \exists b' \in B : A_i g b' = A_i p\}$ enthalten sind. Aus den Blöcken dieser Menge Λ_i werden in einem Fusingschritt die Blöcke der Menge $\Lambda_{i+1} = \{\Delta_{\{i+1,k\}}^g \mid A_k g = A_k p \wedge \exists b' \in B : A_{i+1} g b' = A_{i+1} p\}$ konstruiert.

Abgeschlossenheit der Mengen Λ_i und Λ_{i+1}

Die Menge $\Lambda_j = \{\Delta_{\{j,k\}}^g \mid A_k g = A_k p \wedge \exists b' \in B : A_j g b' = A_j p\}$ ist für alle Indizes $1 \leq j < k$ abgeschlossen unter der Operation der Gruppe $H \leq B \cap p^{-1} A_k p$. Die Operation der Gruppe H auf der Menge Λ_j ist daher eine Gruppenoperation. Dies kann wie folgt gezeigt werden:

Es sei $\Delta_{\{j,k\}}^g$ ein Block aus der Menge Λ_j und $b \in B$ ein Element für das

$A_jgb = A_jp$ gilt. Weiterhin sei h ein beliebiges Element aus der Gruppe H . Dann liegt auch der Block $\Delta_{\{j,k\}}^{gh}$ in der Menge Λ_j , denn für das Element $b' = h^{-1}b \in B$ gilt $A_jghb' = A_jgb = A_jp$ und da H eine Untergruppe des Stabilisators der Nebenklasse A_kp ist, muss auch $A_kgh = A_kph = A_kp$ gelten.

Bahnen und Repräsentantensysteme in Λ_i und Λ_{i+1}

Angenommen es existiert ein Pfad $(A_1r, \dots, A_kr) \prec \rho$ mit $A_kr \in A_kp^B$, so wurde bereits gezeigt, dass dieser Pfad entweder in einem der vorangehenden Splittingschritten gefunden werden kann oder ein $b \in B$ existiert, so dass $\Delta_{\{i+1,k\}}^{rb} \in \Lambda_{i+1}$ und $(A_1r, \dots, A_kr) \in \Delta_{\{i+1,k\}}^r$ ist.

Jedes minimale Repräsentantensystem aller Bahnen der Gruppe H auf der Menge Λ_{i+1} enthält dann genau einen Repräsentanten aus der Bahn $\Delta_{\{i+1,k\}}^{rB}$. Um einen Pfad (A_1r, \dots, A_kr) zu finden, durch den sich die Nebenklasse A_kq als nicht kanonisch entlarven lässt, genügt es daher statt der Menge Λ_{i+1} ein minimales Repräsentantensystem aller Bahnen der Gruppe H auf der Menge Λ_{i+1} zu konstruieren. Im weiteren Text stehen die Bezeichnungen T_1, \dots, T_{k-1} entsprechend ihrer Indizes für minimale Repräsentantensysteme der Bahnen von H auf den Mengen Λ_1 bis Λ_{k-1} .

Aus einem gegebenen minimalen Repräsentantensystem T_i aller Bahnen der Gruppe H auf der Menge Λ_i soll ein Repräsentantensystem T_{i+1} aller Bahnen der Gruppe H auf der Menge Λ_{i+1} konstruiert werden. Die Konstruktion der Menge T_{i+1} soll erfolgen, indem zu jedem $t \in T_i$ der Block $\varphi_{(\{i,k\},\{i+1,k\})}(t)$ berechnet wird und ein Repräsentant aus der Bahn dieses Blockes zur Menge T_{i+1} hinzugefügt wird. Zeige, dass zu jedem Block $\Delta_{\{i+1,k\}}^g \in \Lambda_{i+1}$ ein Element $h \in H$ und ein Block $\Delta_{\{i,k\}}^{g'}$ im Urbild von $\Delta_{\{i+1,k\}}^g$ existieren, so dass $\Delta_{\{i,k\}}^{g'h} \in T_i$ ist:

Da $\Delta_{\{i+1,k\}}^g$ in der Menge Λ_{i+1} enthalten ist, existiert ein b aus der Gruppe B , so dass $A_{i+1}gb = A_{i+1}p$ ist. Dann ist $(A_1pb^{-1}, \dots, A_i pb^{-1}, A_{i+1}g, \dots, A_kg)$ ein Pfad aus dem Block $\Delta_{\{i+1,k\}}^g$. Da (A_1, \dots, A_k) eine starke Leiter ist, existiert nach Definition 16 ein $g' \in G$, so dass der Pfad (A_1g', \dots, A_kg') gleich $(A_1pb^{-1}, \dots, A_i pb^{-1}, A_{i+1}g, \dots, A_kg)$ ist. Der Block $\Delta_{\{i,k\}}^{g'}$ liegt im Urbild des Blockes $\Delta_{\{i+1,k\}}^g$ unter der Abbildung $\varphi_{(\{i,k\},\{i+1,k\})}$ und es gilt weiterhin, dass

$A_i g' \in A_i p^B$ und $A_k g' = A_k g$ ist. Daher ist $\Delta_{\{i,k\}}^{g'}$ auch in der Menge Λ_i enthalten und es existiert ein $h \in H$, so dass $\Delta_{\{i,k\}}^{g'h}$ in der Menge T_i enthalten ist.

Ergebnisse des Fusing Orbits Schrittes

Es sei ein minimales Repräsentantensystem T_i aller Bahnen von H in der Menge Λ_i gegeben, das bedeutet für jeden Block $\Delta_{\{i,k\}}^g \in \Lambda_i$ existiert genau ein $t \in T_i$, das in der Bahn $\Delta_{\{i,k\}}^{gH}$ enthalten ist. Wenn $i+1 < k$ ist, so sei der Stabilisator $C_{i+1} = B \cap p^{-1} A_{i+1} p$ der Nebenklasse $A_{i+1} p$ in der Gruppe B gegeben. Zu jedem Block $\Delta_{\{i,k\}}^g \in T_i$ sei weiterhin der Stabilisator $D_i = H \cap g^{-1} A_i g$ dieses Blockes in der Gruppe H und ein $b \in B$ mit $A_i g b = A_i p$ gegeben.

Aus diesen Angaben werden im Fusing Orbits Schritt folgende Dinge berechnet:

- Ein minimales Repräsentantensystem T_{i+1} aller Bahnen der Gruppe H auf der Menge Λ_{i+1} konstruiert.
- Zu jedem Block $\Delta_{\{i+1,k\}}^g \in T_{i+1}$ wird der zugehörige Stabilisator dieses Blockes unter der Operation der Gruppe H berechnet.
- Zu jedem Block $\Delta_{\{i+1,k\}}^g \in T_{i+1}$ wird ein fusionierendes Element $b \in B$ berechnet, für das $A_{i+1} g b = A_{i+1} p$ ist.
- Ist $i+1 = k$, so wird der volle Stabilisator $C_k = B \cap q^{-1} A_k q$ von $A_k q$ in der Gruppe B berechnet. Dieser Stabilisator ist identisch mit dem Stabilisator des Blockes $\Delta_{\{k\}}^g$ in B .

Berechnung des Stabilisators von $A_k q$

Wenn $A_{k-1} \leq A_k$ ist, erfolgt die Berechnung des Stabilisators der Nebenklasse $A_k q$ im letzten Fusingschritt. Es seien das Repräsentantensystem T_{k-1} , die Gruppe H und der Stabilisator C_{k-1} gegeben. Es sei weiterhin ein Algorithmus $ExtendGroup(b, U)$ gegeben, der zu jeder Untergruppe U von B und zu jedem Element $b \in B$ die kleinste Untergruppe von B bestimmt, die sowohl das Element b als auch die Gruppe U enthält. Der Algorithmus $ExtendGroup$ wird in Kapitel 7.6.4 beschrieben.

Nach dem Homomorphieprinzip 3.2.1 ist C_{k-1} eine Untergruppe von C_k , dem Stabilisator von $A_k q$ in B . Daher kann C_k zur Initialisierung gleich C_{k-1} gesetzt werden. Wenn die Gruppe C_{k-1} eine Untergruppe der Gruppe H ist, kann C_k auch gleich H gesetzt werden. Andernfalls muss die Gruppe C_k , die mit dem Stabilisator C_{k-1} initialisiert wurde, erweitert werden bis H eine Untergruppe von C_k ist.

Es soll zuerst die Vorgehensweise beschrieben werden, wie der Stabilisator berechnet werden kann, wenn $C_k \geq H$ ist. Jedem Repräsentanten $\Delta_{\{k-1\}}^g \in T_{k-1}$ ist nach Voraussetzung ein fusionierendes Element $b \in B$ zugeordnet, für das $A_{k-1}g^b = A_{k-1}p$ ist. Dann ist $\Delta_{\{k\}}^{pb} = \varphi_{(\{k-1\}, \{k\})}(\Delta_{\{k-1\}}^{gb}) = \Delta_{\{k\}}^p$, daher liegt b im Stabilisator von $\Delta_{\{k\}}^p$ und damit auch im Stabilisator von $A_k q$. Die Gruppe C_k soll unter Zuhilfenahme des Algorithmus $ExtendGroup$ erweitert werden, bis diese die fusionierenden Elemente aller Repräsentanten der Menge T_{k-1} enthält. Die so erweiterte Gruppe C_k ist dann gleich dem Stabilisator von $A_k q$ in B .

Denn angenommen, es existiert ein Element c im Stabilisator von $A_k q$ in B , das nicht in der so erweiterten Gruppe C_k enthalten ist. Dann ist $\Delta_{\{k-1\}}^{pc}$ in der Menge Λ_{k-1} enthalten und es existiert ein $h \in H$, so dass $\Delta_{\{k-1\}}^{pch}$ in T_{k-1} liegt. Diesem Repräsentanten $\Delta_{\{k-1\}}^{pch} \in T_{k-1}$ wurde ein fusionierendes Element $b \in B$ zugeordnet, für das $A_{k-1}pchb = A_{k-1}p$ gilt. Daher liegt chb in der Gruppe C_{k-1} und da $C_{k-1} \leq H \leq C_k$ gilt, liegt chb auch in C_k . Da auch die Elemente b und h in der Gruppe C_k liegen, muss auch $c = chb b^{-1}h^{-1}$ in der Gruppe C_k liegen und es folgt die Behauptung.

Wenn C_{k-1} nicht Untergruppe von H ist, dann wird C_k nicht mit H , sondern mit C_{k-1} initialisiert und anschließend mit dem Algorithmus *ExtendGroup* erweitert, bis $C_k \geq H$ gilt. Für jeden Block $\Delta_{\{k-1\}}^g$ aus der Menge Λ_{k-1} wird mit Hilfe des *FindOrbitRep* Algorithmus aus Kapitel 7.6.5 ein Element $h \in H$ bestimmt, so dass $\Delta_{\{k-1\}}^{gh}$ in T_{k-1} liegt. Wenn $\Delta_{\{k-1\}}^{gh}$ gleich $\Delta_{\{k-1\}}^p$ ist, so wird die Gruppe C_k um das Element h erweitert. Nachdem alle Blöcke aus der Menge Λ_{k-1} geprüft und C_k gegebenenfalls erweitert wurde, ist H eine Untergruppe von C_k . Anschließend kann C_k , wie zuvor beschrieben, erweitert werden, bis C_k den vollen Stabilisator von $A_k q$ in B fasst.

Berechnung des Repräsentantensystems T_{i+1}

Für alle $i < k$ mit $A_i \leq A_{i+1}$ ist die Berechnung der Repräsentanten und Stabilisatoren aller Blöcke aus der Menge T_{i+1} der entscheidende Schritt dieses Fusingschrittes. Auf Seite 119 wurde bereits gezeigt, dass die Menge aller Bildelemente von Blöcken aus der Menge T_i unter der Abbildung $\varphi(\{i,k\},\{i+1,k\})$ ein Repräsentantensystem aller Bahnen der Gruppe H auf der Menge Λ_{i+1} ist. Allerdings ist dieses Repräsentantensystem nicht minimal, das heißt es sind darin mitunter mehrere Repräsentanten aus derselben Bahn enthalten. Um die Effizienz des Algorithmus zu gewährleisten soll aus dieser Menge ein minimales Repräsentantensystem T_{i+1} konstruiert werden. Weiterhin muss zu jedem Repräsentanten $t \in T_{i+1}$ auch der Stabilisator dieses Blockes unter der Operation der Gruppe H berechnet werden.

Für die Auswahl der Blöcke, die im minimalen Repräsentantensystem T_{i+1} enthalten sein sollen, sind mehrere unterschiedliche Strategien denkbar. Da die Auswahl der Repräsentanten eng mit der Berechnung der zugehörigen Stabilisatoren verknüpft ist, wird festgelegt, dass die Menge T_{i+1} genau denjenigen Block jeder Bahn enthalten soll, der den kleinsten Pfad enthält.

Um zu einem gegebenen Block $\Delta_{\{i+1,k\}}^g \in \Lambda_{i+1}$ den entsprechenden kanonischen Repräsentanten dieses Blockes zusammen mit dessen zugehörigen Stabilisator zu berechnen, kann der in Kapitel 7.3 beschriebene Breadthfirst StroLL Algorithmus eingesetzt werden. Eine andere Möglichkeit besteht darin, eine Variante des Depthfirst StroLL oder Leiterspiel Light Algorithmus zur Be-

rechnung der Stabilisatoren einzusetzen. Dieses Vorgehen wird in Kapitel 7.6.3 beschrieben. Die Eigenschaften des Algorithmus, der zur Berechnung dieses Bahnrepräsentanten eingesetzt wird, wirken sich dabei prägend auf das Verhalten des Depthfirst StroLL Algorithmus aus.

Fusionierende Elemente

Es sei $\Delta_{\{i,k\}}^g$ ein Block aus der Menge T_i und $h \in H$ ein Element, so dass $\varphi(\{i,k\},\{i+1,k\})(\Delta_{\{i,k\}}^g)^h$ in T_{i+1} liegt. Es sei weiterhin ein Element $b \in B$ gegeben für das $A_i gb = A_i p$ ist. Zu diesem Block $\Delta_{\{i+1,k\}}^{gh} = \varphi(\{i,k\},\{i+1,k\})(\Delta_{\{i,k\}}^g)^h$ soll ein Element $b' \in B$ berechnet werden mit der Eigenschaft, dass $A_{i+1} gb' = A_{i+1} p$ ist.

Weil $A_i \leq A_{i+1}$ ist, gilt auch $A_{i+1} gb = A_{i+1} p$ und da H eine Untergruppe von B ist, liegt auch das Element $b' = h^{-1}b$ in der Gruppe B . Daher ist $A_{i+1} ghb' = A_{i+1} gb = A_{i+1} p$ und das Element b' besitzt die geforderte Eigenschaft.

Auswahlregel

In den Fusingschritten werden meist mehrere Bahnen von Blöcken aus der Menge T_i zu einer einzigen Bahn eines Blockes der Menge T_{i+1} vereinigt. In Kapitel 6.1.1 wurde die Erweiterungsfunktion Υ_i beschrieben, die jedem Element der Menge Ω_i eine einelementige Teilmenge der Menge Ω_{i+1} zuweist. Wird zu einem gegebenen Block $t \in T_{i+1}$ die Menge aller Blöcke der Menge T_i betrachtet, deren Bildmengen unter der Erweiterungsfunktion Υ_i aus einem Element aus der Bahn t^B besteht, so kann t aus jedem dieser Blöcke konstruiert werden. Um zu vermeiden, dass der Block t mehrfach konstruiert wird, soll genau ein Block aus der Menge T_i ausgewählt werden, aus dem der Block t dann tatsächlich konstruiert wird.

Beim Depthfirst StroLL Algorithmus sollen die Mengen T_1, \dots, T_k in Tiefensuche konstruiert werden. Um den Speicherbedarf gering zu halten soll dabei vermieden werden, dass die bereits konstruierten Repräsentanten der Menge T_i im Speicher gehalten werden müssen. Daher soll ein Auswahlkriterium ver-

wendet werden, das es ermöglicht, für ein einzelnes Element aus der Menge T_i festzustellen, ob aus diesem Element der entsprechende Repräsentant aus der Menge T_{i+1} konstruiert werden soll oder nicht.

Zu jedem Block $t \in T_{i+1}$ wird die Schnittmenge aus der Menge der Urbilder von t und der Menge Λ_i berechnet. Der kleinste Block aus dieser Schnittmenge ist der Block, der den kleinsten Pfad unter allen Blöcken dieser Schnittmenge enthält. Es bezeichne $t' \in T_i$ den Bahnrepräsentanten dieses kleinsten Blockes. Aus diesem Block $t' \in T_i$ soll der Block $t \in T_{i+1}$ konstruiert werden.

Berechnung der Schnittmenge

Im Folgenden soll beschrieben werden, wie festgestellt werden kann, ob aus einem Repräsentanten $\Delta_{\{i,k\}}^g \in T_i$ der entsprechende Bahnrepräsentant $\Delta_{\{i+1,k\}}^{gh} \in T_{i+1}$ des Bildblockes $\varphi(\{i,k\},\{i+1,k\})(\Delta_{\{i,k\}}^g)$ konstruiert werden soll. Nach der auf Seite 123 beschriebenen Auswahlregel ist der erste Schritt zur Bestimmung des ausgewählten Repräsentanten, die Menge der Urbilder des Blockes $\Delta_{\{i+1,k\}}^{gh}$ zu finden, die gleichzeitig in der Menge Λ_i liegen.

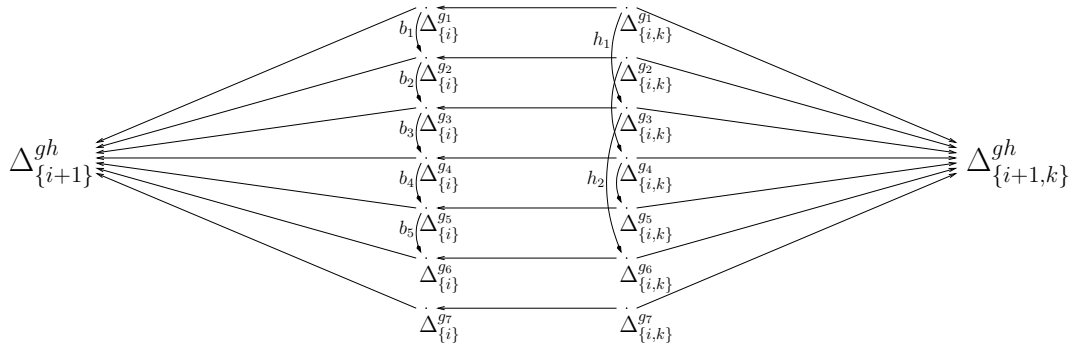


Abbildung 7.5: Bahnen von H (rechte Seite) in der Menge $\mathcal{S} \cap \Lambda_i$

Es bezeichne $\Delta_{\{i,k\}}^g$ einen Block aus dem Repräsentantensystem T_i und es sei ein $h \in H$ gegeben, so dass $\Delta_{\{i+1,k\}}^{gh} = \varphi(\{i,k\},\{i+1,k\})(\Delta_{\{i,k\}}^g)^h$ in der Menge T_{i+1} liegt. Es bezeichne \mathcal{S} die Menge der Urbilder des Blockes $\Delta_{\{i+1,k\}}^{gh}$ unter dem G -Homomorphismus $\varphi(\{i,k\},\{i+1,k\})$. Es sei ein $b \in B$ gegeben mit der Eigenschaft, dass $A_i g b = A_i p$ ist. Dann kann die Menge $\mathcal{S} \cap \Lambda_i$ wie folgt bestimmt

werden:

Lemma 19. *Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Es seien $i, k \leq n$ mit $1 \leq i < k$ und $A_i \leq A_{i+1}$ gegeben. Die Gruppe B , die Elemente g, b, h und die Mengen Λ_i und \mathcal{S} seien wie im voranstehenden Text gegeben. Dann ist die Menge $\mathcal{S} \cap \Lambda_i$ gleich der Menge $\{s \in \mathcal{S} \mid \varphi_{(\{i,k\}, \{i\})}(s) \in \Delta_{\{i\}}^{pB}\}$.*

Beweis. „ \subseteq “ Nach Lemma 16 ist die Menge $\{\Delta_{\{i,k\}}^{agh} \mid a \in A_k \cap A_{i+1}\}$ identisch mit der Menge \mathcal{S} . Es sei ein $a \in A_k \cap A_{i+1}$, so dass $\Delta_{\{i,k\}}^{agh}$ einen Block aus der Schnittmenge $\mathcal{S} \cap \Lambda_i$ bezeichnet. Dann existiert nach Definition der Menge Λ_i ein $b' \in B$ mit $A_i a g h = A_i p b'$.

Nach Lemma 7 ist der Stabilisator des Blockes $\Delta_{\{i\}}$ gleich A_i . Daher existiert nach Lemma 1 ein G -Isomorphismus der jeden Block $\Delta_{\{i\}}^{g'}$ auf die Nebenklasse $A_i g'$ abbildet. Aus $A_i a g h = A_i p b'$ folgt daher $\Delta_{\{i\}}^{agh} = \Delta_{\{i\}}^{pb'}$ und es gilt $\varphi_{(\{i,k\}, \{i\})}(\Delta_{\{i,k\}}^{agh}) = \Delta_{\{i\}}^{agh} = \Delta_{\{i\}}^{pb'}$.

„ \supseteq “ Es sei $\Delta_{\{i,k\}}^{agh}$ ein Block aus der Menge $\{s \in \mathcal{S} \mid \varphi_{(\{i,k\}, \{i\})}(s) \in \Delta_{\{i\}}^{pB}\}$. Da $\varphi_{(\{i,k\}, \{i\})}(\Delta_{\{i,k\}}^{agh}) \in \Delta_{\{i\}}^{pB}$ ist, existiert ein $b' \in B$, so dass $\varphi_{(\{i,k\}, \{i\})}(\Delta_{\{i,k\}}^{agh}) = \Delta_{\{i\}}^{p}$ ist. Nach Lemma 1 existiert ein G -Isomorphismus der jeden Block $\Delta_{\{i\}}^{g'}$ auf die Nebenklasse $A_i g'$ abbildet. Aus $\Delta_{\{i\}}^{agh b'} = \Delta_{\{i\}}^p$ folgt daher $A_i a g h b' = A_i p$ und daher liegt $\Delta_{\{i,k\}}^{agh} \in \mathcal{S}$ auch in der Menge Λ_i . □

Kleinstes Element der Schnittmenge

Es bezeichne $\Delta_{\{i,k\}}^g$ einen Block aus dem Repräsentantensystem T_i und es sei ein $h \in H$ gegeben, so dass $\Delta_{\{i+1,k\}}^{gh} = \varphi_{(\{i,k\}, \{i+1,k\})}(\Delta_{\{i,k\}}^g)^h$ in der Menge T_{i+1} liegt. Es bezeichne \mathcal{S} die Menge der Urbilder des Blockes $\Delta_{\{i+1,k\}}^{gh}$ unter dem G -Homomorphismus $\varphi_{(\{i,k\}, \{i+1,k\})}$ und \mathcal{T} die Menge der Urbilder des Blockes $\Delta_{\{i+1\}}^{gh}$ unter dem G -Homomorphismus $\varphi_{(\{i\}, \{i+1\})}$. Nach Lemma 17 existiert eine bijektive Abbildung $\zeta : \mathcal{S} \longrightarrow \mathcal{T}$, die mit dem auf die Menge \mathcal{S} eingeschränkten G -Homomorphismus $\varphi_{(\{i,k\}, \{i\})}$ übereinstimmt.

Der nächste Schritt zur Bestimmung des ausgewählten Repräsentanten ist, den kleinsten Block aus der Menge $\mathcal{S} \cap \Lambda_i$ zu bestimmen. Das folgende Lemma zeigt, dass die Abbildung ζ eine ordnungserhaltende Abbildung ist. Daher liegt der

kleinste Block aus der Menge $\mathcal{S} \cap \Lambda_i$ im Urbild des kleinsten Blockes aus der Bahn $\{\Delta_{\{i\}}^{ghc} \mid c \in h^{-1}bC_{i+1}b^{-1}h\}$ unter der Abbildung ζ . Dieser kleinste Block aus der Menge $\{\Delta_{\{i\}}^{ghc} \mid c \in h^{-1}bC_{i+1}b^{-1}h\}$ kann leicht bestimmt werden.

Lemma 20. *Die Menge \mathcal{S} und die Abbildung ζ seien wie im voranstehenden Text gegeben. Dann ist die Abbildung ζ ordnungserhaltend bezüglich der auf den Blockmengen definierten Totalordnung:*

$$\forall s_1, s_2 \in \mathcal{S} : s_1 \preceq s_2 \iff \zeta(s_1) \preceq \zeta(s_2)$$

Beweis. Nach Lemma 16 existieren zu allen Blöcken $s_1, s_2 \in \mathcal{S}$ Elemente $a_1, a_2 \in A_{i+1} \cap A_k$ mit $s_1 = \Delta_{\{i,k\}}^{a_1gh}$ und $s_2 = \Delta_{\{i,k\}}^{a_2gh}$. Zeige zuerst, dass aus $s_1 \preceq s_2$ folgt, dass auch $\zeta(s_1) \preceq \zeta(s_2)$ ist.

Ist $s_1 = s_2$, so ist dies trivial, andernfalls müssen sich die Pfade der beiden Blöcke s_1 und s_2 in der i -ten Komponente voneinander unterscheiden. Angenommen $A_i a_1 gh$ wäre größer als $A_i a_2 gh$, so würde mit Lemma 6 folgen, dass $\Delta_{\{i,k\}}^{a_2gh}$ einen Pfad enthält, der kleiner ist als alle Pfade des Blockes $\Delta_{\{i,k\}}^{a_1gh}$. Dies steht aber im Widerspruch zur Annahme, dass $\Delta_{\{i,k\}}^{a_1gh} \prec \Delta_{\{i,k\}}^{a_2gh}$ ist. Aus $A_i a_1 gh \prec A_i a_2 gh$ folgt mit Lemma 6 wiederum, dass $\Delta_{\{i\}}^{a_1gh} \prec \Delta_{\{i\}}^{a_2gh}$ ist.

Da die Ordnungen auf den Blockmengen $\Delta_{\{i,k\}}^G$ und $\Delta_{\{i\}}^G$ Totalordnungen sind und die Abbildung ζ injektiv ist, folgt die Behauptung. \square

Bestimmung des ausgewählten Repräsentanten

Der kleinste Block δ , der im Urbild des Blockes $\Delta_{\{i+1,k\}}^{gh} \in T_{i+1}$ unter dem Homomorphismus $\varphi_{(\{i,k\}, \{i+1,k\})}$ und gleichzeitig in der Menge Λ_i liegt, sei jetzt als bekannt vorausgesetzt. Um festzustellen, ob der ursprünglich betrachtete Block $\Delta_{\{i,k\}}^g \in T_i$ der nach dem Kriterium auf Seite 123 ausgewählte Repräsentant ist, muss überprüft werden, ob $\Delta_{\{i,k\}}^g \in T_i$ in derselben Bahn unter der Gruppe H liegt wie der Block δ .

Wenn sich dabei herausstellt, dass $\Delta_{\{i,k\}}^g$ der ausgewählte Repräsentant ist, dann soll der Block $\Delta_{\{i+1,k\}}^{gh}$ aus diesem Block $\Delta_{\{i,k\}}^g$ konstruiert werden und anschließend daraufhin untersucht werden, ob dieser einen Pfad enthält, der kleiner ist als der Pfad ρ . Der Block $\Delta_{\{i,k\}}^g \in T_i$ ist genau dann der ausgewählte Block, wenn $\Delta_{\{i,k\}}^{gh}$ und δ in derselben Bahn unter der Gruppe H liegen. Die beiden Blöcke $\Delta_{\{i,k\}}^{gh}$ und δ liegen beide im Urbild des Blockes $\Delta_{\{i+1,k\}}^{gh}$. Nach

dem Homomorphieprinzip 3.2.1 liegen $\Delta_{\{i,k\}}^{gh}$ und δ genau dann in derselben Bahn unter der Gruppe H , wenn die beiden Blöcke auch in derselben Bahn unter dem Stabilisator D_{i+1} des Blockes $\Delta_{\{i+1,k\}}^{gh}$ in der Gruppe H liegen. Die Berechnung dieses Stabilisators D_{i+1} wurde bereits auf Seite 122 beschrieben. Ist der Gruppenindex $(A_{i+1} : A_i)$ klein, so sind die Bahnen von $\Delta_{\{i,k\}}^{gh}$ und δ unter der Operation der Gruppe D_{i+1} kurz. In diesem Fall ist es eine leichte Aufgabe, einen kanonischen Repräsentanten zu beiden Bahnen zu berechnen. Wenn die beiden kanonischen Repräsentanten identisch sind, so liegen $\Delta_{\{i,k\}}^{gh}$ und δ in derselben Bahn. In Kapitel 7.6.2 wird ein geeigneter Algorithmus beschrieben, um bei kleinem Gruppenindex $(A_{i+1} : A_i)$ den kanonischen Repräsentanten zu finden.

7.4.5 Pseudocode

Unterprogramm NextStep

Die Methode *NextStep* ist eine Hilfsfunktion, die es ermöglicht, den Pseudocode übersichtlicher darzustellen. In dieser Methode werden entsprechende Unterprogramme aufgerufen, mit denen vor allem festgestellt werden kann, ob ein gewisser Block einen Pfad enthält, der kleiner ist als der Pfad ρ . Welcher Block dabei untersucht wird, hängt von den Eingabeparametern ab. Wenn ein entsprechender Pfad gefunden wurde, der kleiner ist als ρ , so wird *false* zurückgegeben, andernfalls ist der Rückgabewert *true*.

Die Eingabeparameter der Methode *NextStep* sind erstens zwei Indizes i und k , zweitens ein Element $g \in G$ und ein Element $b \in B$ und drittens eine Gruppe, die mit dem Buchstaben H bezeichnet wird. Die Eingabeparameter müssen so gewählt sein, dass für die Nebenklasse $A_{i-1}g$ und den Pfad $\rho = (A_1p, \dots, A_kp)$ gilt, dass $A_{i-1}gb = A_{i-1}p$ ist.

Die Gruppe H bezeichnet eine Untergruppe des Stabilisators des Blockes $\Delta_{\{k\}}^g$. Weiterhin werden die Gruppen C_1, \dots, C_{k-1} , die bereits bekannten Stabilisatoren der Nebenklassen $A_1p, \dots, A_{k-1}p$, und eine weitere Menge von Gruppen D_1, \dots, D_k übergeben. Die Gruppe D_{i-1} ist dabei identisch mit dem Stabilisator des Blockes $\Delta_{\{i-1,k\}}^g$ unter der Operation der Gruppe H . Diese Gruppen

werden als Referenz übergeben, so dass Änderungen an diesen Gruppen auch nach dem Aufruf der Methode *NextStep* erhalten bleiben.

- Wenn $i < k$ ist und $A_{i-1} \leq A_i$ ist, so wird in der Methode *NextStep* der Block $\Delta_{\{i,k\}}^{gb}$ daraufhin untersucht, ob dieser einen Pfad enthält, der kleiner ist als der Pfad ρ .
- Wenn $i < k$ ist und umgekehrt $A_i \leq A_{i-1}$ ist, so wird in der Methode *NextStep* der Block $\Delta_{\{i-1,k\}}^{gb}$ daraufhin untersucht, ob dieser einen Pfad enthält, der kleiner ist als der Pfad ρ .
- Wenn $i = k$ ist und $A_{k-1} \leq A_k$ ist, so kann die Gruppe C_k , die eine Untergruppe des Stabilisators der Nebenklasse $A_k g$ unter der Operation der Gruppe B enthält, erweitert werden. Dieser Schritt wurde bereits auf Seite 105 beschrieben.
- Wenn $i = k$ ist und $A_{k-1} \geq A_k$ ist, so kann der Stabilisator von $A_k g$ unter der Operation der Gruppe B berechnet werden. Dieser Schritt wurde bereits auf Seite 105 beschrieben.

Pseudocode NextStep

Outside Declared: Group G
 (A_1, \dots, A_n) $\backslash \backslash$ strong Ladder from G to A_n
 (A_1p, \dots, A_kp) $\backslash \backslash$ smallest Path in $\Delta_{\{k\}}^p$
 Group B
 Group H $\backslash \backslash$ $H \leq B_{A_kp}$
Input: Index i $\backslash \backslash$ $i \leq k$
 Index k $\backslash \backslash$ $k \leq n$
 $g \in G$ $\backslash \backslash$ examine Path Block $\Delta_{\{i-1,k\}}^g$
 $b \in B$ $\backslash \backslash$ $A_{i-1}gb = A_{i-1}p$
Referenced : $C = (C_1, \dots, C_n)$ $\backslash \backslash$ $\forall j < k : C_j = B_{A_jp}$
 $D = (D_1, \dots, D_k)$ $\backslash \backslash$ $D_{i-1} = H_{A_{i-1}g}$
Output: Bool

```

1   Bool  $r \leftarrow true$ 
2   if (  $2 == i$  )
3        $C_1 \leftarrow B$ 
4        $D_1 \leftarrow H$ 
5        $C_k \leftarrow H$ 
6   endif
7   if (  $i < k$  )
8       if (  $A_{i-1} \leq A_i$  )
9            $r \leftarrow FuseOrbits(i, k, g, b, C, D)$ 
10      else
11           $r \leftarrow SplitOrbits(i, k, g, b, C, D)$ 
12      endif
13  else
14      if (  $A_{i-1} \leq A_i$  )
15           $C_k \leftarrow ExtendGroup(b, C_k)$ 
16      else
17           $r \leftarrow SplitOrbits(i, k, g, b, C, D)$ 
18      endif
19  endif
20  return  $r$ 

```

Unterprogramm *CheckCanonical*

Um die Implementierung des Algorithmus zu erleichtern, werden die wichtigsten Unterprogramme des Algorithmus auch in Pseudocode dargestellt. Auf Seite 131 steht der Pseudocode des Unterprogramms *CheckCanonical*.

In diesem Unterprogramm wird die Berechnung der Nebenklassenstabilisatoren C_1, \dots, C_{k-1} durchgeführt und im Anschluss daran das Unterprogramm *NextStep* aufgerufen, das den eigentlichen Depthfirst StroLL Algorithmus startet.

Der Rückgabewert des Unterprogramms *CheckCanonical* ist *true* oder *false*, je nachdem, ob die Nebenklasse $A_k q$ kanonisch ist oder nicht. Im Falle, dass $A_k q$ kanonisch ist, wird der Stabilisator der Nebenklasse $A_k q$ in die referenzierte Variable C_k geschrieben. Das Referenzieren einer Variable ermöglicht es, dass der Wert dieser Variable von dem entsprechenden Unterprogramm verändert und auf diese Weise als Rückgabewert verwendet werden kann.

Pseudocode CheckCanonical

Outside Declared: Group G
 (A_1, \dots, A_n) $\backslash \backslash$ strong Ladder from G to A_n
Input: Index k $\backslash \backslash$ $k \leq n$
 $g \in G$
 Group B
Referenced : Group H $\backslash \backslash$ $H = 1_G$
 $C = (C_1, \dots, C_k)$ $\backslash \backslash$ $\forall j \leq k : C_j = 1_G$
Output: Bool

```

1   Bool  $r \leftarrow true$ 
2   Grouparray  $D = (D_1, \dots, D_k)$                                  $\backslash \backslash$   $\forall j \leq k : D_j \leftarrow 1_G$ 
3    $C_1 \leftarrow B$ 
4   GroupElement  $p, c$ 
5    $(A_1p, \dots, A_kp) \leftarrow FindSmallestPath(k, g)$              $\backslash \backslash$  smallest Path in  $\Delta_{\{k\}}^g$ 
6   for (  $i$  from 2 to  $k$  )
7       if (  $A_{i-1} \geq A_i$  )
8            $c \leftarrow FindOrbitRep(A_ip, C_{i-1})$                          $\backslash \backslash$   $c \leftarrow \underset{x \in C_{i-1}}{\operatorname{argmin}} A_ipx$ 
9           if (  $A_ipc \prec A_ip$  )
10              return false
11          endif
12           $C_i \leftarrow ReduceStab(A_ip, C_{i-1})$                          $\backslash \backslash$   $C_i \leftarrow B_{A_ip}$ 
13      else
14           $H \leftarrow C_{i-1}$ 
15           $r \leftarrow NextStep(2, i, p, 1_G, C, D)$                          $\backslash \backslash$   $C_i \leftarrow B_{A_ip}$ 
16          if ( false ==  $r$  )
17              return false
18          endif
19      endif
20  end
21  return true
  
```

Pseudocode SplitOrbits

Outside Declared: Group G
 $A = (A_1, \dots, A_n)$ $\backslash\backslash$ strong Ladder from G to A_n
 $E = (E_1, \dots, E_k)$ $\backslash\backslash \forall j \leq k : E_j = A_j \cap A_k$
 (A_1p, \dots, A_kp) $\backslash\backslash$ smallest Path in $\Delta_{\{k\}}^p$
 Group B
 Group H $\backslash\backslash H \leq B_{A_kp}$
Input: Index i $\backslash\backslash i \leq k$
 Index k $\backslash\backslash k \leq n$
 $g \in G$ $\backslash\backslash$ examine Path Block $\Delta_{\{i-1,k\}}^g$
 $b \in B$ $\backslash\backslash A_{i-1}gb = A_{i-1}p$
Referenced : $C = (C_1, \dots, C_k)$ $\backslash\backslash \forall j < k : C_j = B_{A_jp}$
 $D = (D_1, \dots, D_k)$ $\backslash\backslash D_{i-1} = H_{A_{i-1}g}$
Output: Bool

```

1   Bool  $r \leftarrow true$ 
2   GroupElement  $c, d, h$ 
3   foreach (  $E_i h' \in \{E_i e g \mid e \in E_{i-1}\}$  )
4        $h \leftarrow h'$                                      $\backslash\backslash$  choose any  $h \in E_i h'$ 
5        $d \leftarrow FindOrbitRep(A_i h, D_{i-1})$                      $\backslash\backslash d \leftarrow \operatorname{argmin}_{x \in D_{i-1}} A_i h x$ 
6       if (  $A_i h d == A_i h$  )
7            $c \leftarrow FindOrbitRep(A_i h b, C_{i-1})$                      $\backslash\backslash c \leftarrow \operatorname{argmin}_{x \in C_{i-1}} A_i h b x$ 
8           if (  $A_i h b c \prec A_i p$  )
9               return false                                     $\backslash\backslash A_k p b c \prec A_k p$ 
10          else
11              if (  $A_i h b c == A_i p$  )
12                   $D_i \leftarrow ReduceStab(A_i h, D_{i-1})$                      $\backslash\backslash D_i \leftarrow H_{A_i h}$ 
13                   $r \leftarrow NextStep(i+1, k, h, bc, C, D)$ 
14                  if (  $false == r$  )
15                      return false
16                  endif
17              endif
18          endif
19      endif
20  end
21  return true

```

<i>Outside Declared:</i>	Group G	
	(A_1, \dots, A_n)	$\backslash \backslash$ strong Ladder from G to A_n
	(A_1p, \dots, A_kp)	$\backslash \backslash$ smallest Path in $\Delta_{\{k\}}^p$
	Group B	
	Group H	$\backslash \backslash$ $H \leq B_{A_kp}$
<i>Input:</i>	Index i	$\backslash \backslash$ $i \leq k$
	Index k	$\backslash \backslash$ $k \leq n$
	$g \in G$	$\backslash \backslash$ examine Path Block $\Delta_{\{i-1,k\}}^g$
	$b \in B$	$\backslash \backslash$ $A_{i-1}gb = A_{i-1}p$
<i>Referenced :</i>	$C = (C_1, \dots, C_k)$	$\backslash \backslash$ $\forall j < k : C_j = B_{A_jp}$
	$D = (D_1, \dots, D_k)$	$\backslash \backslash$ $D_{i-1} = H_{A_{i-1}g}$
<i>Output:</i>	Bool	

7.5 Leiterspiel Light Algorithmus

Nicht immer zahlt sich der Aufwand zur Durchführung der in Kapitel 7.3 und 7.4 beschriebenen Algorithmus aus. Der in diesem Kapitel beschriebene Leiterspiel Light Algorithmus ist für kleinere Isomorphieprobleme gedacht, bei denen sich der Bahnrepräsentant leicht berechnen lässt und bei denen sich die aufwändigen Berechnungen, die beim Einsatz des Breadthfirst StroLL und des Depthfirst StroLL Algorithmus anfallen, nicht auszahlen. Ziel dieses Algorithmus ist es, das sprichwörtliche „mit Kanonen auf Spatzen schießen“ zu vermeiden.

Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Für je zwei aufeinander folgende Gruppen der Untergruppenleiter sei der Gruppenindex klein und insbesondere endlich. Für alle $i \leq k$ wird die Gruppe $A_i \cap A_k$ mit E_i bezeichnet. Es sei B eine Untergruppe von G , $k \leq n$ und $A_k q$ eine Nebenklasse, zu der überprüft werden soll, ob diese der kanonische Repräsentant in der Bahn $A_k q^B = \{A_k q b \mid b \in B\}$ ist. Wenn $A_k q$ kanonischer Bahnrepräsentant ist, soll weiterhin der Stabilisator C_k von $A_k q$ in B berechnet werden.

Es bezeichne $J \subset \{1, \dots, n\}$ eine beliebige Indexmenge und Δ_J bezeichne, wie in Lemma 7 beschrieben, den entsprechenden Block zu dieser Indexmenge. Das Fundamentallemma 1 ermöglicht es, einen G -Isomorphismus von der Menge $\{\Delta_J^g \mid g \in G\}$ in die Menge der Nebenklassen des Stabilisators G_{Δ_J} anzugeben, so dass für alle $g \in G$ der Block Δ_J^g auf die Nebenklasse $G_{\Delta_J} g$ abgebildet wird. Aufgrund dieser Isomorphie wird im folgenden Text nicht immer genau zwischen der Betrachtung der Blöcke und der Betrachtung der entsprechenden Nebenklassen unterschieden. Immer dann, wenn ohne weitere Hinweise von Betrachtung der Blöcke zur Betrachtung der entsprechenden Nebenklassen übergegangen wird, wird dieser G -Isomorphismus zugrunde gelegt.

7.5.1 Überblick

Sind die Bahnen dieses Stabilisators C_k auf den Nebenklassenmengen $E_1 \setminus A_k$ bis $E_{k-1} \setminus A_k$ kurz, so lohnt sich der Aufwand zur Berechnung der Bahnre-

präsentanten nicht immer. Indem im Gegensatz zum Depthfirst StroLL Algorithmus aus Kapitel 7.4 alle Bahnelemente der relevanten Bahnen von C_k auf den Mengen $E_1 \setminus A_k$ bis $E_{k-1} \setminus A_k$ durchlaufen werden, kann der zusätzliche Aufwand zur Berechnung der entsprechenden Bahnrepräsentanten vermieden werden.

Für alle $1 \leq i \leq k$ bezeichne $\psi_i : E_i \setminus A_k \rightarrow A_i \setminus G$ einen Homomorphismus von Gruppenoperationen, der jede Nebenklasse $E_i a \in E_i \setminus A_k$ auf die Nebenklasse $\psi_i(E_i a) = A_i a q$ abbildet. Für alle $1 \leq i \leq k$ bezeichne Θ_i die Menge $\{\psi_i(E_i a) \mid E_i a \in E_i \setminus A_k\}$. Für alle $1 \leq i \leq k$ bezeichne $\eta_i : \Theta_i \rightarrow B$ eine fusionierende Abbildung. Diese fusionierende Abbildung hat die Eigenschaft, dass für alle $A_i g \in \Theta_i$ gilt, dass $A_i g^{\eta_i(A_i g)}$ der kanonische Repräsentant aus der Bahn $A_i g^B$ ist. Die Auswahl der kanonischen Repräsentanten erfolgt auf Grundlage des in Kapitel 6.1.3 definierten Kanonizitätsprädikats und der in Kapitel 6.1.2 vorausgesetzten Ordnung auf der Menge der Nebenklassen.

Um mit dem Leiterspiel Light Algorithmus einen vollständigen Kanonizitätstest durchführen zu können oder den Stabilisator zu einer Nebenklasse $A_k q$ berechnen zu können, müssen eine Reihe von Voraussetzungen erfüllt sein:

- die Untergruppen der Leiter (E_1, \dots, E_k) müssen bekannt sein,
- der kleinste Pfad $\rho = (A_1 p, \dots, A_k p)$ im Block $\Delta_{\{k\}}^q$ muss bekannt sein,
- für alle $i < k$ muss der Stabilisator $B_{A_i p} = B \cap p^{-1} A_i p$, der im Folgenden auch mit C_i bezeichnet wird, bekannt sein.

Der Algorithmus kann in die folgenden Bestandteile zerlegt werden, wobei der Ablauf des Algorithmus nicht an die angegebene Reihenfolge gebunden ist.

1. Berechne für alle $1 \leq i \leq k$ eine Auswahl von Nebenklassen aus der Menge $E_i \setminus A_k$. Um zu prüfen, ob $A_k q$ der kanonische Repräsentant der Bahn $A_k q^B$ ist, müssen nur diejenigen Nebenklassen $E_i a \in E_i \setminus A_k$ und $\psi_i(E_i a) \in \Theta_i$ untersucht werden, für die folgende Bedingung erfüllt ist: Der kanonische Bahnrepräsentant der Bahn von $\psi_i(E_i a)$ ist minimal in der Menge aller Bahnrepräsentanten von Nebenklassen aus der Menge Θ_i .

2. Berechne für alle $1 \leq i \leq k$ und zu jeder Nebenklasse $E_i a \in E_i \setminus A_k$ aus der in Punkt 1 genannten Auswahl von Nebenklassen die Nebenklasse $\psi_i(E_i a) = A_i a q$.
3. Berechne zu allen $1 \leq i \leq k$ und zu jeder Nebenklasse $A_i g$ aus der Menge der in Punkt 2 berechneten Nebenklassen ein $b \in B$, so dass $A_i g b$ der kanonische Repräsentant der Bahn $A_i g^B$ ist.
4. Prüfe, zu jedem dieser in Punkt 3 berechneten kanonischen Repräsentanten $A_i g b$, ob $A_i g b$ kleiner als $A_i p$ ist. Wenn $A_i g b \prec A_i p$ ist, so ist die Nebenklasse $A_i p$ nicht kanonisch.
5. Wenn keiner der in Punkt 4 untersuchten kanonischen Repräsentanten $A_i g b$ kleiner ist als die entsprechende Nebenklasse $A_i p$, so folgt daraus, dass die Nebenklasse $A_k q$ kanonisch in ihrer Bahn ist. In diesem Fall kann der Stabilisator C_k der Nebenklasse $A_k q$ berechnet werden. Dieser Stabilisator kann berechnet werden, indem die Gruppe C_{k-1} unter Verwendung von einigen, der in Punkt 3 berechneten Gruppenelementen erweitert wird.

7.5.2 Vorbereitung

Berechnung der Leiter (E_1, \dots, E_k)

Für alle $1 \leq i \leq k$ bezeichne E_i diejenige Gruppe, die genau aus den Elementen der Schnittmenge der beiden Gruppen $A_i \cap A_k$ besteht. Dann ist die Folge von Untergruppen (E_1, \dots, E_k) auch eine Untergruppenleiter. Da die Leiter (E_1, \dots, E_k) außer vom Index k unabhängig von anderen Eingabeparametern ist und nur von der verwendeten Leiter (A_1, \dots, A_n) abhängt, genügt es, diese für jedes $k \leq n$ nur ein einziges Mal zu berechnen und abzuspeichern.

Operiert die Gruppe A_k durch Rechtsmultiplikation auf der Menge $A_i \setminus G$, so ist der Stabilisator der Nebenklasse A_i gleich der Schnittmenge $A_i \cap A_k$. Dieser Stabilisator kann zum Beispiel durch einen Aufruf des Leiterspiel Light Algorithmus berechnet werden. Die Gruppen E_1, \dots, E_{k-1} können unter Verwendung der entsprechenden Leitern $(A_1 \cap A_j, \dots, A_{j-1} \cap A_j, A_j)$ mit Indizes

j kleiner als k berechnet werden, so dass keine Probleme mit zirkulären Voraussetzungen entstehen.

Ermittle kleinsten Pfad im Block $\Delta_{\{k\}}^q$

Um zu überprüfen, ob die Nebenklasse A_kq kanonisch ist, wird zuerst der kleinste Pfad ρ im Block $\Delta_{\{k\}}^q$ ermittelt und ein Gruppenelement p bestimmt, für das $\rho = (A_1p, \dots, A_kp)$ gilt. Nach Lemma 5 ist A_kq genau dann kanonisch, wenn in keinem der Blöcke der Bahn $\Delta_{\{k\}}^{qB}$ ein Pfad enthalten ist, der kleiner ist als ρ .

Mit Hilfe des Unterprogrammes *FindSmallestPath*, das in Kapitel 7.6.6 beschrieben wird, kann der Pfad ρ ohne großen Aufwand berechnet werden.

Stabilisatoren C_1, \dots, C_{k-1}

In den Splitting- und Fusingschritten des Leiterspiel Light Algorithmus werden die Stabilisatoren C_1, \dots, C_{k-1} der Nebenklassen A_1p bis $A_{k-1}p$ unter der Operation der Gruppe B als bekannt vorausgesetzt. Unter der Voraussetzung, dass die Nebenklassen A_1p bis $A_{k-1}p$ kanonisch sind, können diese Stabilisatoren zum Beispiel durch einen Aufruf des Leiterspiel Light Algorithmus berechnet werden.

Sollte hingegen eine der Nebenklassen A_1p bis $A_{k-1}p$ nicht kanonisch sein, so ist auch die Nebenklasse $A_kp = A_kq$ nicht kanonisch und der Algorithmus kann mit diesem Ergebnis sofort abgebrochen werden.

Denn angenommen eine Nebenklasse A_ip ist nicht kanonisch, so existiert ein $b \in B$ für das gilt, dass A_ipb kleiner ist als A_ip . Nach Lemma 6 enthält in diesem Fall der Block $\Delta_{\{i,k\}}^{pb}$ einen Pfad, der kleiner ist als der Pfad ρ . Daraus folgt, dass auch der Block $\Delta_{\{k\}}^{pb}$ einen Pfad enthält, der kleiner ist als der Pfad ρ . Nach Lemma 5 folgt daraus $A_kpb \prec A_kp$ und die Nebenklasse A_kq ist in diesem Fall nicht kanonisch.

Eingrenzung des Suchraumes

Die auf Seite 135 in Punkt 1 angesprochene Einschränkung der zu untersuchenden Nebenklassen soll an dieser Stelle erläutert werden. Nach Lemma 5 ist $A_k q$ genau dann kanonisch, wenn in keinem der Blöcke der Bahn $\Delta_{\{k\}}^{qB}$ ein Pfad enthalten ist, der kleiner ist als ρ .

Im Folgenden soll unter der Annahme, dass $A_k q$ nicht der kanonische Bahnrepräsentant ist, gezeigt werden, wie einer dieser Pfade, der kleiner ist als ρ , gefunden werden kann. Es bezeichne $b \in B$ ein Element, das so gewählt ist, dass $A_k q b$ der kanonische Repräsentant der Bahn $A_k q^B$ ist. Der kleinste Pfad aus der Menge P_k dessen letzte Komponente gleich $A_k q b$ ist, soll mit $(A_1 r, \dots, A_k r)$ bezeichnet werden.

Wenn $A_k r \prec A_k q$ ist und gleichzeitig $A_k r \neq A_k q$ ist, so folgt nach Lemma 5, dass $(A_1 r, \dots, A_k r) \prec (A_1 p, \dots, A_k p)$ sein muss. Daher existiert ein $1 \leq i < k$, so dass $A_{i+1} r \prec A_{i+1} p$ ist und für alle $h \leq i$ gilt, dass $A_h r = A_h p$ ist. Nach der Definition der Erweiterungsfunktion Υ_i und der Definition der Pfade ist A_{i+1} dann notwendigerweise eine Untergruppe von A_i und der Konstruktionsschritt, in dem die Nebenklasse $A_{i+1} r$ konstruiert wird, ist ein Splittingschritt. Im Folgenden sei i so gewählt, dass $A_{i+1} r \prec A_{i+1} p$ ist und für alle $h \leq i$ gilt, dass $A_h r = A_h p$ ist.

Nach Voraussetzung ist $A_k r = A_k q b$, daraus folgt, dass das Element $r b^{-1} q^{-1}$ in der Gruppe A_k liegt. Die Nebenklasse $E_{i+1} r b^{-1} q^{-1}$ ist daher in der Menge $E_{i+1} \setminus A_k$ enthalten und $\psi_{i+1}(E_{i+1} r b^{-1} q^{-1}) = A_{i+1} r b^{-1}$ liegt in Θ_{i+1} . Diese Nebenklasse $E_{i+1} r b^{-1} q^{-1}$ liegt im Urbild der Nebenklasse $E_i r b^{-1} q^{-1}$ unter dem A_k -Homomorphismus $\varphi : E_{i+1} \setminus A_k \rightarrow E_i \setminus A_k$, der jede Nebenklasse $E_{i+1} a \in E_{i+1} \setminus A_k$ auf die Nebenklasse $E_i a$ abbildet. Daher kann die Nebenklasse $E_{i+1} r b^{-1} q^{-1}$ in einem Splittingschritt aus der Nebenklasse $E_i r b^{-1} q^{-1}$ konstruiert werden. Da nach Voraussetzung $A_i r = A_i p$ ist, muss $\psi_i(E_i r b^{-1} q^{-1}) = A_i r b^{-1} = A_i p b^{-1}$ gelten und die Nebenklasse $A_i r b^{-1}$ liegt in der Bahn $A_i p^B$.

Daher kann die Nebenklasse $A_{i+1} r$ gefunden werden, indem in allen Splittingschritten die Menge aller Nebenklassen berechnet wird, die im Urbild einer Nebenklasse $E_i a \in E_i \setminus A_k$ liegen, für die $\psi_i(E_i a) \in A_i p^B$ gilt. Wenn für

eines dieser Urbilder $E_{i+1}a$ gilt, dass eine Nebenklasse $A_{i+1}aqb'$ aus der Bahn $\psi_{i+1}(E_{i+1}a)^B$ kleiner als die Nebenklasse $A_{i+1}p$ ist, so ist nach Lemma 6 auch der Block $\Delta_{\{i+1,k\}}^{aqb'} \prec \Delta_{\{i+1,k\}}^p$. Dann folgt daraus sofort $\Delta_{\{k\}}^{qb'} \prec \Delta_{\{k\}}^q$ und A_kq ist nicht kanonisch. Der kleinste Pfad im Block $\Delta_{\{k\}}^{qb'}$ wäre in diesem Fall kleiner als der Pfad ρ .

Auf diese Weise würde auch die Nebenklasse $A_i r$ aus der Bahn $\psi_i(E_i r b^{-1} q^{-1})^B$ gefunden werden.

Die Mengen Λ_i und Λ_{i+1}

Es seien ein Pfad $(A_1 r, \dots, A_k r) \prec \rho$ und ein $b \in B$ gegeben, so dass $A_k r = A_k p b$ ist. Wie zuvor auf Seite 138 beschrieben wurde, existiert in diesem Fall ein $1 \leq i < k$, so dass $A_{i+1} r \prec A_{i+1} p$ und für alle $j \leq i$ $A_j r = A_j p$ ist.

Dann muss notwendigerweise gelten, dass $A_{i+1} \leq A_i$ ist. Weiterhin muss gelten, dass der Block $\Delta_{\{i,k\}}^r$, der den Pfad $(A_1 r, \dots, A_k r)$ enthält, in der Bahn des Blockes $\Delta_{\{i,k\}}^{rb^{-1}}$ liegt. Dieser Block $\Delta_{\{i,k\}}^{rb^{-1}}$ ist wiederum ein Urbild von $\Delta_{\{k\}}^p$ unter der Abbildung $\varphi(\{i,k\}, \{k\})$. Daraus folgt, dass $\Delta_{\{k\}}^{rb^{-1}p^{-1}} = \Delta_{\{k\}}^p$ ist und $rb^{-1}p^{-1}$ liegt im Stabilisator A_k des Blockes $\Delta_{\{k\}}^p$. Dann liegt $E_i r b^{-1} p^{-1}$ in der Menge $E_i \setminus A_k$ und $\psi_i(E_i r b^{-1} p^{-1})$ liegt in der Menge Θ_i . Daher müssen im Splittingschritt von A_i nach A_{i+1} nur diejenigen Blöcke untersucht werden, die in der Menge $\Lambda_i = \{\Delta_{\{i,k\}}^g \mid A_k g = A_k p \wedge A_i g \in A_i p^B\}$ enthalten sind.

Die Urbilder aller dieser Blöcke der Menge Λ_i unter dem G -Homomorphismus $\varphi(\{i+1,k\}, \{i,k\})$ werden daraufhin untersucht, ob in deren Bahn ein Block enthalten ist, dessen kleinster Pfad kleiner ist als ρ . Der Block $\Delta_{\{i+1,k\}}^{rb^{-1}}$ liegt im Urbild von $\Delta_{\{i,k\}}^{rb^{-1}} \in \Lambda_i$ unter der Abbildung $\varphi(\{i+1,k\}, \{i,k\})$. Da $A_{i+1} r \prec A_{i+1} p$ ist, gilt auch $\Delta_{\{i+1,k\}}^r \prec \Delta_{\{i+1,k\}}^p$ und der Block, der den Pfad $(A_1 r, \dots, A_k r)$ enthält, wird im Splittingschritt von E_i nach E_{i+1} gefunden.

Für alle $1 \leq j < i$ gilt, dass die Menge $\Lambda_{j+1} = \{\Delta_{\{j+1,k\}}^g \mid A_k g = A_k p \wedge A_{j+1} g \in A_{j+1} p^B\}$ aus der Menge $\Lambda_j = \{\Delta_{\{j,k\}}^g \mid A_k g = A_k p \wedge A_j g \in A_j p^B\}$ konstruiert werden kann.

Wenn $A_j \geq A_{j+1}$ ist, so liegt jeder Block $\Delta_{\{j+1,k\}}^g \in \Lambda_{j+1}$ im Urbild eines Blockes $\Delta_{\{j,k\}}^g$ unter dem G -Homomorphismus $\varphi(\{j+1,k\}, \{j,k\})$. Dieser Block $\Delta_{\{j,k\}}^g$ ist in der Menge Λ_j enthalten, denn $\Delta_{\{j+1,k\}}^g$ liegt in der Menge Λ_{j+1} ,

daher muss auch $A_k g = A_k p$ gelten. Weiterhin folgt aus $A_j \geq A_{j+1}$ und $A_{j+1} g \in A_{j+1} p^B$, dass auch $A_j g$ in der Bahn $A_j p^B$ liegen muss.

Gilt umgekehrt, dass $A_j \leq A_{j+1}$ ist, so existiert zu jedem Block $\Delta_{\{j+1,k\}}^g \in \Lambda_{j+1}$ ein $b' \in B$, so dass $A_{j+1} g b' = A_{j+1} p$ ist. Daher existiert nach Definition 16 ein $g' \in G$, so dass $(A_1 g', \dots, A_k g') = (A_1 g b, \dots, A_j g b, A_{j+1} p, \dots, A_k p)$ ist. Der Block $\Delta_{\{j,k\}}^{g'}$ liegt in der Menge Λ_j und im Urbild des Blockes $\Delta_{\{j+1,k\}}^g$ unter dem G -Homomorphismus $\varphi_{(\{j,k\}, \{j+1,k\})}$.

7.5.3 Splitting Orbits

Es sei A_{i+1} Untergruppe von A_i und $E_i a$ eine Nebenklasse aus der Menge $E_i \backslash A_k$, für die $\psi_i(E_i a)$ in der Bahn $A_i p^B$ liegt. Es sei ein fusionierendes Element $b \in B$ zur Nebenklasse $\psi_i(E_i a) = A_i a q$ gegeben, so dass die Nebenklasse $A_i a q b = A_i p$ ist. Weiterhin sei der Stabilisator C_i der Nebenklasse $A_i p$ in B gegeben, dessen Berechnung bereits in Kapitel 7.5.2 auf Seite 137 beschrieben wurde.

Es bezeichne $\varphi_i : A_{i+1} \backslash G \rightarrow A_i \backslash G$ den G -Homomorphismus von $A_{i+1} \backslash G$ nach $A_i \backslash G$, der jede Nebenklasse $A_{i+1} h \in A_{i+1} \backslash G$ auf die Nebenklasse $A_i h \in A_i \backslash G$ abbildet. Weiterhin bezeichne $\phi_i : E_{i+1} \backslash A_k \rightarrow E_i \backslash A_k$ den A_k -Homomorphismus, der jede Nebenklasse $E_{i+1} h \in E_{i+1} \backslash A_k$ auf die Nebenklasse $E_i h \in E_i \backslash A_k$ abbildet.

Mit Hilfe der Abbildung ϕ_i werden die Urbilder von $E_i a$ unter der Abbildung ϕ_i berechnet. Zu jedem dieser Urbilder $E_{i+1} a'$ wird die Nebenklasse $A_{i+1} a' q = \psi_{i+1}(E_{i+1} a')$ berechnet.

Die Nebenklasse $A_{i+1} a' q b$ liegt im Urbild von $A_i a q b$ unter der Abbildung φ_i . Die Ordnung auf der Menge der Nebenklassen wurde in Kapitel 6.1.2 so definiert, dass die kleinste Nebenklasse der Bahn von $A_{i+1} a' q$ im Urbild der kleinsten Nebenklasse der Bahn von $A_i a q$ unter der Abbildung φ_i liegen muss. Daher kann mit Hilfe des in Kapitel 7.6.5 beschriebenen Algorithmus die kleinste Nebenklasse $A_{i+1} a' q b c$ aus der Bahn der Nebenklasse $A_{i+1} a' q b$ unter der Operation der Gruppe C_i berechnet werden. Die so berechnete Nebenklasse $A_{i+1} a' q b c$ ist gleichzeitig der kanonische Repräsentant der Bahn der Nebenklasse $A_{i+1} a' q$ unter der Operation der Gruppe B .

Wenn der so berechnete kanonische Repräsentant $A_{i+1}a'qbc$ kleiner ist als die Nebenklasse $A_{i+1}p$, so ist auch der Block $\Delta_{\{i+1,k\}}^{a'qbc} = \Delta_{\{i+1,k\}}^{qbc}$ kleiner als der Block $\Delta_{\{i+1,k\}}^p$. Dann muss, da a' in der Gruppe A_k liegt, der Block $\Delta_{\{k\}}^{a'qbc}$ identisch mit dem Block $\Delta_{\{k\}}^{qbc}$ sein und es gilt $\Delta_{\{k\}}^{qbc} \prec \Delta_{\{k\}}^q$. Daraus folgt wiederum $A_kqbc \prec A_kq$ und da bc in der Gruppe B liegt, kann A_kq nicht der kanonische Repräsentant sein.

Wenn der so berechnete kanonische Repräsentant $A_{i+1}a'qbc$ größer als die Nebenklasse $A_{i+1}p$ ist, so liegen die beiden Bahnrepräsentanten $A_{i+1}a'qbc$ und $A_{i+1}p$ nicht in derselben Bahn. Daher ist auch $\Delta_{\{i+1,k\}}^{a'q}$ nicht in der Menge Λ_{i+1} enthalten und kann von der weiteren Untersuchung ausgeschlossen werden.

Wenn der so berechnete kanonische Repräsentant $A_{i+1}a'qbc$ identisch zur Nebenklasse $A_{i+1}p$ ist, so liegt der Block $\Delta_{\{i+1,k\}}^{a'q}$ in der Menge Λ_{i+1} und muss in den folgenden Fusing- und Splittingschritten weiter untersucht werden.

7.5.4 Fusing Orbits

In den Fusingschritten von A_i nach A_{i+1} des Leiterspiel Light Algorithmus besteht die einzige Aufgabe darin, aus der Menge Λ_i die Menge Λ_{i+1} zu berechnen. In Kapitel 7.5.2 auf Seite 139 wurde bereits gezeigt, dass jeder Block der Menge Λ_{i+1} ein Urbild aus der Menge Λ_i unter der Abbildung $\varphi_{(\{i,k\},\{i+1,k\})}$ besitzt.

Es sei A_i Untergruppe von A_{i+1} und $\Delta_{\{i,k\}}^{aq}$ ein Block aus der Menge Λ_i . Da $\Delta_{\{i,k\}}^{aq} \in \Lambda_{i+1}$ ist, liegt die Nebenklasse $A_i a q$ in der Bahn $A_i p^B$. Es bezeichne b ein fusionierendes Element zur Nebenklasse $A_i a q$ aus der Gruppe B , so dass die Nebenklasse $A_i a q b = A_i p$ ist. Es sei der Stabilisator C_{i+1} der Nebenklasse $A_{i+1}p$ in B gegeben, dessen Berechnung bereits in Kapitel 7.5.2 auf Seite 137 beschrieben wurde.

Im Fundamentallemma 1 wurde ein G -Isomorphismus beschrieben, der für alle $g \in G$ den Block $\Delta_{\{i,k\}}^g$ auf die Nebenklassen $E_i g$ abbildet. Durch diesen G -Isomorphismus wird der Block $\Delta_{\{i,k\}}^{aq}$ auf die Nebenklasse $E_i a q$ abgebildet. Daraus kann durch Rechtsmultiplikation mit q^{-1} die Nebenklasse $E_i a$

berechnet werden. Da $\Delta_{\{k\}}^{aq} = \Delta_{\{k\}}^q$ ist, liegt $aq q^{-1}$ in der Gruppe A_k . Daher liegt die Nebenklasse $E_i a$ in der Menge $E_i \backslash A_k$.

Es bezeichne $\varphi_i : A_i \backslash G \rightarrow A_{i+1} \backslash G$ den G -Homomorphismus von $A_i \backslash G$ nach $A_{i+1} \backslash G$, der jede Nebenklasse $A_i h \in A_i \backslash G$ auf die Nebenklasse $A_{i+1} h \in A_{i+1} \backslash G$ abbildet. Weiterhin bezeichne $\phi_i : E_i \backslash A_k \rightarrow E_{i+1} \backslash A_k$ den A_k -Homomorphismus, der jede Nebenklasse $E_i h \in E_i \backslash A_k$ auf die Nebenklasse $E_{i+1} h \in E_{i+1} \backslash A_k$ abbildet.

Das Bild von $\Delta_{\{i,k\}}^{aq}$ unter der Abbildung $\varphi_{(\{i,k\},\{i+1,k\})}$ ist der Block $\Delta_{\{i+1,k\}}^{aq}$. Nach Voraussetzung gilt $A_i a q b = A_i p$. Da A_i eine Untergruppe von A_{i+1} ist, muss auch $A_{i+1} a q b = A_{i+1} p$ sein. Zusammen folgt daraus, dass $\Delta_{\{i+1,k\}}^{aq} \in \Lambda_{i+1}$ ist.

Um zu vermeiden, dass der Block $\Delta_{\{i+1,k\}}^{aq}$ mehrfach konstruiert wird, soll genau einer der Blöcke aus der Menge Λ_i ausgewählt werden, um daraus den Block $\Delta_{\{i+1,k\}}^{aq}$ zu konstruieren. Zur Bestimmung dieses ausgewählten Blockes soll wie folgt vorgegangen werden:

Wie in Abbildung 7.6 dargestellt und in Lemma 14 bewiesen wurde, ord-

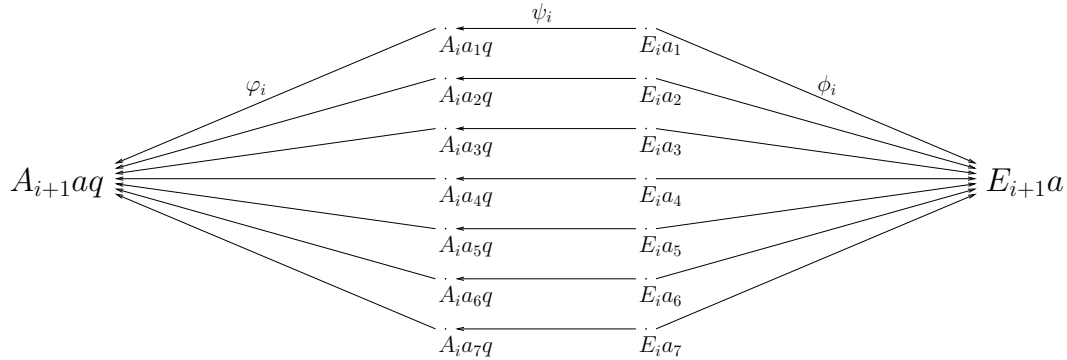


Abbildung 7.6: Bijektion zwischen den Urbildmengen von $A_{i+1} a q$ und $E_{i+1} a$

net die Abbildung ψ_i jedem Urbild von $A_{i+1} a q$ unter der Abbildung φ_i genau ein Urbild von $E_{i+1} a$ unter der Abbildung ϕ_i zu. Zuerst soll die Menge aller Nebenklassen berechnet werden, die sowohl im Urbild von $A_{i+1} a q$ unter der Abbildung φ_i als auch in der Bahn $A_i p^B$ liegen. Aus dieser Menge soll die kleinste Nebenklasse ausgewählt werden. Wenn $\psi_i(E_i a)$ identisch mit der auf diese Weise berechneten kleinsten Nebenklasse ist, soll aus dem Block $\Delta_{\{i,k\}}^{aq}$

der Block $\Delta_{\{i+1,k\}}^{aq}$ berechnet werden.

Da $A_{i+1}aq$ in der Bahn der Nebenklasse $A_{i+1}p$ liegt, kann die kleinste Nebenklasse aus dem Urbild von $A_{i+1}aq$, die gleichzeitig in der Bahn von $A_i p$ liegt, mit Hilfe des Stabilisators $bC_{i+1}b^{-1}$ und des in Kapitel 7.6.5 beschriebenen Algorithmus berechnet werden.

Wenn die Nebenklasse $E_i a$ nicht der ausgewählten Nebenklasse entspricht, so muss die Nebenklasse $E_i a$ nicht weiter untersucht werden. Andernfalls wird aus der Nebenklasse $E_i a$ die Nebenklasse $E_{i+1} a$ berechnet und die Suche nach dem kleinsten Pfad wird mit dem Block $\Delta_{\{i+1,k\}}^{aq}$ fortgesetzt.

7.5.5 Berechnung des Stabilisators von $A_k q$

Unter der Voraussetzung, dass $A_k q$ der kanonische Repräsentant der Bahn $A_k q^B$ ist, soll mit dem Leiterspiel Light Algorithmus auch der Stabilisator C_k der Nebenklasse $A_k q$ unter der Operation der Gruppe B berechnet werden.

Wenn A_k eine Untergruppe von A_{k-1} ist, so ist der Stabilisator C_k nach dem Homomorphieprinzip 3.2.1 eine Untergruppe des Stabilisators C_{k-1} , dessen Berechnung in Kapitel 7.5.2 auf Seite 137 beschrieben wurde. Daher kann C_k in diesem Fall mit dem Unterprogramm *ReduceStab* aus Kapitel 7.6.2 berechnet werden.

Ist A_{k-1} hingegen eine Untergruppe von A_k , so ist nach dem Homomorphieprinzip 3.2.1 der Stabilisator C_{k-1} eine Untergruppe des Stabilisators von $A_k q$. Im Fundamentallema 1 wurde ein G -Isomorphismus beschrieben, der für alle $g \in G$ den Block $\Delta_{\{k-1,k\}}^g$ auf die Nebenklassen $A_{k-1}g$ abbildet. Durch diesen G -Isomorphismus wird jeder Block $\Delta_{\{k-1,k\}}^{aq} \in \Lambda_{k-1}$ auf die Nebenklasse $A_{k-1}aq$ abgebildet. Zu jedem Block $\Delta_{\{k-1,k\}}^{aq}$ aus der Menge Λ_{k-1} wird während des Leiterspiel Light Algorithmus ein fusionierendes Element $b \in B$ berechnet, so dass $A_{k-1}aqb = A_{k-1}p$ ist. Aus dem Homomorphieprinzip folgt, dass jedes dieser fusionierenden Elemente im Stabilisator C_k liegen muss. Denn sowohl $\Delta_{\{k-1,k\}}^{aq}$ als auch $\Delta_{\{k-1,k\}}^p$ liegen im Urbild des Blockes $\Delta_{\{k\}}^q$ unter dem G -Homomorphismus $\varphi_{(\{k-1,k\},\{k\})}$. Der Stabilisator von $A_k q$ ist aufgrund der

beschriebenen Isomorphie zwischen den Blöcken und den Nebenklassen identisch mit dem Stabilisator der Blockes $\Delta_{\{k\}}^q$. Unter Verwendung des Satzes von Lagrange 3.1.1 und des Homomorphieprinzips kann gezeigt werden, dass die kleinste Gruppe, die sowohl den Stabilisator C_k als auch alle fusionierenden Elemente von Blöcken aus der Menge Λ_{k-1} enthält, der volle Stabilisator von $A_k q$ sein muss.

Diese kleinste Gruppe, die sowohl den Stabilisator C_k als auch alle fusionierenden Elemente von Blöcken aus der Menge Λ_{k-1} enthält, kann mit Hilfe des Unterprogramms *ExtendGroup* aus Kapitel 7.6.4 berechnet werden.

7.6 Unterprogramme des Kanonizitätstests

7.6.1 Identitäten und inverse Homomorphismen

Um das Leiterspiel Light, den Breadthfirst StroLL und den Depthfirst StroLL Algorithmus durchführen zu können, muss ein Algorithmus bereitgestellt werden, mit dem überprüft werden kann, ob zwei Nebenklassen identisch sind.

Für manche Gruppen kann sehr leicht festgestellt werden, ob zwei Nebenklassen identisch sind. Im Allgemeinen ist dies jedoch für zwei Nebenklassen einer Untergruppe einer endlich erzeugten Gruppe nicht entscheidbar. Betrachtet man die Menge der Nebenklassen der trivialen Untergruppe einer endlich erzeugten Gruppe, so wird dieses Entscheidungsproblem „Wortproblem“ genannt.

Beim Leiterspiel Light, dem Breadthfirst StroLL und dem Depthfirst StroLL Algorithmus wird jedoch vorausgesetzt, dass der Gruppenindex zweier aufeinander folgender Gruppen der Leiter immer endlich sein muss. Ist U eine Untergruppe einer Gruppe V und ist der Gruppenindex $(V : U)$ endlich, so kann mit dem Todd-Coxeter-Algorithmus die Menge der Nebenklassen von U in V berechnet werden [TC36, CDHW73]. Die Tabellen, die bei der Durchführung des Todd-Coxeter-Algorithmus erstellt werden, ermöglichen es, für die entsprechenden Nebenklassen zu entscheiden, ob diese identisch sind oder nicht. Es bezeichne U, V, G drei Gruppen mit $U \leq V \leq G$ und endlichem Gruppenindex $(G : U)$. Weiterhin bezeichne $\varphi : U \backslash G \rightarrow V \backslash G$ einen G -Homomorphismus, der für alle $g \in G$ die Nebenklasse Ug auf die Nebenklasse Vg abbildet. Für alle $g \in G$ kann der Todd-Coxeter-Algorithmus auch dazu verwendet werden, die Urbilder der Nebenklasse Vg unter dem G -Homomorphismus φ zu berechnen.

Es seien eine starke Untergruppenleiter (A_1, \dots, A_n) , ein $k \leq n$ und ein Element $g \in G$ gegeben. Es soll überprüft werden, ob das Element g in der Gruppe A_k enthalten ist.

Für alle $j \leq k$ bezeichne E_j die Gruppe $A_k \cap A_j$. Weiterhin sei für alle $1 < j \leq k$ mit $A_{j-1} \geq A_j$ ein minimales Repräsentantensystem $T_j \subseteq G$ aller Nebenklassen $E_j \backslash E_{j-1}$ gegeben. Dieses minimale Repräsentantensystem T_j kann zu einem minimalen Repräsentantensystem S_j aller Nebenklassen $A_j \backslash A_{j-1}$ er-

weitert werden. Für alle $1 < j \leq k$ mit $A_{j-1} \geq A_j$ sei eine Abbildung $\phi_j : A_{j-1} \rightarrow S_j$ gegeben, mit der Eigenschaft, dass für alle $h \in A_{j-1}$ das Element $h\phi_j(h)$ in der Gruppe A_j liegt. Dann kann mit folgendem Algorithmus herausgefunden werden, ob das Element g in der Nebenklasse A_k enthalten ist:

Pseudocode Membership Test

<i>Input:</i>	(A_1, \dots, A_n)	\\Subgroup Ladder
	Index k	\\ $k \leq n$
	$g \in G$	\\check if $g \in A_k$
<i>Output:</i>	Bool	

```

1  GroupElement  $s \leftarrow 1_G$ 
2  foreach (  $j \in \{2, \dots, k\}$  )           \\go through in increasing order
3      if (  $A_j \leq A_{j-1}$  )
4           $s \leftarrow \phi_j(g)$ 
5          if (  $s \in T_j$  )
6               $g \leftarrow gs$ 
7          else
8              return false
9          endif
10 endif
11 end
12 return true

```

Der Todd-Coxeter-Algorithmus muss nicht bei jeder Identitätsprüfung und bei jeder Berechnung der Urbilder einer Nebenklasse erneut durchgeführt werden. Es genügt vielmehr, wenn der Algorithmus nur einmal für die entsprechenden Gruppen ausgeführt wird und die Ergebnisse gespeichert werden. Die Gruppen, für die der Todd-Coxeter-Algorithmus durchgeführt werden muss, lassen sich allein anhand der gegebenen Untergruppenleiter bestimmen. Daher genügt es, zu einer gegebenen Leiter den Todd-Coxeter-Algorithmus für jedes benötigte Gruppenpaar nur ein einziges Mal auszuführen. Aus diesen Ergebnissen können für alle Indizes $j \leq k$ die entsprechenden Abbildungen ϕ_j berechnet werden. Danach können beliebig viele Problemstellungen zu dieser

gegebenen Leiter bearbeitet werden, ohne dass der Algorithmus erneut ausgeführt werden müsste.

Für viele spezielle Gruppen existieren effizientere Methoden, um zu überprüfen, ob zwei Nebenklassen identisch sind oder um zu einer gegebenen Nebenklasse die Menge der Urbilder dieser Nebenklassen unter einem der genannten G -Homomorphismen zu berechnen. Auf diese Methoden einzugehen, würde aber den Rahmen dieser Arbeit sprengen. Daher soll an dieser Stelle nur noch auf den Membership Test von C. Sims [Sim71] verwiesen werden, der für viele Permutationsgruppen geeignet ist. Dieser Membership Test wurde durch M. Jerrum [Jer86] und G. Cooperman, L. Finkelstein und P. W. Purdom [CFP89] weiterentwickelt.

7.6.2 ReduceStab

Es seien eine Gruppe G , eine Untergruppe U von G und ein Element $g \in G$ gegeben. Weiterhin sei eine Untergruppe C von G gegeben, die durch Rechtsmultiplikation auf der Menge der Nebenklassen $U \backslash G$ operiert. Mit dem Unterprogramm *ReduceStab* soll der Stabilisator D der Nebenklasse Ug unter der Operation der Gruppe C berechnet werden.

Im Allgemeinen sind derartige Problemstellungen nicht leicht zu lösen. Wenn eine entsprechende Untergruppenleiter bekannt ist, so kann der Breadthfirst StroLL oder der in Kapitel 7.6.3 beschriebene Algorithmus zusammen mit dem Depthfirst StroLL oder dem Leiterspiel Light Algorithmus zur Lösung dieser Problemstellung verwendet werden.

Wenn aber bereits bekannt ist, dass der Gruppenindex $(C : D)$ des gesuchten Stabilisators D in der Gruppe C klein ist, dann kann diese Problemstellung auch mit Hilfe des erweiterten Bahnenalgorithmus gelöst werden. Diese Situation taucht beim Leiterspiel-Algorithmus häufig auf und soll im Folgenden beschrieben werden.

Es sei (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n mit der Eigenschaft, dass der Gruppenindex zwischen je zwei aufeinanderfolgenden

Gruppen der Leiter klein und insbesondere endlich ist. Es sei ein Element $g \in G$ und ein Index $1 < i \leq n$ gegeben, für den $A_i \leq A_{i-1}$ gilt. Weiterhin sei eine Untergruppe $B \leq G$ gegeben, die für alle $1 \leq j \leq n$ durch Rechtsmultiplikation auf der Menge $A_j \backslash G$ operiert. Die Untergruppe $C \leq G$ sei gleich dem Stabilisator der Nebenklasse $A_{i-1}g$ unter der Operation der Gruppe B . Nach dem Homomorphieprinzip 3.2 ist der Stabilisator D der Nebenklasse $A_i g$ eine Untergruppe von C .

Üblicherweise hängt die Laufzeit des erweiterten Bahnenalgorithmus von der Anzahl der Erzeuger der Gruppe C und von der Größe des Gruppenindex $(C : D)$ ab. Die Abbildung $\varphi : A_i \backslash G \rightarrow A_{i-1} \backslash G$, die für alle $h \in G$ die Nebenklasse $A_i h$ auf die Nebenklasse $A_{i-1} h$ abbildet, ist ein G -Homomorphismus. Aus dem Homomorphieprinzip folgt, dass der Gruppenindex $(C : D)$ maximal so groß sein kann, wie der Gruppenindex $(A_{i-1} : A_i)$. Daher sollte bei der Auswahl der Untergruppenleiter, die für den Leiterspiel-Algorithmus verwendet wird, darauf geachtet werden, dass die Gruppenindizes zwischen je zwei aufeinander folgenden Gruppen der Leiter immer klein sind.

Das Untergruppenlemma nach Schreier kann dazu verwendet werden, den Bahnenalgorithmus so zu erweitern, dass dieser auch die Erzeuger des gesuchten Stabilisators berechnet.

Lemma 21. (*Schreier*)

Es sei G eine Gruppe, S ein Erzeugendensystem von G und U eine Untergruppe von G . Es sei $\mathcal{T} \subseteq G$ ein minimales Repräsentantensystem aller Rechtsnebenklassen von U in G , das bedeutet:

$$\forall g \in G \exists! t \in \mathcal{T} : Ug = Ut$$

Weiterhin sei eine Abbildung $\phi : G \rightarrow \mathcal{T}$ gegeben, mit der Eigenschaft, dass für alle $g \in G$ und $t = \phi(g)$ gilt, dass $Ug = Ut$ ist.

Dann ist die Menge $\{tsr \mid t \in \mathcal{T}, s \in S, r = \phi(ts)^{-1}\}$ ein Erzeugendensystem der Gruppe U .

Beweis. (in Anlehnung an den Beweis in [Hal59, HR48])

Es sei u ein beliebiges Element aus der Untergruppe U . Es sei $x_0 = \phi(1_G)$ und $v = x_0^{-1}ux_0 \in U$. Da S ein Erzeugendensystem ist, läßt sich v als die

Verknüpfung endlich vieler Elemente $v = s_1 \dots s_n$ mit $s_1, \dots, s_n \in S$ schreiben. Für alle $i \leq n$ sei $b_i = s_1 \dots s_i$ und $x_i = \phi(b_i)$. Da v in U liegt, gilt auch $x_n = \phi(v) = \phi(1_G)$. Dann ist für alle $i < n$ das Element $x_{i+1} = \phi(b_{i+1}) = \phi(b_i s_{i+1}) = \phi(x_i s_{i+1})$. Es gilt:

$$\begin{aligned} u &= x_0 v x_0^{-1} = x_0 s_1 \dots s_n x_n^{-1} \\ u &= x_0 s_1 (x_1^{-1} x_1) s_2 (x_2^{-1} x_2) \dots s_{n-1} (x_{n-1}^{-1} x_{n-1}) s_n x_n^{-1} \\ u &= (x_0 s_1 x_1^{-1}) (x_1 s_2 x_2^{-1}) \dots (x_{n-1} s_n x_n^{-1}) \end{aligned}$$

Für alle $i \leq n$ gilt aber, dass $x_{i-1} s_i x_i^{-1} = x_{i-1} s_i \phi(x_{i-1} s_i)^{-1}$ ist.

□

Operiert die Gruppe G auf einer Menge Ω , so kann mit dem einfachen Bahnenalgorithmus die Bahn eines Elementes δ unter der Operation einer Gruppe C berechnet werden. Dem Bahnenalgorithmus wird eine Menge S von Erzeugern der Gruppe C und das Element δ übergeben. Ist die Bahn von δ endlich, so berechnet der Bahnenalgorithmus daraus die Menge Δ , die Bahn des Elementes δ .

Beim erweiterten Bahnenalgorithmus wird zusätzlich ein Abbildung ϕ berechnet. Die Abbildung ϕ wird so konstruiert, dass nach Durchführung des Algorithmus für alle $\omega \in \Delta$ gilt, dass $\delta^{\phi(\omega)} = \omega$ ist. Die Menge E enthält am Ende eine Menge von Erzeugern, so dass für den gesuchten Stabilisator $C_\delta = \langle E \rangle$ gilt.

Um den Bahnenalgorithmus zur Bestimmung des Stabilisators verwenden zu können, wird ein Algorithmus benötigt, mit dessen Hilfe festgestellt werden kann, ob zwei Nebenklassen Ag_1 und Ag_2 identisch sind. In Kapitel 7.6.1 wurden mehrere Algorithmen beschrieben, die dazu geeignet sind.

Die Anzahl der Erzeuger in der Menge E kann mitunter sehr groß werden, was insbesondere dann ein Nachteil sein kann, wenn diese Erzeuger wieder als Eingabe für einen weiteren Durchlauf des Bahnenalgorithmus verwendet werden sollen. Denn die Laufzeit des Bahnenalgorithmus wächst linear mit der Anzahl der Erzeuger. Daher sollte der erweiterte Bahnenalgorithmus mit anderen Algorithmen kombiniert werden, die in der Lage sind, die Anzahl der

Erzeuger zu reduzieren.

Die Suche nach Methoden, mit denen die Anzahl der Erzeuger reduziert werden kann, bildet einen eigenen Forschungsbereich. Denn im Allgemeinen ist schon allein das Wortproblem für endlich repräsentierte Gruppen nicht entscheidbar. Daher ist die auch Frage, ob zwei Gruppenelemente $s_{i_1}s_{i_2}\dots s_{i_n}$ und $s_{j_1}s_{j_2}\dots s_{j_m}$ mit $s_{i_1}, s_{i_2}, \dots, s_{i_n}, s_{j_1}, s_{j_2}, \dots, s_{j_m} \in S$ dasselbe Gruppenelement bezeichnen, im Allgemeinen nicht entscheidbar. Zu einer Menge von Erzeugern kann jedoch in manchen Fällen mit Hilfe des Knuth-Bendix Vervollständigungsverfahrens eine kleinere Erzeugermenge gefunden werden [KB70, Sim94].

Erweiterter Bahnalgorithmus

<i>Input:</i>	Set S	$\backslash\backslash C = \langle S \rangle$
	SetElement δ	
<i>Referenced :</i>	Set Δ	$\backslash\backslash \Delta = \{\delta\}$
	$\phi : \Delta \longrightarrow C$	$\backslash\backslash \phi(\delta) = 1_C$
	Set E	$\backslash\backslash \langle E \rangle \leq C_\delta$

```

1  foreach (  $\delta \in \Delta$  )
2      foreach (  $g \in S$  )
3           $\omega \leftarrow \delta^g$ 
4          if (  $\omega \notin \Delta$  )
5               $\Delta \leftarrow \Delta \cup \omega$ 
6               $\phi(\omega) \leftarrow \phi(\delta)g$ 
7          else
8               $E \leftarrow E \cup \{\phi(\delta)g\phi(\omega)^{-1}\}$ 
9          endif
10     end
11 end
12 exit
```

Im Folgenden soll noch auf einen Spezialfall eingegangen werden, für den ein besonders effizienter Algorithmus existiert. Es sei Δ eine kleine endliche Menge und C eine Permutationsgruppe auf der Menge Δ . Wenn zum gesuchten Stabilisator der Nebenklasse Ag ein Element $\delta \in \Delta$ existiert, so dass C_δ

identisch mit dem gesuchten Stabilisator ist, kann ein sogenannter „Schreier-Vektor“ zur Berechnung des Stabilisators verwendet werden.

Der Schreier-Sims Algorithmus [Sim71] kann zur Berechnung eines Schreier-Vektors zur Gruppe C eingesetzt werden, einer Datenstruktur zur Speicherung einer Permutationsgruppe. In [CF94] beschreiben G. Cooperman und L. Finkelstein zwei verschiedene Algorithmen, um einen Basiswechsel durchzuführen. Insbesondere der Basiswechsel vom Typ „Las Vegas“ ist sehr effizient und eignet sich hervorragend zur Berechnung dieser Stabilisatoren. Indem das Element $\delta \in \Delta$ durch einen Basiswechsel an die erste Position der Basis des Schreier-Vektors permutiert wird, kann der Stabilisator von δ anschließend mühelos abgelesen werden.

7.6.3 Canonizer

Es sei eine Gruppe G gegeben und es sei (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Zu je zwei aufeinanderfolgenden Gruppen der Leiter soll der Gruppenindex klein und insbesondere endlich sein. Für alle $1 \leq i \leq n$ wird die Menge der Nebenklassen von A_i in G mit Ω_i bezeichnet. Auf jeder dieser Mengen Ω_i sei eine Totalordnung gegeben, die die in Kapitel 6.1 geforderte Bedingung an die Ordnungen der Nebenklassen erfüllt. Es sei eine weitere Untergruppe B von G gegebene, die durch Rechtsmultiplikation auf jeder der Mengen $\Omega_1, \dots, \Omega_n$ operiert.

Es seien ein $1 \leq k \leq n$ und ein Element $q \in G$ gegeben. Im Unterprogramm *Canonizer* soll zur Nebenklasse $A_k q$ sowohl deren kanonischer Bahnrepräsentant als auch der Stabilisator dieses kanonischen Bahnrepräsentanten unter der Operation der Gruppe B bestimmt werden. Optional kann der Methode *Canonizer* eine Untergruppe des Stabilisators der Nebenklasse $A_k g$ übergeben werden, die es ermöglicht, die Berechnungen zu beschleunigen.

Der in Kapitel 7.3 beschriebene Breadthfirst StroLL kann ohne jede Änderung zur Berechnung dieses kanonischen Bahnrepräsentanten und dessen Stabilisators eingesetzt werden. Auch das Leiterspiel Light und der Depthfirst StroLL können so abgewandelt werden, dass zu einer gegebenen Nebenklasse $A_k q$ der zugehörige kanonische Bahnrepräsentant und dessen Stabilisator

berechnet werden. Ist die gegebene Nebenklasse A_kq kanonisch, so kann der zugehörige Stabilisator bereits mit Hilfe von jedem der drei Algorithmen aus Kapitel 7 berechnet werden, ohne dass dazu Änderungen an den Algorithmen notwendig wären. Sobald aber zur Nebenklasse A_kq ein Element $b \in B$ gefunden wurde, für das $A_kqb \prec A_kq$ gilt, werden beim Depthfirst StroLL und beim Leiterspiel Light alle weiteren Berechnungen abgebrochen und es wird das Ergebnis „ A_kq nicht kanonisch“ zurückgegeben.

Anhand des Depthfirst StroLL Algorithmus aus Kapitel 7.4 soll beschrieben werden, welche Anpassungen notwendig sind, um den Depthfirst StroLL oder das Leiterspiel Light zur Berechnung des kanonischen Repräsentanten und dessen Stabilisator einzusetzen. Die erste Änderung am Depthfirst StroLL Algorithmus ist, dass zusätzlich zu der Information, dass die untersuchte Nebenklasse A_kq nicht kanonisch ist, ein beliebiges Element $b \in B$ zurückgegeben wird, für das $A_kqb \prec A_kq$ ist. Weiterhin soll die zu diesem Zeitpunkt bekannte Untergruppe des Stabilisators der Nebenklasse A_kq zurückgegeben werden.

Nach Satz 3.1.2 existiert ein Homomorphismus von Gruppenoperationen, der jede Nebenklasse auf ihren Stabilisator abbildet. Daher ist der Stabilisator der Nebenklasse A_kqb gleich dem b -konjugierten Stabilisator der Nebenklasse A_kq . Wenn der so abgewandelte Depthfirst StroLL Algorithmus abbricht, weil eine kleinere Nebenklasse A_kqb gefunden wurde, so kann aus einer Untergruppe C_k des Stabilisators der Nebenklasse A_kq die Untergruppe $b^{-1}C_kb$ berechnet werden, die wiederum eine Untergruppe des Stabilisators der Nebenklasse A_kqb ist.

Anschließend kann der Depthfirst StroLL Algorithmus erneut aufgerufen werden, nur diesmal unter Eingabe der Nebenklasse A_kqb und des Stabilisators $b^{-1}C_kb$. Das Ergebnis dieses Aufrufs ist dann entweder eine noch kleinere Nebenklasse A_kqb' zusammen mit einer eventuell vergrößerten Untergruppe des Stabilisators der Nebenklasse A_kqb oder die Nebenklasse A_kqb ist bereits kanonisch. Wenn eine kleinere Nebenklasse gefunden wurde, so wird die beschriebene Vorgehensweise solange wiederholt, bis der kanonische Repräsentant gefunden wurde. In diesem Fall wird der Depthfirst StroLL Algorithmus den vollen Stabilisator dieser kanonischen Nebenklasse bereitstellen.

Da eine Totalordnung auf der Menge der Nebenklassen Ω_k gegeben ist und die Menge Ω_k endlich ist, führt diese Vorgehensweise zu einem Algorithmus, der zu jeder Nebenklasse $A_k g$ den kanonischen Bahnrepräsentanten und dessen Stabilisator berechnet. Denn bei jedem der Aufruf des Depthfirst StroLL Algorithmus wird entweder eine Nebenklasse gefunden, die kleiner als die kleinste bisher bekannte Nebenklasse der Bahn $A_k q^B$ ist, oder der Algorithmus endet damit, dass der kanonische Repräsentant gefunden wird. Die Effizienz des ursprünglichen Depthfirst StroLL Algorithmus bleibt dabei weitgehend erhalten.

7.6.4 ExtendGroup

In diesem Kapitel werden verschiedene Methoden beschrieben, wie eine Untergruppe U einer Gruppe B um einen Erzeuger b erweitert werden kann. Genauer gesagt soll die kleinste Untergruppe V von B berechnet werden, die sowohl U als auch den Erzeuger b enthält.

Die Lösung dieses Problems hängt stark davon ab, welche Datenstruktur zur Speicherung der Gruppe verwendet werden soll. Im einfachsten Fall ist bereits die Gruppe U in Form eines Erzeugendensystems gegeben und es soll lediglich ein Erzeugendensystem der Gruppe V bestimmt werden. In diesem Fall sind keine weiteren Berechnungen notwendig, sondern es genügt, das Element b zur Menge der Erzeuger von U hinzuzufügen. Denn diese Menge ist bereits ein Erzeugendensystem der gesuchten Gruppe V .

Ein Nachteil dieser Methode ist, dass das Erzeugendensystem auf diese Weise sehr groß werden kann. Beim Breadthfirst StroLL, beim Depthfirst StroLL und beim Leiterspiel Light Algorithmus wurde vorausgesetzt, dass der Index je zweier aufeinander folgender Gruppen der Leiter klein und insbesondere immer endlich sein muss. Daher kann auch bei den Erweiterungsproblemen, die im Zusammenhang mit den beschriebenen Algorithmen auftauchen, davon ausgegangen werden, dass der Gruppenindex $(V : U)$ endlich ist. J. A. Todd und H. S. M. Coxeter fanden eine sehr allgemeine Methode, die es erlaubt, die Nebenklassen von U in V zu berechnen [TC36]. Abhängig von den Eigenschaften der untersuchten Gruppe existiert eine ganze Reihe weiterer Verfahren,

deren Beschreibung jedoch den Rahmen dieser Arbeit sprengen würde. Daher soll an dieser Stelle nur noch auf den Spezialfall eingegangen werden, wenn die Gruppe B in Form einer sogenannten „Short-Schreier-Vector“ Datenstruktur vorliegt.

Ist die Gruppe B eine Permutationsgruppe und soll die Gruppe V in Form eines Short-Schreier-Vektors gespeichert werden, so kann der Algorithmus von G. Cooperman, L. Finkelstein und P. W. Purdom [CFP89] zur Berechnung eines starken Erzeugendensystems verwendet werden. Ein weiterer Vorteil, der dafür spricht, die Gruppen, wenn möglich, in Form eines Short-Schreier-Vektors zu speichern, ist es, dass auf diese Weise auch die Anzahl der Erzeuger leicht unter Kontrolle gehalten werden können. Mit Hilfe des von G. Cooperman, L. Finkelstein und P. W. Purdom beschriebenen Algorithmus zur Berechnung eines starken Erzeugendensystems kann die Anzahl der Erzeuger auf $21 \log_2(|V|)$ beschränkt werden.

7.6.5 FindOrbitRep

Im Unterprogramm *FindOrbitRep* soll zu einer Nebenklasse Ag und gegebener Gruppe B der kleinste Repräsentant der Bahn Ag^B gefunden werden. Ist die Bahn Ag^B sehr lang, so ist diese Aufgabe schwer zu lösen und es müssen Kanonisierungsalgorithmen wie der in Kapitel 7.6.3 beschriebene eingesetzt werden. Ist die Bahn Ag^B hingegen kurz, so kann wie in Kapitel 7.6.2 der Bahnalgorithmus zur Lösung eingesetzt werden. Der Bahnalgorithmus konstruiert dann alle Nebenklassen, die in der Bahn Ag^B liegen.

Unter diesen Nebenklassen kann dann unter Verwendung eines Kanonizitätsprädikates der kanonische Bahnrepräsentant identifiziert werden. Ist auf der Menge der Nebenklassen Ag^B eine Totalordnung gegeben, so kann zum Beispiel die kleinste Nebenklasse dieser Bahn als die kanonische ausgezeichnet werden.

7.6.6 FindSmallestPath

Es sei G eine Gruppe und (A_1, \dots, A_n) eine starke Untergruppenleiter von G nach A_n . Die Gruppenindizes zwischen je zwei aufeinander folgenden Gruppen der Untergruppenleiter seien klein und insbesondere auch endlich. Es sei ein $k \leq n$ und eine Nebenklasse $A_k q$ mit $q \in G$ gegeben. In der Vorbereitungsphase des Depthfirst StroLL und Leiterspiel Light Algorithmus, die in Kapitel 7 beschrieben werden, soll der kleinste Pfad $(A_1 p, \dots, A_k p) \in P_k$ berechnet werden, dessen letzte Komponente $A_k p$ gleich der gegebenen Nebenklasse $A_k q$ ist. Die dabei zugrunde gelegte Ordnung ist die in Kapitel 6.2 beschriebene Ordnung auf der Menge P_k .

Jeder Pfad aus der Menge P_k , dessen letzte Komponente gleich $A_k q$ ist, liegt im Block $\Delta_{\{k\}}^q$. Da $A_1 = G$ ist und die erste Komponente aller Pfade aus der Menge P_k daher gleich G ist, ist der Block $\Delta_{\{1,k\}}^q$ identisch mit dem Block $\Delta_{\{k\}}^q$. Dieser Block kann in kleinere Blöcke zerlegt werden, so dass alle Pfade, deren zweite Komponenten identisch sind, jeweils in demselben Block liegen.

Nach Lemma 16 ist die Urbildmenge des Blockes $\Delta_{\{1,k\}}^q$ unter der Abbildung $\varphi_{(\{1,2,k\},\{1,k\})}$ gleich $\{\Delta_{\{1,2,k\}}^{aq} \mid a \in A_1 \cap A_k\}$. Da $A_1 = G$ ist, muss auch $A_1 \cap A_k = A_k$ gelten. Die Ordnung auf der Menge der Blöcke wurde so definiert, dass der kleinste Block der Menge $\{\Delta_{\{1,2,k\}}^{aq} \mid a \in A_k\}$ genau derjenige ist, der den kleinsten Pfad enthält. Nach Lemma 18 bilden die Blöcke im Urbild von $\Delta_{\{1,k\}}^q$ unter dem G -Homomorphismus $\varphi_{(\{1,2,k\},\{1,k\})}$ eine Partition der Menge der Pfade dieses Blockes. Daher ist der Pfad $(A_1 p, \dots, A_k p)$ im kleinsten Block unter allen Urbildern von $\Delta_{\{1,k\}}^q$ unter der Abbildung $\varphi_{(\{1,2,k\},\{1,k\})}$ enthalten.

Der kleinste Block aus der Menge $\{\Delta_{\{1,2,k\}}^{aq} \mid a \in A_k\}$ ist nach Lemma 6 derjenige Block, dessen Pfade die kleinste zweite Komponente besitzen. Die Menge $\{A_2 a q \mid a \in A_k\}$ enthält alle Nebenklassen, die als zweiten Komponenten in einem Pfad eines Blockes aus der Menge $\{\Delta_{\{2,k\}}^{aq} \mid a \in A_k\}$ vorkommen. Wenn der Gruppenindex $(A_1 : A_2)$ klein ist, kann die Menge $\{\Delta_{\{2,k\}}^{aq} \mid a \in A_k\}$ einfach durchlaufen werden, um den kleinsten darin enthaltenen Block zu finden. Da der Pfad $(A_1 p, \dots, A_k p)$ in diesem Block enthalten sein muss, kann auf diese Weise auch die zweite Komponente des Pfades $(A_1 p, \dots, A_k p)$ bestimmt

werden.

Im Folgenden wird ein iteratives Verfahren beschrieben, mit dem die übrigen Komponenten des Pfades (A_1p, \dots, A_kp) in aufsteigender Reihenfolge bestimmt werden können. Bei der Bestimmung der jeweils nächsten Komponente werden zwei Fälle unterschieden, je nachdem ob die darauffolgende Untergruppe der Leiter eine Untergruppe der vorherigen ist oder nicht. Bei der Bestimmung der $i + 1$ -ten Komponente wird vorausgesetzt, dass im vorangehenden Iterationsschritt ein Element $g \in G$ bestimmt wurde, so dass (A_1p, \dots, A_kp) im Block $\Delta_{\{i,k\}}^g$ liegt. Bezeichnet $\Delta_{\{1,2,k\}}^{ag}$ den Block, der im ersten Schritt als kleinster Block im Urbild bestimmt wurde, so setze für den folgenden Iterationsschritt $g = ag$.

Fall 1: $A_i \geq A_{i+1}$

Es sei ein Element $g \in G$ gegeben, so dass (A_1p, \dots, A_kp) im Block $\Delta_{\{i,k\}}^g$ liegt. Unter Verwendung von Lemma 16 kann die Urbildmenge des Blockes $\Delta_{\{i,k\}}^g$ unter dem G -Homomorphismus $\varphi_{(\{i,i+1,k\}, \{i,k\})}$ berechnet werden. Dies ist die Menge $\{\Delta_{\{i,i+1,k\}}^{ag} \mid a \in A_i \cap A_k\}$. Die Gruppe $E_i = A_i \cap A_k$ braucht dabei für jede Untergruppenleiter nur ein einziges Mal bestimmt und gespeichert werden und kann danach aus dem Speicher abgerufen werden. Die Berechnung der Gruppe E_i kann mit Hilfe des Breadthfirst StroLL Algorithmus oder, wie in Kapitel 7.6.3 beschrieben, mit dem Breadthfirst StroLL oder dem Leiter-spiel Light Algorithmus erfolgen.

Die Ordnung auf der Menge der Blöcke wurde so definiert, dass der gesuchte kleinste Pfad im kleinsten Block der Urbildmenge $\{\Delta_{\{i,i+1,k\}}^{ag} \mid a \in E_i\}$ enthalten ist. Der kleinste Block der Menge $\{\Delta_{\{i,i+1,k\}}^{ag} \mid a \in E_i\}$ ist nach Lemma 6 derjenige Block, dessen Pfade die kleinste $i + 1$ -te Komponente besitzen. Ist der Gruppenindex $(A_i : A_{i+1})$ klein, so ist auch die Menge $\{A_{i+1}ag \mid a \in E_i\}$ klein und die Nebenklassen aus dieser Menge können leicht durchlaufen werden. Dabei kann die kleinste Nebenklasse $A_{i+1}ag$ aus dieser Menge ermittelt werden. Die $i + 1$ -te Komponente des Pfades (A_1p, \dots, A_kp) muss dann gleich $A_{i+1}ag$ sein und der gesuchte Pfad liegt im Block $\Delta_{\{i+1,k\}}^{ag}$.

Im folgenden Iterationsschritt wird vorausgesetzt, dass ein Element $h \in G$ be-

stimmt wurde, für das $(A_1p, \dots, A_kp) \in \Delta_{\{i+1,k\}}^h$ gilt. Bezeichnet $\Delta_{\{i+1,k\}}^{aq}$ den Block, der den Pfad (A_1p, \dots, A_kp) enthält, so setze für den folgenden Iterationsschritt $h = aq$.

Fall 2: $A_i \leq A_{i+1}$

Ist $A_i \leq A_{i+1}$ und ist die i -te Komponente des Pfades (A_1p, \dots, A_kp) bereits bekannt, so kann die darauf folgende Pfadkomponente leicht bestimmt werden. Es bezeichne $g \in G$ wieder ein Element, für das $(A_1p, \dots, A_kp) \in \Delta_{\{i,k\}}^g$ gilt. Aufgrund der Definition der Erweiterungsfunktion Υ_i in Kapitel 6.1 muss die $i+1$ -te Komponente des Pfades gleich $A_{i+1}g$ sein. Denn das Bild der i -ten Komponente $A_i g$ unter der Erweiterungsfunktion Υ_i ist die einelementige Menge $\{A_{i+1}g\}$. Da die Definition eines Pfades vorschreibt, dass die $i+1$ -te Komponente in dieser einelementigen Menge enthalten sein muss, kann die $i+1$ -te Komponente sofort festgelegt werden. Für den darauf folgenden Iterationsschritt wird wieder ein Element $h \in G$ benötigt, für das $(A_1p, \dots, A_kp) \in \Delta_{\{i+1,k\}}^h$ gilt. Für das Element g ist diese Voraussetzung bereits erfüllt.

Auf diese Weise kann der gesuchte Pfad innerhalb von $k-2$ Schritten gefunden werden. Sind die zuvor angesprochenen Gruppenindizes der Leitergruppen klein, so kann jeder der einzelnen Schritte mit geringem Aufwand durchgeführt werden.

Kapitel 8

Analyse des Laufzeitverhaltens

In diesem Kapitel werden die Eigenschaften der drei in dieser Arbeit neu vorgestellten Leiterspiel-Algorithmen anhand von konkreten Rechenbeispielen untersucht.

In Kapitel 8.2 wird jeder der drei Algorithmen auf ein konkretes Kanonisierungsproblems angesetzt. Daraufhin werden die Algorithmen in Bezug auf ihre Laufzeit, ihren Bedarf an Hauptspeicher und die Anzahl der berechneten Nebenklassen zur Lösung des Kanonisierungsproblems verglichen.

In Kapitel 8.3 wird der Leiterspiel Light Algorithmus mit dem ursprünglichen Leiterspiel-Algorithmus von Schmalz verglichen. Beide Algorithmen werden unabhängig voneinander zur Konstruktion aller schlichten Graphen auf bis zu 9 Knoten eingesetzt. Dabei werden die Algorithmen in Bezug auf ihre Laufzeit und ihren Bedarf an Hauptspeicher miteinander verglichen.

An dieser Stelle soll auf die Veröffentlichung von M. De Santo, P. Foggia, C. Sansone und M. Vento hingewiesen werden, in der die Leistungsfähigkeit verschiedener Graphenkanonisierer [SFSV03] miteinander verglichen werden. Die in dieser Arbeit beschriebenen Algorithmen unterscheiden sich in vielerlei Hinsicht von den dort verwendeten Algorithmen. Weiterhin haben die in dieser Arbeit beschriebenen Algorithmen noch nicht den jahrelangen Optimierungsprozess, wie zum Beispiel der aus dem Graphenkanonisierer *nauty* hervorgegangene Graphenkanonisierer *traces* von B. D. McKay, durchlaufen [MP14]. Ein Vergleich verschiedener Kanonisierungsalgorithmen, der auch die Algorithmen dieser Arbeit miteinbezieht, wäre für die Zukunft wünschenswert.

8.1 Kurzbeschreibung der Algorithmen

8.1.1 Leiterspiel-Algorithmus von Schmalz

Der ursprüngliche Leiterspiel-Algorithmus von Schmalz wurde in Kapitel 4 vorgestellt. Dieser Algorithmus zeichnet sich insbesondere dadurch aus, dass mit diesem Algorithmus auch schwierigste Isomorphieprobleme gelöst werden können.

Mit dem Leiterspiel-Algorithmus von Schmalz kann zu gegebenen Gruppen G , A und B , mit $A \leq G$ und $B \leq G$, die Menge aller Doppelnebenklassen $A \backslash G / B = \{AgB \mid g \in G\}$ bestimmt werden. Genauer gesagt wird zu jeder Doppelnebenklasse AgB eine Rechtsnebenklasse Agb mit $b \in B$ bestimmt, die die entsprechende Doppelnebenklasse repräsentiert. Der Repräsentant einer Doppelnebenklasse AgB kann jedoch nicht einzeln bestimmt werden, sondern nur zusammen mit allen anderen Repräsentanten von Doppelnebenklassen aus der Menge $A \backslash G / B$.

8.1.2 Leiterspiel Light

Der Leiterspiel Light Algorithmus wurde in Kapitel 7.5 beschrieben. Es seien G , A und B drei Gruppen mit $A \leq G$ und $B \leq G$ und g ein Element aus der Gruppe G . Dieser Algorithmus ermöglicht es, zu jeder Doppelnebenklasse AgB ein Element $b \in B$ und eine Rechtsnebenklasse Agb zu bestimmen, die diese Doppelnebenklasse repräsentiert.

Um diesen Algorithmus einsetzen zu können, müssen mehrere Voraussetzungen erfüllt sein. Es wird eine Untergruppenleiter (A_1, \dots, A_n) von G nach A benötigt, die nach den in Kapitel 6.5 beschriebenen Kriterien eine starke Untergruppenleiter ist. Weiterhin wird für alle $i \leq n$ eine Totalordnung auf der Menge $A_i \backslash G$ der Rechtsnebenklassen der Gruppe A_i benötigt, die den in Kapitel 6.1.2 genannten Kriterien genügt.

In vielen Fällen ist der Leiterspiel Light Algorithmus ein schneller, einfach zu implementierender Algorithmus, der zur Lösung von kleineren Nebenklassenisomorphieproblemen verwendet werden kann. Dieser Algorithmus eignet sich in Kombination mit dem im Anschluss beschriebenen Depthfirst StroLL Algorithmus auch zur Lösung von großen Nebenklassenisomorphieproblemen.

8.1.3 Depthfirst StroLL

Der Depthfirst StroLL Algorithmus wurde in Kapitel 7.4.2 bis 7.4.5 beschrieben. Genau wie der Leiterspiel Light Algorithmus ermöglicht es dieser Algorithmus, zu jeder Doppelnebenklasse AgB ein Element $b \in B$ und eine Rechtsnebenklasse Agb zu bestimmen, die diese Doppelnebenklasse repräsentiert.

Der Depthfirst StroLL benötigt dieselben Voraussetzungen wie das Leiterspiel Light, eignet sich im Gegensatz zu diesem jedoch auch zur Lösung von großen Nebenklassenisomorphieproblemen. Dieser Algorithmus ermöglicht es, große Isomorphieprobleme in mehrere kleinere zu zerlegen, die dann anschließend rekursiv gelöst werden können. In der Praxis hat sich herausgestellt, dass das Leiterspiel Light wesentlich effizienter ist bei der Lösung von kleineren Isomorphieproblemen. Daher empfiehlt es sich, kleinere Isomorphieprobleme, die beim Depthfirst StroLL in den Rekursionsschritten auftauchen, immer mit Hilfe des Leiterspiel Light Algorithmus zu lösen.

8.1.4 Breadthfirst StroLL

Der Breadthfirst StroLL Algorithmus wurde in Kapitel 7.3 beschrieben. Dieser Algorithmus ermöglicht es, genau wie der Depthfirst StroLL und das Leiterspiel Light, zu einer einzelnen gegebenen Doppelnebenklasse eine repräsentierende Rechtsnebenklasse zu berechnen.

Die Voraussetzungen für diesen Algorithmus sind identisch zu denen des Depthfirst StroLL und des Leiterspiel Light Algorithmus. Im Gegensatz zu diesen Algorithmen benötigt der Breadthfirst StroLL Algorithmus jedoch deutlich mehr Arbeitsspeicher. Unter den drei Algorithmen Leiterspiel Light, Depthfirst StroLL und Breadthfirst StroLL ist dieser Algorithmus dem ursprünglichen Algorithmus von Schmalz am ähnlichsten.

8.2 Vergleich der Eigenschaften

In diesem Kapitel werden die drei neuen Algorithmen dieser Arbeit miteinander verglichen. Anhand eines Beispiels wird gezeigt, wie unterschiedlich die drei Algorithmen in Bezug auf ihre Laufzeit, ihren Bedarf an Arbeitsspeicher und die Anzahl der durchlaufenen Nebenklassen sind. Dieser Vergleich dient jedoch

ausdrücklich nicht dazu, die Algorithmen hinsichtlich ihrer Leistungsfähigkeit zu vergleichen. Die Leistungsfähigkeit des neuen Ansatzes wird in Kapitel 8.3 unter Beweis gestellt.

Der Depthfirst StroLL arbeitet rekursiv, große Problemstellungen werden in kleinere Probleme zerlegt, die anschließend wieder mit einem Kanonisierungsalgorithmus gelöst werden müssen. Wird der Depthfirst StroLL auch wieder zur Lösung dieser kleineren Probleme eingesetzt, so muss insgesamt eine sehr große Zahl sehr kleiner Kanonizitätsprobleme mit diesem Algorithmus gelöst werden. Da jedoch kleine Isomorphieprobleme mit dem Depthfirst StroLL nur sehr langsam gelöst werden können, führt dies zu einem insgesamt langsamen Algorithmus.

In diesem Kapitel sollen jedoch die Unterschiede der drei Algorithmen im Vordergrund stehen, daher wurde darauf verzichtet, die verschiedenen Algorithmen miteinander zu kombinieren. In der Praxis hingegen sollten beim Depthfirst StroLL die kleinen Isomorphieprobleme aus den Rekursionsschritten immer mit einem anderen Algorithmus wie zum Beispiel dem Leiterspiel Light gelöst werden.

8.2.1 Der Beispielgraph

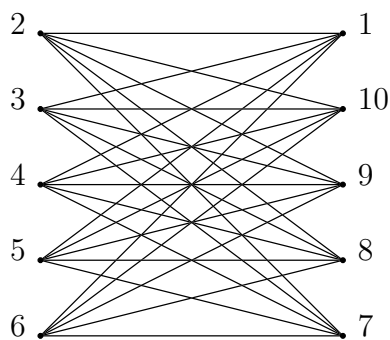


Abbildung 8.1: Graph aus 10 Knoten und 25 Kanten

Um die drei Algorithmen vergleichen zu können, wird jedem der drei die Aufgabe gestellt, den Graphen aus Abbildung 8.1 zu kanonisieren. Nach dem Split Lemma aus Kapitel 3.5 kann der Graph aus Abbildung 8.1 auf eine Rechtsnebenklasse abgebildet werden. Dieser Schritt wurde bereits in dem Beispiel

auf Seite 24 beschrieben. Dort wurde auch beschrieben, wie die Gruppen G , A und B zur Doppelnebenklassenmenge $A \backslash G / B$, der Bildmenge der Split Lemma Abbildung, bestimmt werden.

Wurde der Graph unter der Abbildung des Split Lemmas auf eine Nebenklasse Ag abgebildet, so kann mit jedem der drei Algorithmen überprüft werden, ob Ag die kanonische Nebenklasse in ihrer Bahn $Ag^B = \{Agb | b \in B\}$ ist. Bei jedem der drei Testfälle wird dasselbe Kanonizitätsprädikat zugrunde gelegt, das in Kapitel 8.2.2 erläutert wird.

In einem weiteren Schritt wird dann das Urbild der kanonischen Nebenklasse unter der Abbildung des G -Isomorphismus des Split Lemmas berechnet. Das Kanonizitätsprädikat auf der Menge der Graphen kann so gewählt werden, dass das Urbild jeder kanonischen Nebenklasse auch der kanonische Repräsentant der Bahn des betreffenden Graphen ist. Auf diese Weise kann mit jedem der drei Algorithmen die Kanonizität des Graphen in Abbildung 8.1 überprüft werden.

8.2.2 Voraussetzungen

Der vollständige Graph auf 10 Knoten besitzt genau 45 Kanten. Um den Graphen aus Abbildung 8.1 auf eine Doppelnebenklasse abzubilden, wird die Gruppe G gleich der Symmetrischen Gruppe S_{45} gewählt, entsprechend dem Beispiel auf Seite 24.

Bei allen drei Algorithmen wurde folgende starke Untergruppenleiter zugrunde gelegt:

$$\begin{aligned} A_1 &= G \\ A_{2k} &= S_{(\{1, \dots, k\}, \{k+1, \dots, 45\})} \quad \forall 1 \leq k \leq 45 \\ A_{2k+1} &= S_{(\{1, \dots, k\}, \{k+1\}, \{k+2, \dots, 45\})} \quad \forall 1 \leq k < 45 \end{aligned}$$

Auf der Menge der Gruppenelemente von G ist durch die lexikographische Ordnung eine Totalordnung gegeben. Entsprechend dieser Ordnung kann für alle Untergruppe U von G eine Totalordnung auf der Menge $U \backslash G$ festgelegt werden: Zu je zwei gegebenen Elementen $g_1, g_2 \in G$ soll $Ug_1 \preccurlyeq Ug_2$ gelten, genau dann, wenn das kleinste Element g'_1 aus der Menge $\{ug_1 | u \in U\}$ kleiner ist als das kleinste Element der Menge $\{ug_2 | u \in U\}$. Das Kanonizitätsprädikat wurde so gewählt, dass für alle Elemente $g \in G$ und Untergruppen U

und B von G genau die nach dieser Ordnung kleinste Nebenklasse in der Bahn $Ug^B = \{Ugb \mid b \in B\}$ als kanonisch gelten soll.

Sämtliche Berechnungen wurden auf einem einzelnen Prozessor des Linux-Clusters der Universität Bayreuth durchgeführt. Die verwendete CPU ist ein Intel Xeon Prozessor vom Typ E5520, ausgestattet mit 24 Gigabyte 1066 MHz DDR3 RAM Modulen.

8.2.3 Ergebnisse

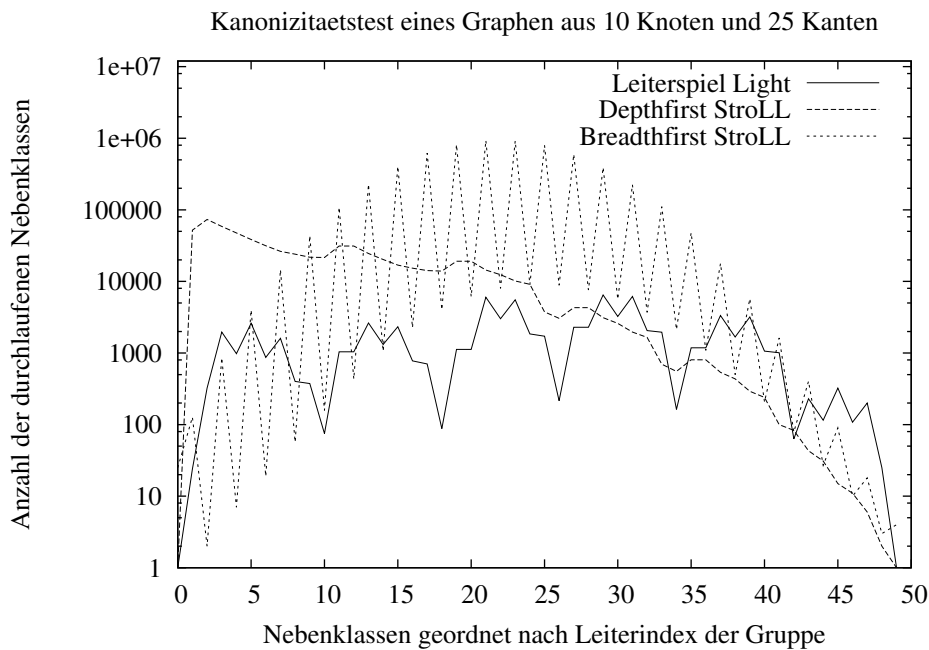


Abbildung 8.2: Anzahl aller durchlaufenen Nebenklassen

In Abbildung 8.2 ist für jeden der drei Algorithmen die Anzahl der durchlaufenen Rechtsnebenklassen dargestellt. Es bezeichne (A_1, \dots, A_{90}) die starke Untergruppenleiter, die bei den Berechnungen der Algorithmen verwendet wird. Zu jeder der durchlaufenen Rechtsnebenklassen existiert ein $g \in G$ und eine Gruppe E , so dass die Rechtsnebenklasse als Eg geschrieben werden kann. Zu dieser Gruppe E existieren wiederum zwei Indizes $i, k \in \{1, \dots, 50\}$ mit $i < k$, so dass $E = A_i \cap A_k$ ist. Für jeden der Algorithmen wurden die während des Kanonizitätstests durchlaufenen Nebenklassen gezählt und entsprechend dem kleineren der beiden Indizes aufgeschlüsselt. Die Anzahl der durchlaufenen

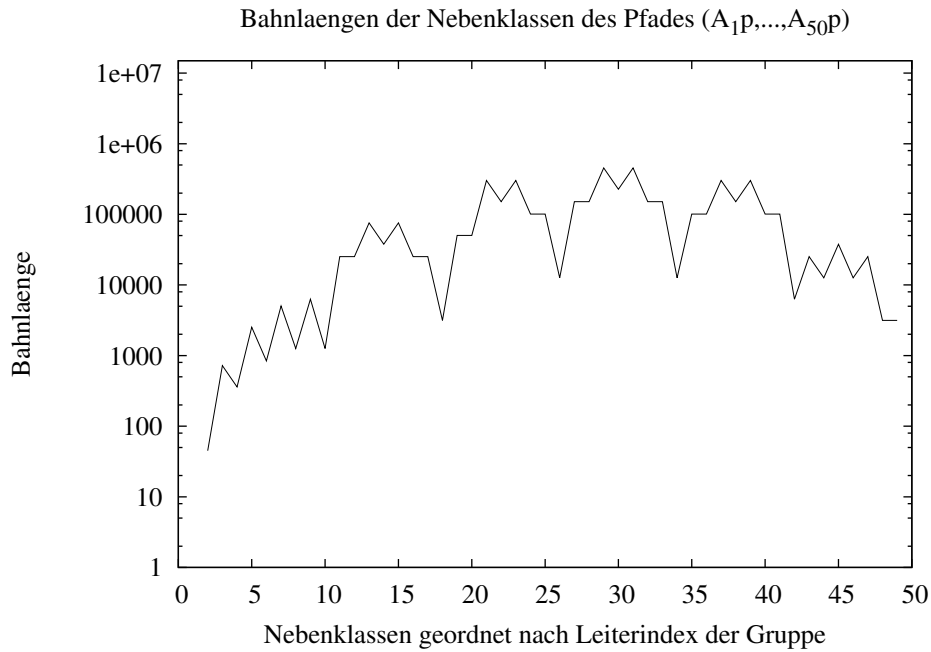


Abbildung 8.3: Bahn­längen aller Nebenklassen $A_i p$ unter der Gruppe B

nen Nebenklassen wurde dann entsprechend diesem Index auf der x-Achse in Abbildung 8.2 dargestellt. Auf diese Weise entstehen drei sehr unterschiedliche Nebenklassen Verteilungen, die wiederum Aufschluss über die Arbeitsweise des jeweiligen Algorithmus geben.

Beim Leiterspiel Light Algorithmus wird zuerst ein Pfad $\rho = (A_1 p, \dots, A_{50} p)$ berechnet, der kleinste Pfad zu der gegebenen Nebenkasse $A_{50} p$, für die überprüft werden soll, ob diese kanonisch ist. Für jeden Index $i \leq 50$ entspricht die Anzahl der durchlaufenen Nebenklassen, bis auf eine geringe, programmiertechnisch bedingte Abweichung, genau der Größe der Schnittmenge der beiden Mengen $\{A_i p a \mid a \in A_{50}\}$ und $\{A_i p b \mid b \in B\}$. Die Mächtigkeit der Menge $\{A_i p b \mid b \in B\}$ ist daher entscheidend für die Anzahl der zu durchlaufenden Nebenklassen. Diese kann wie folgt berechnet werden:

$$|\{A_i p b \mid b \in B\}| = \frac{|B|}{|B \cap p^{-1} A_i p|}$$

Der Nenner im Bruch auf der rechten Seite ist identisch mit der Größe des Stabilisators der Nebenkasse $A_i p$ in B . Daher übt die Größe dieses Stabilisators einen entscheidenden Einfluss auf die Anzahl der zu durchlaufenden Nebenklassen aus. In Abbildung 8.3 ist für alle Indizes i von 1 bis 50 die Mächtigkeit

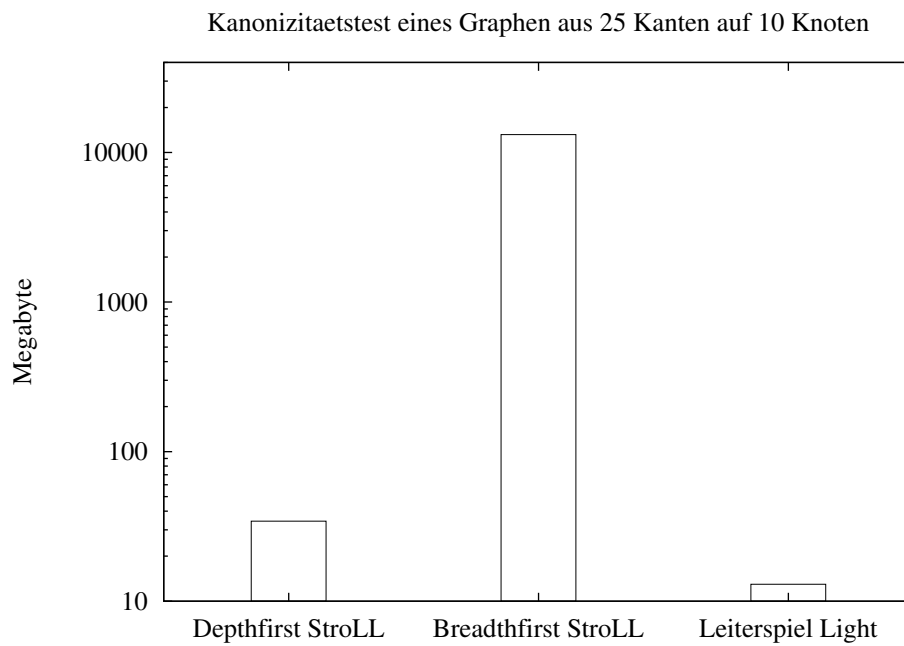


Abbildung 8.4: Bedarf an Arbeitsspeicher der drei neuen Algorithmen

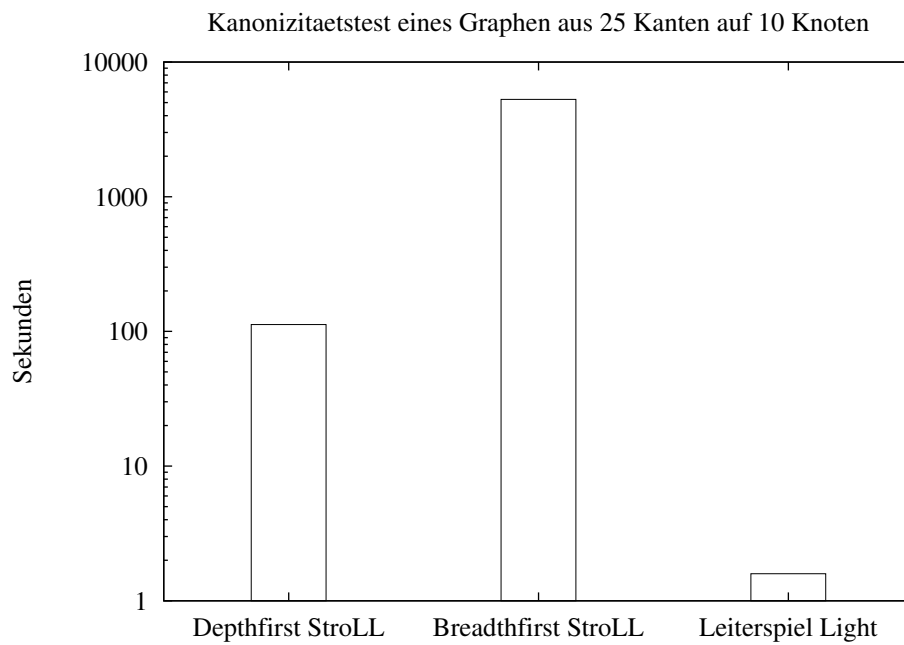


Abbildung 8.5: Bedarf an Rechenzeit der drei neuen Algorithmen

der Menge $\{A_i pb \mid b \in B\}$ dargestellt. Die Korrelation zwischen der Abbildung 8.3 und der Nebenklassenverteilung des Leiterspiel Light Algorithmus in Abbildung 8.2 ist leicht zu erkennen.

Die rekursive Lösungsstrategie des Depthfirst StroLL führt dazu, dass sehr viele Nebenklassen mit kleinem Leiterindex durchlaufen werden. Im Vergleich zu den beiden anderen Algorithmen werden die Nebenklassen mit großem Leiterindex etwas seltener durchlaufen. In der Verteilung der Nebenklassen des Depthfirst StroLL in Abbildung 8.2 ist auch ein geringer Einfluss der Bahnlängen aus Abbildung 8.3 erkennbar. Dieser ist jedoch viel kleiner als beim Leiterspiel Light Algorithmus.

Die Sägezähne, die in der Nebenklassenverteilung des Breadthfirst StroLL in Abbildung 8.2 zu sehen sind, können auf die unterschiedlichen Größen der Gruppen der Leiter zurückgeführt werden. Da sich bei der gegebenen Leiter die Splitting- und Fusingschritte abwechseln, entsteht dieses Sägezahnmuster, das auch in den Darstellungen der beiden anderen Algorithmen zu erkennen ist.

8.3 Vergleich der Leistungsfähigkeit

In diesem Kapitel werden die Eigenschaften des Leiterspiel-Algorithmus von Schmalz mit dem Leiterspiel Light Algorithmus verglichen. Die beiden Algorithmen werden hinsichtlich ihres Bedarfs an Rechenzeit und ihres Bedarfs an Arbeitsspeicher untersucht.

Bezeichnet G eine Gruppe und A und B zwei Untergruppen von G , so kann der Algorithmus von Schmalz zur Konstruktion aller kanonischen Repräsentanten von Doppelnebenklassen $A \backslash G / B$ eingesetzt werden. Es ist mit diesem Algorithmus jedoch nicht möglich, den kanonischen Repräsentanten einer einzelnen Doppelnebenklasse zu bestimmen, ohne die übrigen kanonischen Repräsentanten zu berechnen.

Der Leiterspiel Light Algorithmus hingegen ist ausschließlich dazu in der Lage, zu einer einzelnen Doppelnebenklasse den kanonischen Repräsentanten zu bestimmen. Dieser Algorithmus kann jedoch in Kombination mit der Ordnungstreuen Erzeugung zur Berechnung der kanonischen Repräsentanten aller Doppelnebenklassen $A \backslash G / B$ eingesetzt werden.

Für den Vergleich wurde daher ein Rechenbeispiel gewählt, bei dem mit dem Algorithmus von Schmalz alle kanonischen Repräsentanten von Doppelnebenklassen $A \backslash G / B$ berechnet werden und das Leiterspiel Light in Kombination mit der Ordnungstreu- Erzeugung zur Konstruktion derselben kanonischen Repräsentanten eingesetzt wird.

Nach dem Split Lemma aus Kapitel 3.5 kann die Aufgabe, alle schlichten Graphen auf n Knoten bis auf Isomorphie zu konstruieren, auf ein Doppelnebenklassenproblem abgebildet werden. In dem Beispiel zum Split Lemma auf Seite 24 wurde bereits beschrieben, wie ein Graph auf eine Nebenklasse abgebildet werden kann. Dort wurde auch beschrieben, wie die Gruppen G , A und B zur Doppelnebenklassenmenge $A \backslash G / B$, der Bildmenge der Split Lemma Abbildung, bestimmt werden. Jede Doppelnebenklasse entspricht einem Graphenisomorphietypen, daher muss zur Konstruktion aller schlichten Graphen auf n Knoten außer der Berechnung der Doppelnebenklassen nahezu kein zusätzlicher Aufwand betrieben werden. Daher eignet sich dieses Beispiel besonders gut für einen Vergleich der unterschiedlichen Algorithmen.

An dieser Stelle soll betont werden, dass die im Beispiel konstruierten Graphenmengen längst bekannt sind und dieses Problem nur aufgrund seiner Anschaulichkeit ausgewählt wurde.

8.3.1 Voraussetzungen

Zum Vergleich der beiden Algorithmen wurden alle schlichten Graphen auf 4, 5, 6, 7, 8 und 9 Knoten generiert. Der vollständige Graph auf n Knoten besitzt genau $m = n \cdot (n - 1) / 2$ Kanten. Zur Konstruktion aller Graphen auf n Knoten wurde, entsprechend dem Beispiel auf Seite 24, die Gruppe G gleich der Symmetrischen Gruppe S_m gewählt.

Für die Konstruktion der Graphen auf n Knoten wurde folgende starke Untergruppenleiter verwendet:

$$\begin{aligned} A_1 &= G \\ A_{2k} &= S_{(\{1, \dots, k\}, \{k+1, \dots, m\})} \quad \forall 1 \leq k \leq m \\ A_{2k+1} &= S_{(\{1, \dots, k\}, \{k+1\}, \{k+2, \dots, m\})} \quad \forall 1 \leq k < m \end{aligned}$$

Den Berechnungen wurde dasselbe Kanonizitätsprädikat zugrunde gelegt, das bereits in Kapitel 8.2.2 beschrieben wurde. Sämtliche Berechnungen wurden

auf einem einzelnen Prozessor des Linux-Clusters der Universität Bayreuth durchgeführt. Die verwendete CPU ist ein Intel Xeon Prozessor vom Typ E5520, ausgestattet mit 24 Gigabyte 1066 MHz DDR3 RAM Modulen.

8.3.2 Ergebnisse

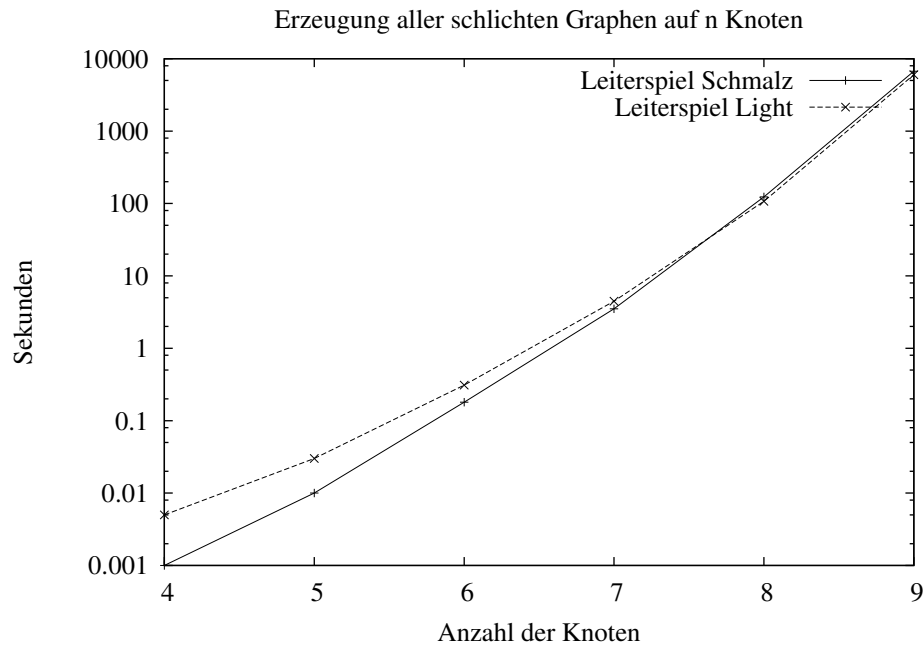


Abbildung 8.6: Vergleich der Laufzeit

Wie in Abbildung 8.6 zu sehen ist, benötigen der Algorithmus von Schmalz und das Leiterspiel Light nahezu dieselbe Rechenzeit zur Berechnung aller schlichten Graphen auf bis zu neun Knoten. In Abbildung 8.7 ist die Laufzeit jedes der beiden Algorithmen geteilt durch die Anzahl der konstruierten Graphen dargestellt. Dies ermöglicht es, die Unterschiede in der Laufzeit der beiden Algorithmen noch etwas genauer darzustellen.

Der Vergleich des Leiterspiels von Schmalz mit dem Leiterspiel Light zeigt insbesondere, dass das Leiterspiel von Schmalz einen um mehrere Größenordnungen größeren Bedarf an Arbeitsspeicher hat. Die Graphen auf 10 Knoten konnten mit dem Leiterspiel von Schmalz nicht mehr berechnet werden, da schon bei der Konstruktion aller einfachen Graphen auf 9 Knoten 34 Gigabyte Speicher benötigt werden. Wie unterschiedlich der Speicherbedarf der beiden

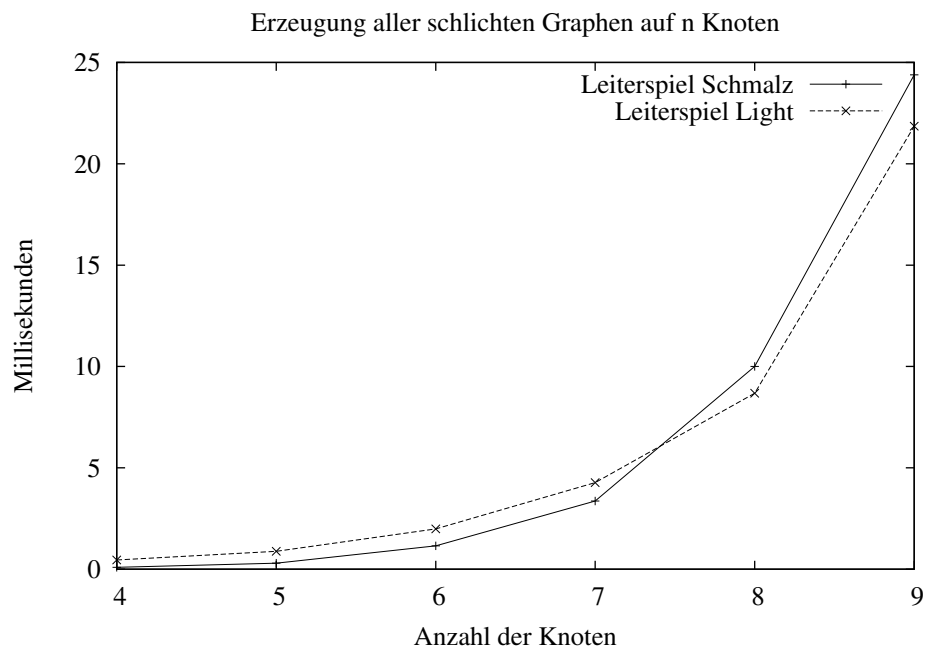


Abbildung 8.7: Vergleich der Laufzeit pro Graph

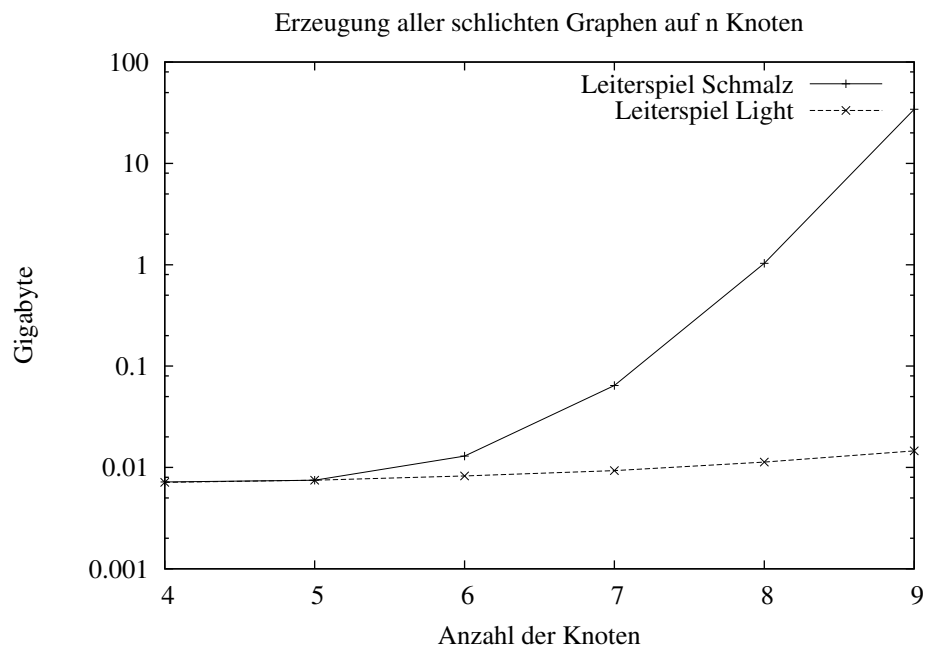


Abbildung 8.8: Bedarf an Arbeitsspeicher

Algorithmen ist, kann sehr deutlich anhand der Abbildung 8.8 gezeigt werden. Dabei ist insbesondere zu beachten, dass die y-Achse der Graphik in Abbildung 8.8 eine logarithmische Skalierung besitzt.

Der Leiterspiel Light Algorithmus konstruiert die gesuchten Graphen in Tiefensuche unter Verwendung der Ordnungstreuenerzeugung. Daher müssen immer nur sehr wenige kanonische Nebenklassen und Stabilisatoren gleichzeitig im Speicher gehalten werden. Beim Leiterspiel von Schmalz hingegen werden alle kanonischen Nebenklassen zusammen mit ihren Stabilisatoren gleichzeitig im Speicher gehalten. Aufgrund der hohen Anzahl von Nebenklassen, die zur Konstruktion der Graphen benötigt werden, ergibt sich dieser enorme Speicherbedarf. Wird beim Algorithmus von Schmalz der Speicherbedarf durch die Anzahl der konstruierten Nebenklassen geteilt, so ergibt sich das Verhältnis, das in Abbildung 8.9 dargestellt ist.

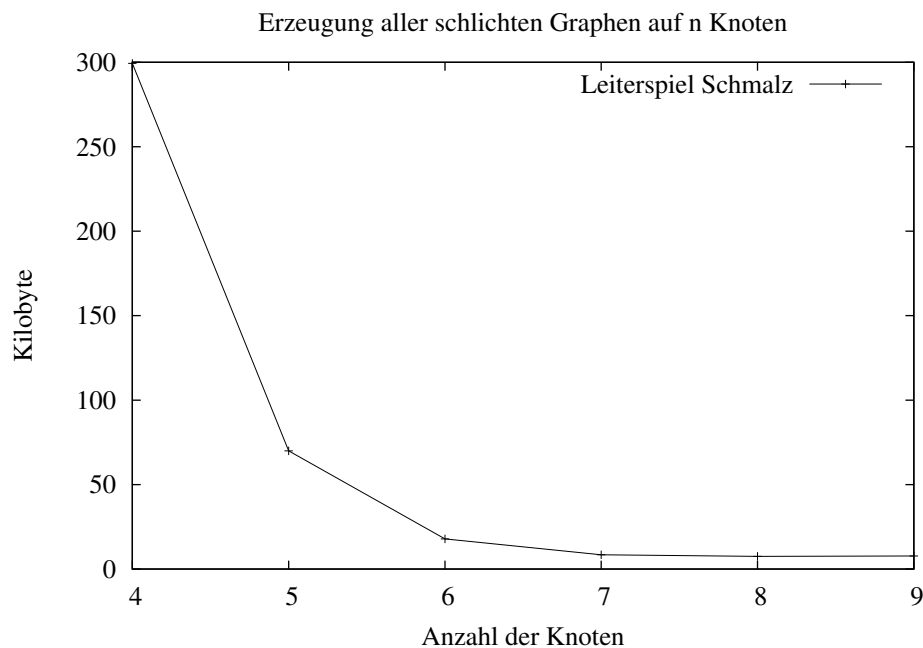


Abbildung 8.9: Durchschnittlicher Speicherbedarf pro Nebenklasse

8.3.3 Ergebnis des Vergleichs

Die Einsatzmöglichkeiten des ursprünglichen Leiterspiel-Algorithmus werden vor allem durch die Größe des zur Verfügung stehenden Hauptspeichers be-

#Knoten	CPU-Zeit	#Nebenklassen	#Graphen	Hauptspeicher
4	0.00 s	24	11	7.184 MByte
5	0.01 s	107	34	7.488 MByte
6	0.18 s	727	156	12.96 MByte
7	3.51 s	7607	1044	64.33 MByte
8	123.51 s	137465	12346	1.032 GByte
9	6698.10 s	4422499	274668	34.18 GByte

Abbildung 8.10: Leiterspiel nach Schmalz, Ergebnisse tabellarisch

#Knoten	CPU-Zeit	#Nebenklassen	#Graphen	Hauptspeicher
4	0.00 s	296	11	7.120 MByte
5	0.03 s	2878	34	7.488 MByte
6	0.31 s	29750	156	8.240 MByte
7	4.46 s	365344	1044	9.296 MByte
8	107.18 s	7500593	12346	11.296 MByte
9	6002.34 s	289138111	274668	14.560 MByte

Abbildung 8.11: Leiterspiel Light, Ergebnisse tabellarisch

schränkt. Zu jeder Doppelnebenklasse AgB wird eine Nebenklasse Agb mit $b \in B$ bestimmt, die als kanonischer Repräsentant dieser Doppelnebenklasse fungiert. Und zu jeder dieser kanonischen Nebenklassen Agb muss das Bild dieser Nebenklasse unter der fusionierenden Abbildung und der Stabilisator $B_{Agb} = B \cap b^{-1}g^{-1}Agb$ dieser Nebenklasse unter der Operation der Gruppe B gespeichert werden. Daher wächst beim ursprünglichen Leiterspiel-Algorithmus mit jeder konstruierten Doppelnebenklasse der Bedarf an Hauptspeicher. Werden die Doppelnebenklassen hingegen mit dem Leiterspiel Light Algorithmus in ordnungstreuer Erzeugung konstruiert, so wird dazu nur eine sehr geringe Menge an Hauptspeicher benötigt. Bei der Konstruktion von schlichten Graphen bis 9 Knoten ist der Leiterspiel Light Algorithmus genauso so schnell wie der ursprüngliche Leiterspiel-Algorithmus von Schmalz.

Kapitel 9

Ausblick

Homomorphismen von Gruppenoperationen werden nicht nur zur Lösung von Isomorphieproblemen verwendet. Viele Beweise gruppentheoretischer Aussagen und viele Algorithmen der Gruppentheorie beruhen auf Homomorphismen von Gruppenoperationen. Untergruppenleitern können als Beschreibung einer Kette von Homomorphismen von Gruppenoperationen auf Nebenklassenmengen betrachtet werden. Die starken Untergruppenleitern, die in dieser Arbeit zum ersten Mal beschrieben wurden, sind ein neues Werkzeug in der Gruppentheorie, das sowohl für Beweise in der Gruppentheorie als auch zur Beschreibung von Algorithmen verwendet werden kann.

Der Unterschied zwischen den bisher verwendeten und den starken Untergruppenleitern besteht in der Transitivität der Gruppenoperation auf der Menge der Pfade. Dieses Konzept kann ohne Weiteres auch auf andere Gruppenoperationen als die Rechtsmultiplikation und auch auf andere Objektmengen als Nebenklassen angewendet werden. Diese Verallgemeinerung ermöglicht weitere Abwandlungen des Leiterspiel-Algorithmus. Insgesamt sind die in dieser Arbeit beschriebenen Algorithmen als Prototypen zu betrachten, um die Tragfähigkeit dieser neuen Konzepte zu zeigen. Indem die beschriebenen Algorithmen miteinander kombiniert wurden, die Ordnungstreue Erzeugung auch auf den Breadthfirst StroLL Algorithmus angewendet wurde und das Konzept der starken Leitern ausgeweitet wurde, sind inzwischen weitere Algorithmen hinzugekommen, die alle auf dem Konzept der starken Leitern beruhen.

Weiterer Forschungsbedarf besteht daher darin, Varianten der Algorithmen zu implementieren und hinsichtlich ihrer Effizienz zu untersuchen. Neben der Effizienz sollte insbesondere auch die Parallelisierung dieser Algorithmen untersucht werden. In Kapitel 7.4.5 wurde zu allen wesentlichen Teilen des Algorithmus auch der Pseudocode angegeben. Dies ist als Hilfestellung an die Leser gedacht und soll diese dazu motivieren, die beschriebenen Algorithmen zu implementieren und anzuwenden. Anhand der Kürze des Pseudocodes soll auch verdeutlicht werden, dass eine Implementierung der Algorithmen trotz der Schwierigkeit der Thematik eine durchaus überschaubare Aufgabe bleibt.

Invarianten können als spezielle Homomorphismen von Gruppenoperationen betrachtet werden. Bei vielen der erfolgreichsten Algorithmen zur Lösung von Isomorphieproblemen werden auf geschickte Weise Invarianten berechnet, mit Hilfe derer sich die Effizienz der Algorithmen mitunter drastisch verbessern lässt. Dieser Aspekt wurde bei den Algorithmen dieser Arbeit weitgehend außer Acht gelassen, um den Fokus auf das Wesentliche zu beschränken. Auch hier besteht weiterer Forschungsbedarf, um herauszufinden, welche Invarianten sich zur Unterscheidung von Doppelnebenklassen eignen. In diesem Zusammenhang soll auch insbesondere auf die Möglichkeit hingewiesen werden, dynamische Leitern zur Lösung von Isomorphieproblemen einzusetzen. Mit dynamischen Leitern sind Leitern gemeint, bei denen die Eigenschaften der betrachteten Doppelnebenklasse einen Einfluss darauf haben, welcher Homomorphismus von Gruppenoperationen im darauffolgend Leiterschritt angewendet werden soll.

Literaturverzeichnis

- [And03] ANDERSON, Amy C.: The Process of Structure-Based Drug Design. In: *Chemistry & Biology* 10 (2003), Nr. 9, S. 787–797
- [BM97] BOHACEK, Regine S. ; McMARTIN, Colin: Modern computational chemistry and drug discovery: structure generating programs. In: *Current Opinion in Chemical Biology* 1 (1997), Nr. 2, S. 157–161
- [CDHW73] CANNON, John J. ; DIMINO, Lucien A. ; HAVAS, George ; WATSON, Jane M.: Implementation and Analysis of the Todd-Coxeter Algorithm. In: *Mathematics of Computation* 27 (1973), Nr. 123, S. 463–490
- [CF94] COOPERMAN, Gene ; FINKELSTEIN, Larry: A random base change algorithm for permutation groups. In: *Journal of Symbolic Computation* 17 (1994), Nr. 6, S. 513–528
- [CFM96] CLARK, David E. ; FIRTH, Mike A. ; MURRAY, Christopher W.: MOLMAKER: De Novo Generation of 3D Databases for Use in Drug Design. In: *Journal of Chemical Information and Computer Sciences* 36 (1996), Nr. 1, S. 137–145
- [CFP89] COOPERMAN, Gene ; FINKELSTEIN, Larry ; PURDOM, Paul W.: Fast group membership using a strong generating test for permutation groups. In: *Computers and mathematics, Proc. Conf., Cambridge/Mass.* Springer-Verlag New York, Inc., 1989, S. 27–36
- [Far78] FARADŽEV, I.A.: Constructive enumeration of combinatorial objects. In: *Problèmes combinatoires et théorie des graphes, No.260.* Colloq. int. CNRS 1976, Orsay, 1978, S. 131–135

- [GLM97] GRÜNER, Thomas ; LAUE, Reinhard ; MERINGER, Markus: Algorithms for group actions applied to graph generation. In: *Groups and computation II. Workshop on groups and computation, June 7–10, 1995, New Brunswick, NJ, USA*. Providence, RI: American Mathematical Society, 1997, S. 113–122
- [Hal59] HALL, Marshall: *The theory of groups*. New York: The Macmillan Company. XIII., 1959
- [HR48] HALL, Marshall ; RADO, Tibor: On Schreier Systems in Free Groups. In: *Transactions of the American Mathematical Society* 64 (1948), Nr. 2, S. 386–408
- [Jer86] JERRUM, Mark: A compact representation for permutation groups. In: *Journal of Algorithms* 7 (1986), Nr. 1, S. 60–78
- [KB70] KNUTH, Donald E. ; BENDIX, Peter B.: Simple word problems in universal algebras. In: *Comput. Probl. abstract Algebra, Proc. Conf. Oxford 1967*, 1970, S. 263–297
- [KK09] KIERMAIER, Michael ; KOCH, Matthias: New complete 2-arcs in the uniform projective Hjelmslev planes over chain rings of order 25. In: *Proceedings of the Sixth International Workshop on Optimal Codes and Related Topics* (2009), S. 106–113
- [KKK11] KIERMAIER, Michael ; KOCH, Matthias ; KURZ, Sascha: 2-arcs of maximal size in the affine and the projective Hjelmslev plane over \mathbb{Z}_{25} . In: *Advances in Mathematics of Communications* 5 (2011), Nr. 2, S. 287–301
- [Kur06] KURZ, Sascha: *Konstruktion und Eigenschaften ganzzahliger Punktmengen*, Bayreuth. Math. Schr. 76. Universität Bayreuth, Diss., 2006
- [Lau82] LAUE, Reinhard: *Zur Konstruktion und Klassifikation endlicher auflösbarer Gruppen*. Bayreuth. Math. Schr. 9. Universität Bayreuth, 1982

- [Lau93] LAUE, Reinhard: Construction of combinatorial objects: A tutorial. In: *Bayreuth. Math. Schr.* 43, Universität Bayreuth, 1993, S. 53–96
- [McK81] MCKAY, Brendan D.: *Practical graph isomorphism*. Numerical mathematics and computing, Proc. 10th Manitoba Conf., Winnipeg/Manitoba 1980, Congr. Numerantium 30, S. 45–87, 1981
- [McK98] MCKAY, Brendan D.: Isomorph-free exhaustive generation. In: *Journal of Algorithms* 26 (1998), Nr. 2, S. 306–324
- [Mer99] MERINGER, Markus: Fast generation of regular graphs and construction of cages. In: *J. Graph Theory* 30 (1999), Nr. 2, S. 137–146
- [MP14] MCKAY, Brendan D. ; PIPERNO, Adolfo: Practical graph isomorphism. II. In: *J. Symb. Comput.* 60 (2014), S. 94–112
- [Rea78] READ, Ronald C.: Every one a winner or how to avoid isomorphism search when cataloguing combinatorial configurations. In: *Ann. Discrete Math.* 2 (1978), S. 107–120
- [Roy98] ROYLE, Gordon F.: An orderly algorithm and some applications in finite geometry. In: *Discrete Math.* 185 (1998), Nr. 1–3, S. 105–115
- [Sch90] SCHMALZ, Bernd: Verwendung von Untergruppenleitern zur Bestimmung von Doppelnebenklassen. (Use of subgroup ladders for the determination of double cosets). In: *Bayreuther Math. Schr.* 31 (1990), S. 109–143
- [Sch92] SCHMALZ, Bernd: *t-Designs zu vorgegebener Automorphismengruppe.*, Bayreuth. Math. Schr. 41. Universität Bayreuth, Diss., 1992
- [SFSV03] SANTO, M. D. ; FOGGIA, P. ; SANSONE, C. ; VENTO, M.: A large database of graphs and its use for benchmarking graph isomorphism algorithms. In: *Pattern Recognition Letters* 24 (2003), Nr. 8, S. 1067–1079

- [Sim71] SIMS, Charles C.: Computation with Permutation Groups. In: *Proceedings of the Second ACM Symposium on Symbolic and Algebraic Manipulation*, ACM, 1971, S. 23–28
- [Sim94] SIMS, Charles C.: *Computation with finitely presented groups*. Encyclopedia of mathematics and its applications. Cambridge University Press, 1994
- [TC36] TODD, J. A. ; COXETER, H. S. M.: A practical method for enumerating cosets of a finite abstract group. In: *Proc. Edinb. Math. Soc., II. Ser.* 5 (1936), S. 27–34
- [VH94] VERLINDE, Christophe L. ; HOL, Wim G.: Structure-based drug design: progress, results and challenges. In: *Structure* 2 (1994), Nr. 7, S. 577–587

Danksagung

Insbesondere möchte ich mich bei meinem Doktorvater Herrn Prof. Dr. Reinhard Laue bedanken. Er hat mich seit dem Augenblick, in dem ich meine Diplomarbeit bei ihm angefangen habe, tatkräftig unterstützt und immer an mich geglaubt. Dieses Vertrauen hat mir sehr dabei geholfen, beharrlich weiter nach dem Beweis zu suchen, der die mathematische Basis der drei Algorithmen dieser Arbeit bildet.

Ich danke Herrn Prof. Dr. Christian Knauer, der zugestimmt hat, sich in ein völlig neues Themengebiet einzuarbeiten um meine Arbeit zu begutachten. Seine Anregungen im Rahmen der Oberseminar-Vorträge haben den Anstoß dazu gegeben, zu zeigen, dass sich das Teilgraphenisomorphieproblem mit den drei neuen Algorithmen dieser Arbeit lösen lässt.

Ich danke Sascha Kurz, der vor vielen Jahren meine Begeisterung für die diskrete Mathematik geweckt hat. Ihm und Michael Kiermaier verdanke ich viele Anregungen und möchte mich bei beiden für die fruchtbare Zusammenarbeit bedanken. Ich danke auch Frau Elvira Rettner, die immer ein offenes Ohr und einen Kaffee zur Hand hatte, wenn die Schwierigkeiten unüberwindbar schienen.

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Ich versichere, dass zu keinem Zeitpunkt gewerbliche Promotionsberater oder Promotionsvermittler in Anspruch genommen wurden. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Nürnberg, 31. August 2015

Matthias Koch
Chlodwigstrasse 7
90431 Nürnberg